

Selected Topics in Decision Modeling
Prof. Biswajit Mahanty
Department of Industrial and Systems Engineering
Indian Institute of Technology, Kharagpur

Lecture – 35
Generic Algorithm Examples

Right, so in our subject, Selected Topics in Decision Modeling, we are in our 35th lecture that is Genetic Algorithm Examples.

(Refer Slide Time: 00:38)

GA – Example-1

Maximize (x^2+1) over $\{0,1,\dots,31\}$

- Representation : binary code e.g. 01101 for 13
- Chromosome length is 5 (11111 is 31)
- Population size: 4
- 1-point crossover
- Roulette wheel Selection

$\sum = 32$

| Randomly generated Initial Population | Binary Code | Value |
|---------------------------------------|-------------|-------|
| 1. | 01101 | 13 |
| 2. | 11000 | 24 |
| 3. | 01000 | 8 |
| 4. | 10010 | 18 |

The slide also features a graph of the function x^2+1 over the range $x \in [0, 35]$. The x-axis is labeled from 0 to 35 in increments of 5, and the y-axis is labeled from 0 to 1200 in increments of 200. The curve starts at (0,1) and rises to approximately (31, 1000). The slide footer includes the IIT Kharagpur logo, NPTEL ONLINE CERTIFICATION COURSES text, and a small video feed of the professor.

So, in the last few lectures, we have seen the Genetic Algorithm Method and the process of genetic algorithm. So, let us see some examples ah. We have also seen one example on the travelling salesman problem ah. Here are some more examples. Let us start with a simple one; that is, let us say we want to maximize x square plus 1 over a very small range that is 0 to 31.

So, when we want to do that. So, we can use let us say binary code. So, binary code here, we need a chromosome length of 5; why 5? So, chromosome length of 5 is required ah. The reason is that you know number is between 0 to 31; that means, a range of you know 32 is should be sufficient; that means, 2 to the power 5 is 32. So, using that, 2 to the power 5 equal to 32. So, we can have a chromosome length of 5. So, you see that if it is 11111, it will be 31 and 0000 will be 0. And let us take only a population size of 4 with a one point crossover and Roulette wheel selection.

So, first is we have to generate an initial population. So, let us say, we have generated an initial population 01101, this is the function and 1100001000 and 10010. So, what exactly we have done? We have flipped some coins and got this 0's and 1's. So, that is our initial population.

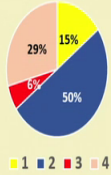
So, initial population has got 4 chromosomes, that is 13, 24, 8 and 18. So, what we do with them?

(Refer Slide Time: 02:35)





GA – Example-1: Selection

| String no. | Initial Population | X value | Fitness $f(x)=(x^2+1)$ | Prob. $f_i/\sum f_i$ | Exp. Count f_i/f_{avg} | Actual Count |
|-------------------|--------------------|---------|------------------------|----------------------|--------------------------|--------------|
| 1 | 01101 | 13 | 170 | 0.15 | 0.60 | 1 |
| 2 | 11000 | 24 | 577 | 0.50 | 2.00 | 2 |
| 3 | 01000 | 8 | 65 | 0.06 | 0.24 | 0 |
| 4 | 10010 | 18 | 325 | 0.29 | 1.16 | 1 |
| Sum $\sum f_i$ | | | 1137 | 1.00 | | |
| Average f_{avg} | | | 284.25 | .25 | | |
| Maximum | | | 577 | | | |

Roulette Wheel



■ 1 ■ 2 ■ 3 ■ 4

Let us see. So, first you know, this is our initial population. So, we compute the fitness that is the functional value in this case, then we compute the probability you know this f_i by $\sum f_i$. So, these are our fitness function value. This is the sum of all the fitness. So, if I divide, then this is the probability. So, with this probability, we constitute the roulette wheel. So, you could see that the first one, the fitness is 170, the sum is 1137. So, 170 by 1137.15 is allocated to the string number 1. So, you see, this is the string number 1.

So, this is our string number 1. So, like this, we allocate all the strings and you know this is our expected counts. So, why these are expected counts because, how many times you know out of if I do it four times, suppose we want to select 4 parents, then what is the time we get these numbers? So, you know these are 0.6, 2 basically 4 times those numbers, but actually you know what we got is 1, 2, 0 and 1. How, how exactly we did this experiment?

We again took some random number between 0 and 1. Suppose in the first time, the random number comes out to be 0.3. So, 0.3 will actually, so, 00 to 0.14, we will select the first one; 0.14 to 0.64, we will select second one. So, this you know particular number say 0.3 or something we will choose the second string. So, like typical random number experiment, we got those actual count. So, if you see the actual counts, we got one you know time the stream 1, two times the stream 2 and one time the stream 3, stream 4. So, that we will constitute our the mating population.




So, that is our mating population.

(Refer Slide Time: 05:13)

GA – Example-1

| String No. | Mating Pool | Crossover point | Offspring After Crossover | X Value | Fitness $F(x) = x^2 + 1$ |
|------------|-------------|-----------------|---------------------------|---------|-----------------------------|
| 1 | 01101 | 4 | 01100 | 12 | 145 |
| 2 | 11000 | 4 | 11001 | 25 | 626 |
| 2 | 11000 | 2 | 11010 | 26 | 677 |
| 4 | 10010 | 2 | 10000 | 16 | 257 |
| Sum | | | | | 1705 |
| Average | | | | | 426.25 |
| Maximum | | | | | 677 |

Solution improved from 577 to 677 after crossover.

So, let us see. So, this is how you know mating pool. So, look at the mating pool. The mating pool is that first one, second one is twice and the fourth one is it alright. So, arbitrarily we took some probability and we got the crossover points. So, 4 and 4 and here, 2 and 2, after that you know if I do let us say crossover between these two and these two, then this is how the crossover has been obtained.

So, how the crossover is obtained? So, look here these four is equal to these four and these 0, that is 1 and these four, that is this four and this 1, that is this. So, like this, we have done. Now here also these two has these three. So, these two has these three; so, these two, these two; these three, these three. So, that is how the crossover is made right.

Now, these are the offspring. So, look here these offspring values are 12, 25, 26 and 16. So, this is going to be our next generation and look at their fitness; their fitness is 1705 as the total fitness and the average fitness is you know 426.06. So, the maximum is 677. So, you see, earlier our maximum was you know 577. So, that was these value and here our maximum is 677.

So, you can see that neat solution has improved. The average solution has also improved which was earlier 284.25, now it is 426.25.

(Refer Slide Time: 07:20)

GA – Example-1: Mutation

- Mutation probability : $1/5=0.2$. (We can also consider :0.01, 0.1 etc.,)
- Generate random number for all 20 genes.

| String No. | Random numbers | | | | |
|------------|----------------|--------|--------|--------|--------|
| 1 | 0.6787 | 0.7577 | 0.7431 | 0.3922 | 0.6555 |
| 2 | 0.7712 | 0.7060 | 0.1318 | 0.2769 | 0.0462 |
| 3 | 0.5971 | 0.8235 | 0.6948 | 0.3171 | 0.9502 |
| 4 | 0.3644 | 0.4387 | 0.3816 | 0.7655 | 0.7952 |

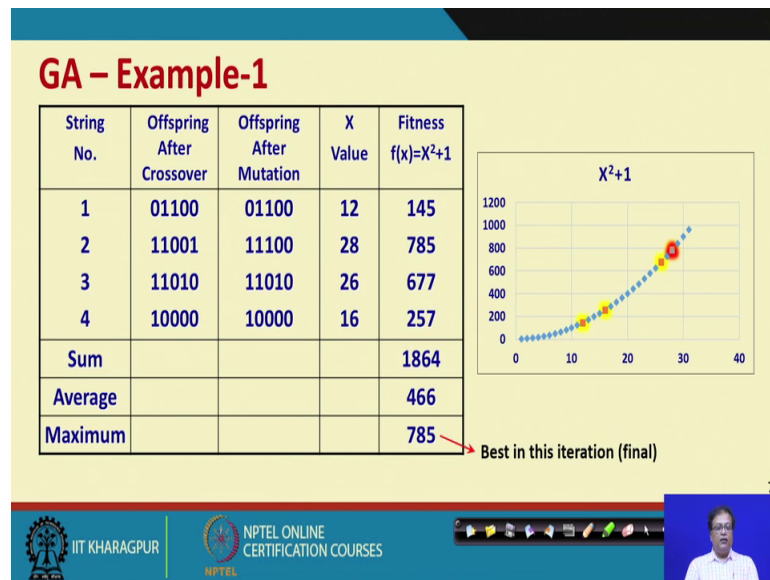
- Genes in the above highlighted places are to be flipped (0 to 1 and 1 to 0).

6

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, like this, you know this things are going on and we can also have some mutation. We took some random probabilities and see the so many random numbers are required because string 1 has 5 positions, same is 2, 3 and 4. So, assuming a mutation probability as 1 by 5, that is 0.2. So, we generate random numbers for all 20 genes and we got in two cases, the number is below 0.2. So, we do mutation out there. So, what is that mutation? 0 will be 1 and 1 will be 0.

(Refer Slide Time: 07:58)



So, like this we can carry out mutation.

So, you can see that, what are the position; the second string, third and fifth. So, this is the second string. So, the second string, third and the fifth, we had some mutation. So, what is that mutation? 0 becomes 1 and 1 becomes 0. Is it alright? So, that is how the offspring has become after mutation. So, after mutation very interestingly that the value has gone up. So, it has become 785.

So, it is not that after mutation the value will always go up you know in this particular case, it has gone up because mutation is mainly done to really improve population diversity. That is the main purpose. So, we have done that and you know that is how by the recombination operators of selection crossover and mutation the system moves from generation to generation. So, this is the essential idea of the process, the GA process.

(Refer Slide Time: 09:09)

GA – Example 2

- A supply chain of a computer manufacturing industry.
 - Demand of each customer zone is satisfied exactly by one production plant
 - Each production plant supplies to exactly one customer zone

| | Customer Zone 1 | Customer Zone 2 | Customer Zone 3 | Customer Zone 4 |
|---------|-----------------|-----------------|-----------------|-----------------|
| Plant 1 | 20 | 25 | 17 | 22 |
| Plant 2 | 14 | 9 | 19 | 6 |
| Plant 3 | 13 | 15 | 23 | 30 |
| Plant 4 | 8 | 25 | 12 | 7 |

Distance (in Km) matrix as shown
Objective function- Min. Distance

8

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, let us take another example that the second example. Suppose we have a supply chain and in this supply chain we have some you know the demands from plant to the customers, is it alright. So, we have the distances in kilometers, you know from plant one to the various customer zones and let us assume that the transportation cost would depend on those distance values, it is alright. So, objective function is minimized the distance. So, that means, plan one should supply to customer 1, 2, 3 or 4 or plant 2 will go to the other. The point is every customer zone must be having a supply; is it alright.

So, it is basically a kind of assignment problem, how do we do assignment ah. So, that the total cost is minimum. Now look here, usually we usually we try to solve such problem as maximization. So, our fitness function will not be same as objective function. So, that is the first thing to note, alright.

(Refer Slide Time: 10:23)

GA – Example 2

- Fitness value (f) of a string = $(100 - D)$
- Selection: expected count $(=f/f_{avg})$
- Crossover Procedure

Before Crossover

Pr 1 → a b c d

Pr 2 → i j k l

After Crossover

Ch 1 ← l j k i

Ch 2 ← d b c a

Permutation encoding
1-2-3-4
2-3-4-1

Random no. used for crossover
 (Selection Criteria:-
 Crossover Probability (C_p) = 0.8)

| String No | Iteration 1 | Iteration 2 |
|-----------|-------------|-------------|
| 1 | 0.98 | 0.47 |
| 2 | 0.93 | 0.55 |
| 3 | 0.45 | 0.88 |
| 4 | 0.57 | 0.93 |

Iteration 1
 String 3 and String 4 are selected for crossover as their random numbers are less than C_p .

Iteration 2
 String 1 and String 2 are selected for crossover as their random numbers are less than C_p .

So, what we do? We could make some changes. So, let us say we make the fitness function as 100 minus D alright. So, minimize D is actually means maximize 100 minus D alright. So, that is how we can do and selection my expected count that is the roulette wheel f by f average and the crossover you see. This is the kind of crossover that we thought about that suppose a, b, c, d and i, j, k, l are two parents, then child will be l, j, k, i and d, b, c, a. Basically, you know these kind of things are required because this is more like a permutation encoding, right.

So, I, have explained permutation encoding while discussing the travelling salesman problem. The permutation encoding will be something like this that 1, 2, 3, 4. So, if another string is 2, 3, 4, then fourth number should be 1. Really in assignment problem situations, it means 1, 2, 3, 4 means 1 to 1, 2 to 2, 3 to 3 and 4 to 4 and 2, 3, 4, 1 means 1 to 2, 2 to 3, 3 to 4 and 4 to 1 alright.

So, I hope you understood the crossover and mutation. So, a, b, c, d and i, j, k, l after crossover will become, l the last one should be first; first one should be last; the middle one remains as they are. So, here also a, b, c, d becomes d, b, c, a alright. Now what happens that we choose some random numbers, let us say crossover probability is 0.8. So, you know we have some probabilities and we see that the string 3 and 4 are selected for crossover in iteration 1; that is these two because the probability is lower than 0.8.

And here string 1 and 2; they will be selected for crossover because their random number is less than the crossover probability. So, this is how we make use of crossover probability to choose whether to go for crossover or not, right.

(Refer Slide Time: 12:55)




GA – Example 2

Initial Population

| String | Details | Objective Function Value (D) | Fitness Function Value (100-D) |
|----------|---------|------------------------------|--------------------------------|
| String 1 | 1-2-3-4 | $20 + 9 + 23 + 7 = 59$ | $100 - 59 = 41$ |
| String 2 | 2-4-1-3 | $14 + 25 + 17 + 30 = 86$ | $100 - 86 = 14$ |
| String 3 | 3-1-4-2 | $13 + 25 + 12 + 6 = 56$ | $100 - 56 = 44$ |
| String 4 | 4-3-2-1 | $8 + 15 + 19 + 22 = 64$ | $100 - 64 = 36$ |

| | Cust Zone 1 | Cust Zone 2 | Cust Zone 3 | Cust Zone 4 |
|---------|-------------|-------------|-------------|-------------|
| Plant 1 | 20 | 25 | 17 | 22 |
| Plant 2 | 14 | 9 | 19 | 6 |
| Plant 3 | 13 | 15 | 23 | 30 |
| Plant 4 | 8 | 25 | 12 | 7 |

10

So, with that now let us see these are the strings. So, this is the distance values and 100 minus D, those are the fitness function. So, this was the original matrix.

So, what we did for all the strings? We compute the fitness values, very simple; simply look at that the first one, let us see what happens to the first one. This is the distance 20, then 9, then 23 and 7; just add them 1 to 1, 2 to 2, 3 to 3, 4 to 1. So, total distance will be 59. But had it been 1 to 2, so, 25, then 4, you know the customer zones, 4, 2 to 4 need not that way I think this is 14, so; that means, 2 to 1, then 25, then your 17, the third one, that 3 should be on 17 and 30; 2 to 4 actually, this 25 and this 17, not this one right; so, 14, 25, 17 and 13.

So, like this, we compute those distances and 100 minus D, that will give us the fitness function value. So, these are our fitness function values for those four strings right. So, having obtained that, now let us see what we get.


(Refer Slide Time: 14:37)

GA – Example 2


Initial Population

| String | Details | Objective Function Value (D) | Fitness Function Value (100-D) | Expected Count |
|----------|---------|------------------------------|--------------------------------|------------------|
| String 1 | 1-2-3-4 | $20 + 9 + 23 + 7 = 59$ | $100 - 59 = 41$ | $1.21 \approx 1$ |
| String 2 | 2-4-1-3 | $14 + 25 + 17 + 30 = 86$ | $100 - 86 = 14$ | $0.41 \approx 0$ |
| String 3 | 3-1-4-2 | $13 + 25 + 12 + 6 = 56$ | $100 - 56 = 44$ | $1.30 \approx 2$ |
| String 4 | 4-3-2-1 | $8 + 15 + 19 + 22 = 64$ | $100 - 64 = 36$ | $1.07 \approx 1$ |
| | | | $f_{total} = 135$ | |
| | | | $f_{avg} = 33.75$ | |



11



IIT KHARAGPUR



NPTEL ONLINE CERTIFICATION COURSES



So, this is our strings, this is the objective function, these are the fitness function, then we get the f_{total} and $f_{average}$. So, if I divide by f_{total} by $f_{average}$, you know, then we get and multiply by 4, we get the expected counts. So, we got some expected counts for the initial population.

So, in this case, we made a simplified system that depending on the expected count, we find how many should be the number of strings. So, you please understand that every genetic algorithm is different and it is up to the design that to follow different procedures that is different selection, crossover or mutation methods right. So, there will be variations of these types, alright.

(Refer Slide Time: 15:33)

GA – Example 2

Mating Pool

| String | Details |
|----------|---------|
| String 1 | 1-2-3-4 |
| String 2 | 3-1-4-2 |
| String 3 | 3-1-4-2 |
| String 4 | 4-3-2-1 |

Crossover Between String 3 and String 4

P1: 3 1 4 2
P2: 4 3 2 1

Ch₁: 1 3 2 4
Ch₂: 2 1 4 3

After Crossover

| String | Details |
|----------|---------|
| String 1 | 1-2-3-4 |
| String 2 | 3-1-4-2 |
| String 3 | 1-3-2-4 |
| String 4 | 2-1-4-3 |

12

IIT KHARAGPUR NPTEL ONLINE CERTIFICATION COURSES

So, then after that, these are our mating pool. You see the string 2 and 3 are the same. This is nothing but the same 3-1-4-2 has got two copies; so, two copies; 1-2-3-4, one copy and 4-3-2-1, one copy. So, this is our the mating pool and through this mating pool, we did crossover. So, 1 and 2, we choose for crossover sorry 3 and 4.

So, 3 is 3, 1, 4, 2 and 4 is 4, 3, 2, 1. So, after crossover, what they will become see this 4, 3, 2, 1 becomes 1, 3, 2, 4; can you see that? These 1 has come here, 4 has gone there and this string, this 2 has come here and this 3 has gone there; is it alright. So, that is how the parents and the childs are obtained. So, like this, we have now some new strings that is 1- 2-3-4, 3-1-4-2; these two remains as they are, you know they are replicated. They remain in the new population and these 1, 2, 3, 1-3-2-4 and 2-1-4-3, they join in the string right. So, having obtained this, that is after crossover then we see what is the new situation.

So, these are our new strings; that is 1-2-3-4, 3-4-2 1-3-2-4 and 2-1-4-3 and again we compute the D and 100 minus D by the procedure already shown. So, this is our new fitness function values. So, same thing is replicated once again and look here, this time f average is 35.75; look what was in our earlier case, in the first generation, it was 33.75. So, after crossover, we found that I mean when the new generation is obtained, the new generation the having a higher average fitness right.

So, once again, we compute those expected counts. Basically, 41 into you know by 143. So, that is if I divide 41 by f average, then I get those expected counts and from the expected counts, we can see some numbers; maybe we can obtain through random numbers also. So, that will be for our next crossover.

In this case there is no mutation considered, alright. So, once again, by same procedure we do the, from the mating pool and crossover and we get a new set of you know strings. So, all these new set of strings, once again we compute that D and 100 minus D , the fitness function values. So, this time if you look then there new fitness values have become 35 . So, this case, what has happened? These fitness value have slightly reduced right. So, it is not always that the procedure has to guarantee that average fitness value will increase, but overall, the solution will be better. So, but then, what is more interesting is that let us see what was the best possible string. That is also very important to see.

So, you see the best fitness value here was 44 in the first generation, in the initial population. Now, at the second also, we had 44 right, but after the third generation, you know also 44 .


(Refer Slide Time: 19:23)

GA – Example 2


Iteration 2

| String | Details | Objective Function Value (D) | Fitness Function Value (100-D) | Expected Count |
|----------|---------|------------------------------|--------------------------------|------------------|
| String 1 | 1-2-3-4 | $20 + 9 + 23 + 7 = 59$ | $100 - 59 = 41$ | $1.14 \approx 1$ |
| String 2 | 3-1-4-2 | $13 + 25 + 12 + 6 = 56$ | $100 - 56 = 44$ | $1.23 \approx 2$ |
| String 3 | 1-3-2-4 | $20 + 15 + 19 + 7 = 61$ | $100 - 61 = 39$ | $1.03 \approx 1$ |
| String 4 | 2-1-4-3 | $14 + 25 + 12 + 30 = 81$ | $100 - 81 = 19$ | $0.53 \approx 0$ |
| | | | $f_{total} = 143$ | |
| | | | $f_{avg} = 35.75$ | |



14



IIT KHARAGPUR



NPTEL ONLINE CERTIFICATION COURSES

(Refer Slide Time: 19:24)


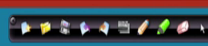

GA – Example 2

Iteration 3

Maximum fitness value = 44
Best solution is 56.

| String | Details | Objective Function Value (D) | Fitness Function Value (100-D) | Expected Count |
|----------|---------|------------------------------|--------------------------------|------------------|
| String 1 | 2-1-4-3 | $14+25+12+30 = 81$ | $100-81 = 19$ | $0.54 \approx 0$ |
| String 2 | 4-2-3-1 | $8+9+23+22 = 62$ | $100-62 = 38$ | $1.08 \approx 1$ |
| String 3 | 3-1-4-2 | $13+25+12+6 = 56$ | $100-56 = 44$ | $1.25 \approx 2$ |
| String 4 | 1-3-2-4 | $20+15+19+7 = 61$ | $100-61 = 39$ | $1.11 \approx 1$ |
| | | | $f_{\text{total}} = 140$ | |
| | | | $f_{\text{avg}} = 35$ | |

17



So, we could not get a better string than a fitness value 44 even after three iterations at the third iteration, maybe we could have got a better string in a in further operations because it is a very simple problem. In a large problem, we go for many generations ah. We have to have a computer installation of such problems and we may get better solutions alright. So, we saw at least one assignment problem and a functional value for how to solve by GA.

Let us see some examples about the Knapsack problem.


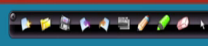

(Refer Slide Time: 20:13)

Knapsack Problem

- Things are available in different sizes and values. Fill up the knapsack of given size so as maximize its value.
- Example: The knapsack of size 50 to be filled with as many of the following:

| SINo | Things | Size | Value |
|------|--------|------|-------|
| 1. | a | 11 | 22 |
| 2. | b | 5 | 12 |
| 3. | c | 7 | 18 |
| 4. | d | 12 | 31 |
| 5. | e | 21 | 39 |
| 6. | f | 18 | 32 |

18



So, we have already seen Knapsack problem in dynamic programming. We have also seen in integer programming. So, it is known to us, what is a knapsack problem? There are certain things. So, like a, b, c, d, e, f and they have different sizes and different values. So, total size is fixed let say 50 and that is to be filled so as to maximize the value. That is the usual knapsack problem.

(Refer Slide Time: 20:46)

Knapsack 0-1 Problem

Knapsack size 50

Encoding: Binary 0-1

How many genes in a chromosome?

Formulation:

$$\text{Maximize } \sum_{i=1}^n v_i x_i$$

$$\text{s.t. } \sum_{i=1}^n s_i x_i \leq 50$$

What should be the Fitness Function?

| SlNo | Things | Size | Value |
|------|--------|------|-------|
| 1. | a | 11 | 22 |
| 2. | b | 5 | 12 |
| 3. | c | 7 | 18 |
| 4. | d | 12 | 31 |
| 5. | e | 21 | 39 |
| 6. | f | 18 | 32 |

Penalty function approach

19

So, what is it? That see, assume that it is a binary knapsack that is 0 or 1 should only be selected. So, if that is so, then we have an objective function maximize sigma i equal to 1 to n v i x i subject to s i x i less than equal to 50. So, what should be the fitness function. So, the maximize part is fine. Now, who will check this constraint every time we select and do crossover and mutation. Say the point is, if you leave it to the process if you think that fine I have got the string and I have this fitness and I will keep checking, you know every time I do selection crossover and mutation, it will be too cumbersome.

So, a better method that is called a Penalty function approach, a Penalty function approach. So, this approach essentially, what it does right? It actually adds you know this constraint in the objective function itself and converts the constraint problem to an unconstrained optimization problem. And however, you know it also means that it allows some infeasible solutions in the solution space.

But their fitness value should be very very low, is it alright. So, how because so that, they are so unfit you know their fitness values are so low that they are chances of getting

selected in the next generation with crossover or mutation is very very low is it alright. So, how exactly this is implemented?

(Refer Slide Time: 22:48)

Knapsack 0-1 Problem: Fitness Function

Fitness Function: Convert the constraint problem to an unconstrained problem

How? Use a penalty function!

Fitness Function: Maximize $\sum_{i=1}^n v_i x_i + \text{Min}(0, 1000 * (50 - \sum_{i=1}^n s_i x_i))$

- Chromosome A 101100
- Chromosome B 110111

Handwritten notes on the slide:

- $\sum_{i=1}^n s_i x_i <= 50$ (Constraint)
- $s_i x_i = 20$
- $1000 * (50 - \sum_{i=1}^n s_i x_i) = 1000 * (50 - 20) = 30000$
- $s_i x_i = 60$
- $1000 * (50 - \sum_{i=1}^n s_i x_i) = 1000 * (-10) = -10000$
- $\sum_{i=1}^n v_i x_i + \text{Min}(0, 1000 * (50 - \sum_{i=1}^n s_i x_i))$

So, look here. This is an example what we have done? You know we have created a new fitness function as $\sum v_i x_i + \text{minimum of } 0 \text{ and a penalty say } 1000 \text{ could be any other value } 50 - \sum s_i x_i$. So, you see how it works. Assume that $\sum s_i x_i$ equal to 20, alright.

So, $\sum s_i x_i$ equal to 20. So, what will be the value of $1000 \text{ into } 50 - \sum s_i x_i$. It will be $1000 \text{ into } 50 - 20$. So, it will be 30000. Can you see that. So, it will become 30000 alright ah. Now assume another example; suppose $\sum s_i x_i$ equal to 60. So, what this term $1000 \text{ into } 50 - \sum s_i x_i$ will become $1000 \text{ into } 50 - 60$ to minus 10000.

So, just see I have given two examples. In the first example, if $\sum s_i x_i$ is 20, that is within the constraint, the constraint was $\sum s_i x_i$ should be less than 50, is it not that was our constraint. So, in the first instance, that is $\sum s_i x_i$ yeah. So, in all these cases, it is actually the sum over, the sum over this missed. So, you know this please put that sum right. So, this is the sum; that means, the total of total value total sorry total I mean wait or something should be within 50. So, that is the sum. So, that is well.

So, you see when the value is 20; that means, within 50 these evaluates to 30000 and when it is above 50; that means, not within 50, it evaluates to minus 10000. In the first

case, when the it does not cross the limits it evaluates to 0 because 0 and 30000, minimum is 0; that means, these term will become nothing but $v_i x_i$; is it not in the these case. In the first case, in the second case what will be these term; these term will be $\sum v_i x_i$ minus 10000, can you see that. So, the fitness function is reduced by a huge margin right. Those strings are going to have very very low values; that means, they are actually infeasible because they cross the weight but they will be still considered, but with a very low fitness value. So, this is exactly how these problems are treated.

Suppose we have these two chromosomes.

(Refer Slide Time: 26:39)

Knapsack 0-1 Problem: Crossover and Mutation

Single point crossover - one crossover point is selected – first part from first parent and second part from the second parent.

- P1: 101|100; P2: 110|111 =>> Child1: 101|111; Child2: 110|100

Two point crossover:

- P1: 10|11|00; P2: 11|01|11 =>> Child1: 10|01|00; Child2: 11|11|11

Mutation: Bit inversion 100100 =>> 110100

21

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, what happens, we do single point crossover. So, we you know already how it is. So, at the third point is the crossover point, so, 101 and 111 and 110 and your 100 right. So, those are two childs. And in two point crossover, if you do then the middle portion will be interchanged and the mutation will be simple bit inversion; is it alright. So, that is how the crossover and mutation is done.




(Refer Slide Time: 27:11)

Knapsack Problem 2

- Things are available in different sizes and values in a number. How many of each can be filled up in the knapsack of given size so as maximize its value.
- Knapsack size is 150.

| SINo | Things | Number | Size | Value |
|------|--------|--------|------|-------|
| 1. | a | 9 | 4 | 12 |
| 2. | b | 9 | 5 | 15 |
| 3. | c | 9 | 6 | 19 |
| 4. | d | 9 | 7 | 20 |

22



Ah let us take another problem which is not a 0, 1 problem. So, there are things a, b, c, d, they are in different numbers and the knapsack size is 150; the values and sizes are there ah; here how do we go about. So, one thing you understand that since it is not a 0, 1 programming, your binary encoding is not going to work. So, what we have to do is we have to do real encoding right.

(Refer Slide Time: 27:43)

Knapsack Problem 2: Encoding




- Encoding: Real Encoding

| | | | |
|---|---|---|---|
| 4 | 8 | 5 | 3 |
|---|---|---|---|
- It means 4 a's, 5 b's, 6 c's, and 7 d's.
- Fitness Function will be as before.
- Single point Crossover and Single Mutation may be used.

| SINo | Things | Nos | Size | Value |
|------|--------|-----|------|-------|
| 1. | a | 9 | 4 | 12 |
| 2. | b | 9 | 5 | 15 |
| 3. | c | 9 | 6 | 19 |
| 4. | d | 9 | 7 | 20 |

Question: What if nos. available are not 9 each?
Hint: Use a suitable penalty function. Why and How?

23



So, how real encoding is done? So, you see you can have something like four digit 4, 8, 5, 3 which means 4 a's, 5 b's, 6 c's and 7 d's alright. Fitness function will be as before and single point cross over and single mutation may be used, right.

So, what if the available are not 9? Then use a suitable penalty approach is it not so supposing if they are within 0 and 9 that is fine. But supposing the number is not 9 number is more than 9 then probably we have to use 2 digits each, right, right, we not necessarily penalty function all the time. We might have the real encoding or 2 bits each. So, the problem has to might have to be encoded in more than one bit for each item fine.

(Refer Slide Time: 28:41)

Tree Encoding

- Tree encoding is used mainly for evolving programs or expressions. Every chromosome is a tree of some objects, such as functions or commands in programming language.

$(+ x (/ 5 y))$

$(do_until\ step\ wall)$

24

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

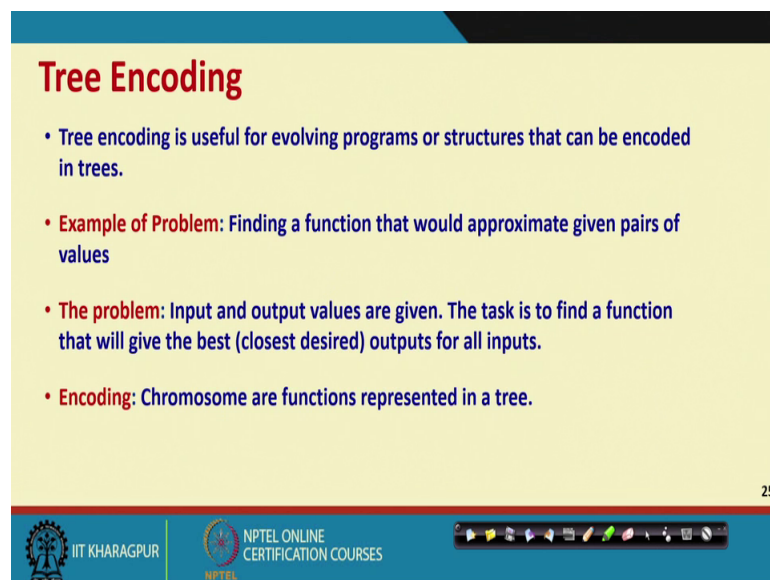
So, this is how now in this lecture, let us also see something very interesting that is called Tree Encoding. See this Tree Encoding is sometimes what happens supposing we have an objective function and we really want a formula for a function, is it alright. So, we want a formula for a function. So, basically, we want a particular value to be maximum some fitness function value should be maximum but we want to know what is that formula.

We have to actually discover the formula, so, something very interesting. So, we can do really tree encoding. So, tree encoding is used mainly for evolving programs or expressions. Every chromosome is a tree of some objects such as functions or commands in programming language. So, it could be a programming language also, something like do until, step wall, something like this.

But this can be read as plus x divided by 5 y, is it alright. So, plus x divided by 5 y. So, you see this is like a formula. So, if you add further trees, you can also really write a formula. So, this is the encoding, the encoding gives a formula right. So, and this string then can be evaluated based on the given x and y values and see what kind of fitness function we are obtaining.

So, sometimes like you know those empirical relationships, we have to obtain. Suppose, we know something depends on three values, we really do not know what is the relationship. We have to discover the relationship; we only know the functional values. So, from the functional values we can discover the best possible empirical relationships by going for tree encoding; is it alright.

(Refer Slide Time: 30:39)



Tree Encoding

- Tree encoding is useful for evolving programs or structures that can be encoded in trees.
- **Example of Problem:** Finding a function that would approximate given pairs of values
- **The problem:** Input and output values are given. The task is to find a function that will give the best (closest desired) outputs for all inputs.
- **Encoding:** Chromosome are functions represented in a tree.

25

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, finding a function that would approximate given pair of values, input and output values are given, the task is to find the function that we give the best or the closest desired outputs for all inputs. Chromosomes are functions represented in a tree.

(Refer Slide Time: 30:58)

Crossover and Mutation for Tree Encoding

Tree crossover – a crossover point is selected in both parents, parents are divided in that point and the parts below crossover points are exchanged to produce new offspring.

Mutation: Changing operator, number - selected nodes are changed.

26

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

And look how the crossover is done.

So, if we have two parents, you know this is a crossover point for parent A and parent B. So, what happens, that you know these portion gets replaced by this. So, we get a new offspring where it is earlier to us x into divide a x and divided by y plus 3 ah. Here, it will become x divided by y square; is it alright. So, that is the kind of interesting things that we can do with the tree encoding right. So, this could be a little more complex, but you know just to mention that we also have Tree Encoding right. So, I stop here and,

Thank you very much.