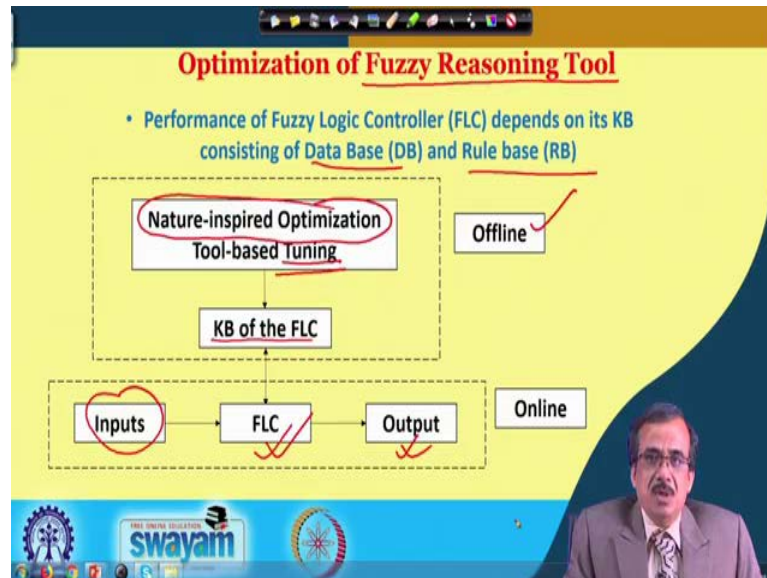


**Fuzzy Logic and Neural Networks**  
**Prof. Dilip Kumar Pratihari**  
**Department of Mechanical Engineering**  
**Indian Institute of Technology, Kharagpur**

**Lecture – 18**  
**Optimization of Fuzzy Reasoning and Clustering Tool (Contd.)**

(Refer Slide Time: 00:15)



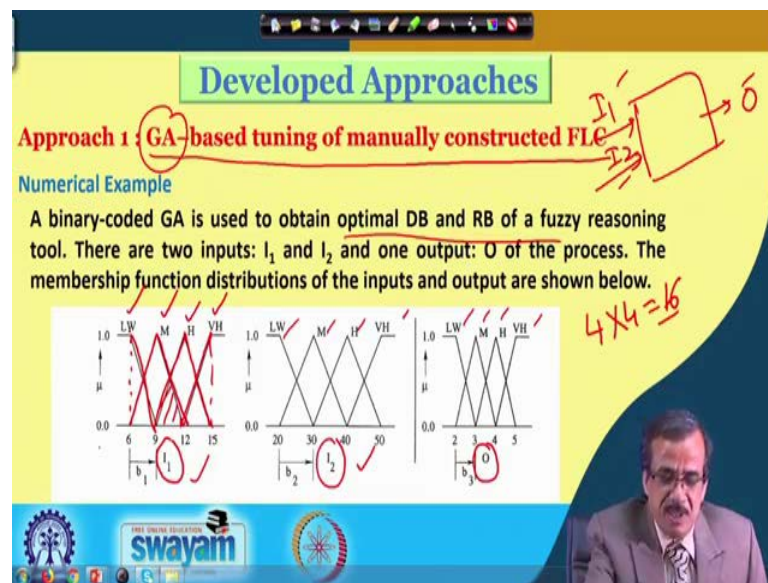
Now, we are going to discuss, how to carry out optimization to achieve one optimized fuzzy reasoning tool. Now, we have seen that the performance of a fuzzy reasoning tool or a fuzzy logic controller depends on its knowledge base, which is nothing, but a collection of its database and rule base. Now, let us see the scheme, how can you optimize or how can you tune this particular knowledge base of the fuzzy logic controller or fuzzy reasoning tool, so that we can model the input-output relationships of a process as accurately as possible.

Now, here what we do is, this knowledge base of the fuzzy logic controller, that is the database and the rule base, we try to optimize with the help of a nature-inspired optimization tool. Now, during this training or the tuning, what we will have to do is, we will have to take the help of some known input-output relationships and that is known as the training scenarios. Now, this nature-inspired optimization tool is an iterative method and it takes a huge amount of time just to converge to the optimal solution. And, that is why, this particular training or the tuning of the parameters is carried out offline. Now,

once you have got the optimized knowledge base of the fuzzy logic controller, now we can pass the set of inputs through the fuzzy logic controller.

So, we will be getting this particular output, online, might be within a fraction of second for a set of inputs. So, we will be getting that particular the output and that is why, once it is trained offline and now, we are in a position to use it online. Now, let us see how does it work.

(Refer Slide Time: 02:35)



Now, to explain this, we are going to take the help of one numerical example, now there are several approaches of optimizing. So, this particular fuzzy reasoning tool or fuzzy logic controller, the first approach is known as the GA-based tuning of manually constructed FLC.

So, what we do is, supposing that I want to determine the input-output relationships of a particular process. Now, what we do is, we try to design the knowledge base which is nothing, but a collection of database and rule base manually. So, based on the information we have we try to design manually first, but that may not be the optimal in any sense. And, after that, we are going to take the help of an optimizer, it is a genetic algorithm, so that we can find out the optimal knowledge base for this particular fuzzy logic controller.

Now, as I told that we are going to take the help of one binary-coded genetic algorithm here. So, a binary-coded genetic algorithm is used to obtain optimal database and rule base of a fuzzy reasoning tool and let me consider a process having say 2 inputs and 1 output, the two inputs are  $I_1$  and  $I_2$  and we have got one output, that is denoted by  $O$ . The membership function distribution of the inputs:  $I_1$  and  $I_2$  and the output  $O$  are assumed to be triangular in nature, for simplicity.

So, this shows the membership function distribution for the first input, that is  $I_1$  and this shows the membership function distribution for the second input, that is  $I_2$  and this shows the membership function distribution for the output, that is  $O$ . Now, both the inputs and the output actually are represented with the help of four linguistic terms each, for example, say the linguistic terms are low, medium, high and very high. Now, there are four linguistic terms for  $I_1$  and four linguistic terms for  $I_2$ .

So, I will have 4 multiplied by 4, there are 16 possible combinations for the inputs and we are going to consider the 16 rules and each rule is nothing, but the relationship between the inputs and the output. Now, as I told that for simplicity, we have considered that membership function distribution is triangular. So, for this medium and high, we have considered isosceles triangle and there is overlapping region also. And, for this low and medium in fact, we are going to consider some sort of say right angled triangle and similarly, for this  $I_2$  and  $O$ .

So, we are having the membership function distribution for your the low, medium, high and very high. And, here, once again low, medium high and very high and once you have got the membership function distribution, as I told, now, we are in a position to design the rule base.

(Refer Slide Time: 06:23)

The manually-constructed RB is given below

		$I_2$				
		LW	M	H	VH	
$I_1$	LW	LW	LW	M	H	
	M	LW	M	H	H	
	H	M	M	H	VH	
	VH	M	H	VH	VH	

16 X 16  
If  $I_1$  is LW AND  $I_2$  is LW  
then O is LW.

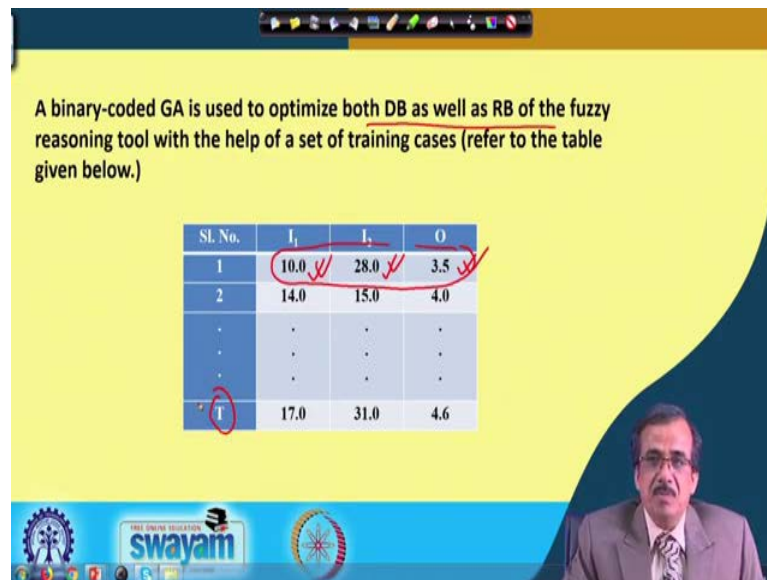
Now, here, we have got the 16 rules. Now, the 16 rules can be read as follows: For example, the first rule could be so if  $I_1$  is say low. So,  $I_1$  is low and  $I_2$  is your low. So,  $I_2$  is low then your output  $O$  is also low. So, this is actually the first rule, similarly we can design all the 16 rules, now all these are manually constructed 16 rules.

Now, the designer will design this particular rule base based on his or her experience of that particular problem. Now, as I told that a particular rule is nothing, but the relationship between the inputs and output. And, let me read once again the first rule and which is as follows: if  $I_1$  is low and  $I_2$  is low then output  $O$  is low and similarly we have got say 16 such rules.

(Refer Slide Time: 07:51)

A binary-coded GA is used to optimize both DB as well as RB of the fuzzy reasoning tool with the help of a set of training cases (refer to the table given below.)

Sl. No.	$I_1$	$I_2$	O
1	10.0 ✓	28.0 ✓	3.5 ✓
2	14.0	15.0	4.0
...	...	...	...
T	17.0	31.0	4.6




Now, I am still continuing with the statement of this particular numerical example. So, here once again let me tell that a binary-coded GA, we will be using to optimize both the database as well as the rule base of this fuzzy reasoning tool. And, here, we have got the set of training cases and a particular trading scenario is nothing, but the relationship between the inputs and output.

For example, say the first training scenario is nothing, but if  $I_1$  is 10 with some units and  $I_2$  is 28 with some other units, then the output is nothing, but 3.5 with some other in unit. So, this is the way actually, we can design and we can collect in fact, the known input output relationship and that is nothing, but the set of training cases or the set of training scenario. Now, here we have got say capital T number of training scenarios.

(Refer Slide Time: 09:05)

An initial population of the BCGA is created at random, as shown below

Sl. No.	GA-string
1	101100110111011100010101011001
2	011001011011010001010110110010
⋮	⋮
N	101000111010111010010011011011



Now, once you got this particular training scenario, now, actually we will have to design that particular GA-string, the GA string for the binary-coded GA. Now, this particular GA-string will carry information of all the design variables, now, what are the design variables, let me have a look fast, then I will concentrate here.


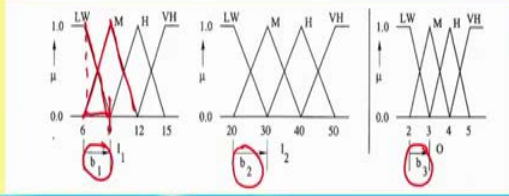
(Refer Slide Time: 09:29)

### Developed Approaches

**Approach 1 : GA-based tuning of manually constructed FLC**

**Numerical Example**

A binary-coded GA is used to obtain optimal DB and RB of a fuzzy reasoning tool. There are two inputs:  $I_1$  and  $I_2$  and one output:  $O$  of the process. The membership function distributions of the inputs and output are shown below.



Now, one of the design variables could be your this particular  $b_1$ . So,  $b_1$  is going to represent, whether this particular triangle will be is a stiffer one or it will be a flatter one. For example, say if it is a right-angled triangle, this indicates the base width of this right

angled triangle and if it is isosceles triangle. So, this  $b_1$  indicates actually the half base width of this particular isosceles triangle.

Similarly, for this particular  $I_2$ ,  $b_2$  is actually the variable. So,  $b_2$  could be large or it could be small, similarly for this particular output. So,  $b_3$  could be your the design variables. Now, we will have to assign some bits to represent; so, this  $b_1$ ,  $b_2$  and  $b_3$ , let me assume that I am assigning. So, 5 bits to represent each of the variables like your  $b_1$ ,  $b_2$  and  $b_3$ .

(Refer Slide Time: 10:41)

The manually-constructed RB is given below

		$I_2$			
		LW	M	H	VH
$I_1$	LW	LW	LW	M	H
	M	LW	M	H	H
	H	M	M	H	VH
	VH	M	H	VH	VH

Handwritten notes on the slide:

$$5 + 5 + 5 = 15 (b_1, b_2, b_3)$$

$$+ 16 (RB)$$

$$16, 0 = 31$$

Now, once you have assigned. So, some bits like 5 bits each to  $b_1$ ,  $b_2$  and  $b_3$ . So, now, we have got, in fact, your for 3 variables, 5 plus 5 plus 5; so 15 bits to represent  $b_1$ ,  $b_2$  and  $b_3$ . Now, I have to represent the rule base, how to represent the rule base? I have got 16 rules, now to represent, in fact, the presence or absence of a rule, we use either 1 or 0; now if it is 1, it means that that particular rule is present and if it is 0, the rule is absent.

Now, what you do is, this indicates actually the rule base. So, we concentrate on the left most top corner. So, we start with this, then we just move in this particular direction, next we move in this particular direction, next we move, in fact, in this particular direction, next we move this particular direction to represent that particular the rule and what you do is. So, if there is 1 here means that particular rule is present, if there is a 0



here means that particular output is absent, that means your that particular output is absent, and so on.

So, there are 16 such rules. So, I need, in fact, 16 bits just to represent, whether the rule is present or not. So, the GA-string will consist of 15 for this b\_1, b\_2 and b\_3 ( this is for b\_1, b\_2 and b\_3) plus 16 for the rule base. So, GA-string will be your 31 bits long. So, the same thing actually, I am just going to represent here.

(Refer Slide Time: 12:43)

An initial population of the BCGA is created at random, as shown below

Sl. No.	GA-string
1	1011001101110111000101010111001
2	0110010110110100010101110110010
...	...
N	1010001110101110100100110111011

Handwritten annotations on the slide: Red arrows point to the first 5 bits of the first two rows, labeled 'b1'. Red arrows point to the next 5 bits of the first two rows, labeled 'b2'. Red arrows point to the next 5 bits of the first two rows, labeled 'b3'. A red arrow points to the last 16 bits of the first row, labeled 'RB'.

Now, if you see the GA-string, this particular population, which are generated at random, the first 5 bits are going to represent the b\_1, the next 5 bits are going to represent in fact, your b\_2, the next 5 bits are going to represent b\_3 and the remaining 16 bits are going to represent the rule base.

So, the rule base is represented by these 16 bits, this is going to represent b\_1, this is going to represent b\_2 and this is going to represent b\_3, and the GA-string will be your 31 bits long. And, similarly, we have got the whole population of solutions generated at random. Now, once you have got this, now let us see, how can it optimize, how can GA optimize that particular database and the rule base.



(Refer Slide Time: 13:49)

Starting from the left most bit-position five bits are assigned to represent each of the  $b$  values (that is,  $b_1$ ,  $b_2$  and  $b_3$ ) and the next 16 bits represent the RB of the fuzzy reasoning tool. Determine the deviation in prediction for the first training case by using the first GA-string. The  $b$  values are assumed to vary in the ranges given below.

$2.0 \leq b_1 \leq 4.0$  ✓  
 $5.0 \leq b_2 \leq 15.0$  ✓  
 $0.5 \leq b_3 \leq 1.5$  ✓

swayam

Now, before we proceed further, let me finish the statement of the problem. Now in fact, our aim is to determine the deviation in prediction for the set of training scenarios; that means, once it is trained, we are going to pass a set of inputs at the test scenario.

And, we will try to find out, what should be the output and how much is the deviation during the optimization, so these are the ranges for your  $b_1$ ,  $b_2$  and  $b_3$ . Now, these are all real variables and we will have to define the ranges for your  $b_1$ ,  $b_2$  and  $b_3$ . So, this completes actually the statement of the problem. Now, let us see, how to find out the solution for this particular problem.

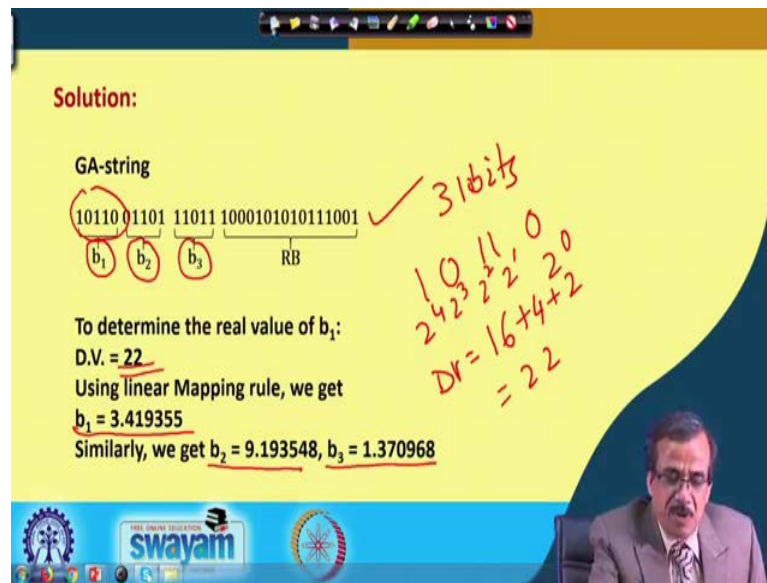
(Refer Slide Time: 14:49)

**Solution:**

GA-string  
1011001101 11011 1000101010111001  
 $b_1$   $b_2$   $b_3$  RB

To determine the real value of  $b_1$ :  
D.V. = 22  
Using linear Mapping rule, we get  
 $b_1 = 3.419355$   
Similarly, we get  $b_2 = 9.193548$ ,  $b_3 = 1.370968$

Handwritten notes:  
31 bits  
1 0 1 1 0  
2<sup>4</sup> 2<sup>3</sup> 2<sup>2</sup> 2<sup>1</sup> 2<sup>0</sup>  
Dr = 16 + 4 + 2  
= 22

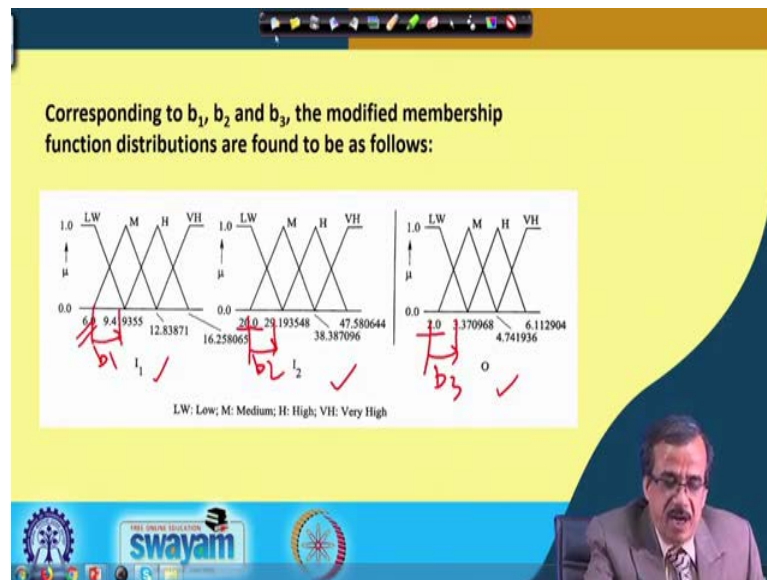


Now, this particular method, I have already discussed a little bit. So, let me discuss once again, let me concentrate on the first GA-string, which is 31 bits long. So, this particular GA-string is 31 bits long. So, first five bits are going to represent  $b_1$ , the  $b_2$  is going to be represented by the next 5 bits, the next 5 bits are going to represent  $b_3$  and the 16 bits are going to represent that particular the rule base. Now, we have already discussed like how to find out the decoded value, now if I try to find out for 1 0 1 1 0.

So, this displays the values: 2 raise to the power 0, 2 raise to the power 1, 2 raise to the power 2, 2 raise to the power 3, 2 raise to the power 4. So, the decoded value will be your 2 raise to the power 4 is nothing, but 16, then 2 raise to the power 2 is nothing, but 4 plus 2 raise to the power 1 is nothing, but 2. So, this is nothing, but 22. So, this is nothing, but is your the decoded value.

And, once you have got the decoded value and once again, we will have to use that linear mapping rule, which has already been discussed and using that linear mapping rule, I can find out what should be the real value for this particular  $b_1$ , that is the first design variable. Now by following the similar procedure, I can also find out, what is  $b_2$ , then what is  $b_3$  and once we got the real values for this  $b_1$ ,  $b_2$  and  $b_3$ .

(Refer Slide Time: 16:41)



So, now in fact, we are in a position to find out like what should be the modified membership function distribution. So, the modified membership function distribution will look like this. So, the starting value for  $I_1$  we keep it fixed, similarly the starting value for  $I_2$  is kept fixed, starting value of output is kept fixed.

Now, we are going to find out, what should be the modified value for this particular your  $b_1$  and depending on the values of this  $b_1$ ,  $b_2$  and  $b_3$ . So, this is your  $b_2$  and this is nothing, but is your  $b_3$ . So, we redraw the modified membership function distribution. So, the modified membership function distribution for  $I_1$ ,  $I_2$  and  $O$  will look like this, and once you have got the modified membership function distribution.

(Refer Slide Time: 17:43)

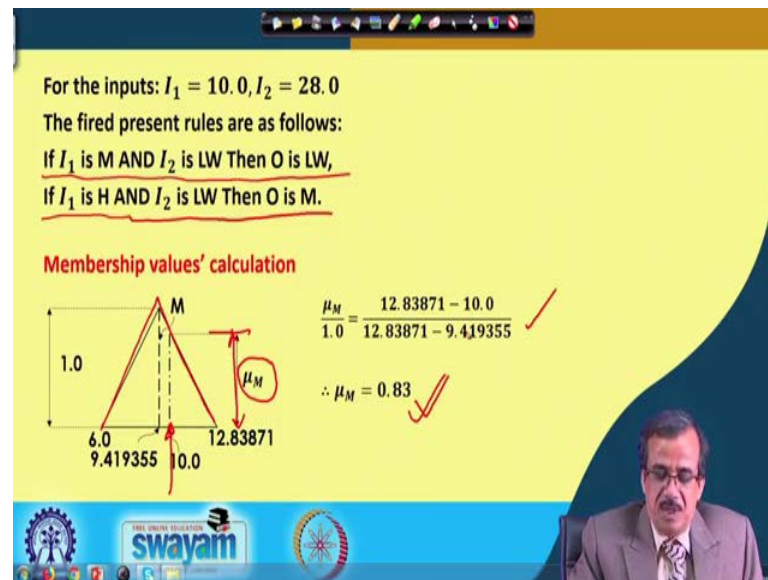
Corresponding to the sub-string: 1000101010111001,  
the RB is found to be as follows:

		$I_2$			
		LW	M	H	VH
$I_1$	LW	LW 1 0 0 0			
	M	LW 0	1	H 0	
	H	M 0	1	H 1	VH
	VH	M 0	0	1	VH

Now, we are going to discuss like how to select the good rules with the help of the GA string, supposing that these 16 bits lying on the first GA-string are going to represent the presence and the absence of the rules. Now, this I have already discussed that these represent actually the output of the rules. So,  $I_1$  has got 4 linguistic terms,  $I_2$  has got 4 linguistic terms.

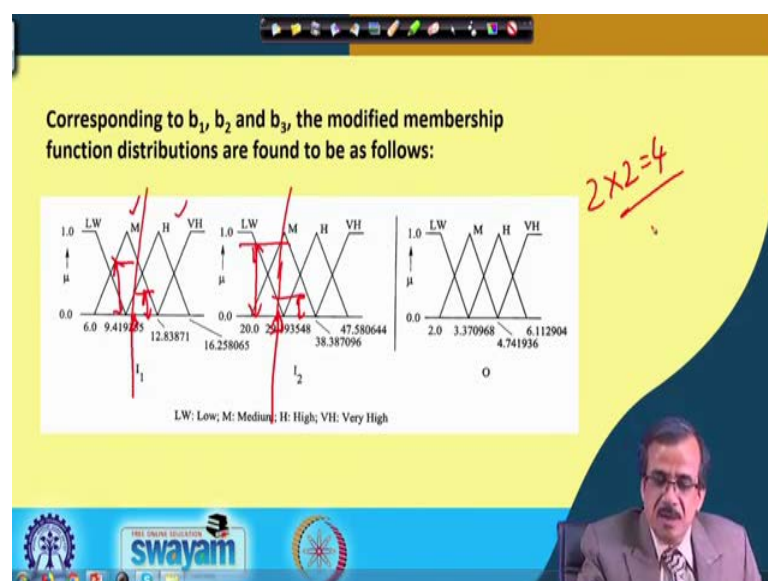
So, this represents actually the first rule. Now if I just write on this particular GA-string here. So, this is 1. So, this particular 1, then there is three 0's 0 0 0 so I am here, next is 1 0 so I am here, next is 1 0 so I am here, next is 1 1 1 so I am here, then 0 0 1. Now, one means your so this particular rule is present, 0 means this rule is absent and so on. So, this is the way actually, we can code in fact, the rule base inside the GA-string.

(Refer Slide Time: 19:09)



Now, once we have got this particular thing, now, actually what we can do is, I can pass a particular training scenario. Now the training scenario is something like this the first training scenario. So, if  $I_1$  is 10,  $I_2$  is 28. So, actually we will have to find out what should be the output of this fuzzy reasoning tool. Now, for  $I_1$  equals to 10 and  $I_2$  equals to 28. So, if you see the modified membership function distribution, if we see the modified membership function distribution, it is like this.

(Refer Slide Time: 19:53)



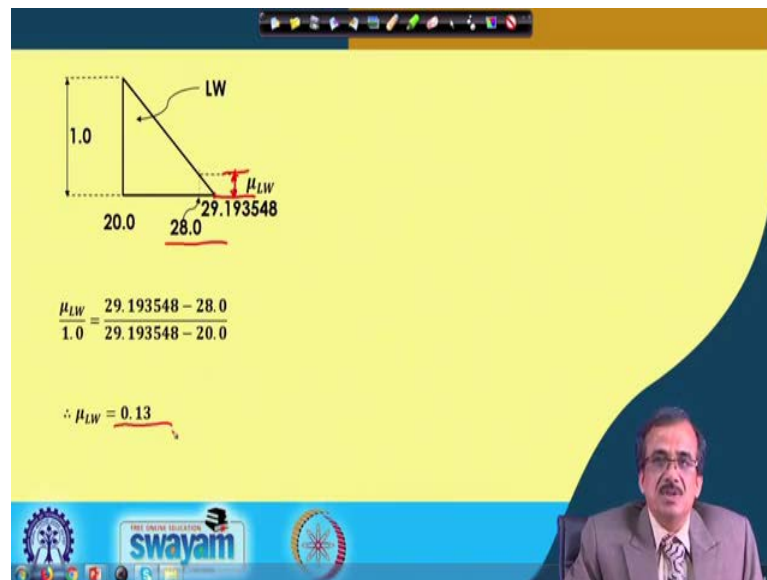
So, I\_1 is 10 means I am here, so if I\_1 is 10 so I am here, so that means, I am here. So, this I\_1 could be either medium with some  $\mu$  value or it could be a high with another  $\mu$  value; now similarly this I\_2 is 28; that means, I am here. So, for this 28, it could be low, with this much of membership function value and it could be medium with this much of membership function value.

So, this particular I\_1 could be medium or high and your this I\_2 could be either this particular your low or medium. So, I have got 2 multiplied by 2, a maximum of 4 fired rules. Now, out of this maximum possibilities, we will have to check, which of the rules are present. So, to check it, in fact, so what we will have to do is, we will have to come here and we will have to find out. So, out of these 4 fired rules, which 1 or which 2 or which 3 or which 4 rules are present here, that we will have to find out.

Now, for this particular problem, if you see. So out of this in fact, only there are 2 fired rules the fired rules are as follows. If I\_1 is medium AND I\_2 is low then output is low, the 2nd fired rule if I\_1 is high AND I\_2 is low then output is actually the medium. Now, let us see, now corresponding these, if I\_1 is your medium, this is the membership function distribution for medium. So, corresponding to this so I am passing I\_1 equals to 10. So, I should be able to find out, what should be the membership function value and this is nothing, but the membership function value.

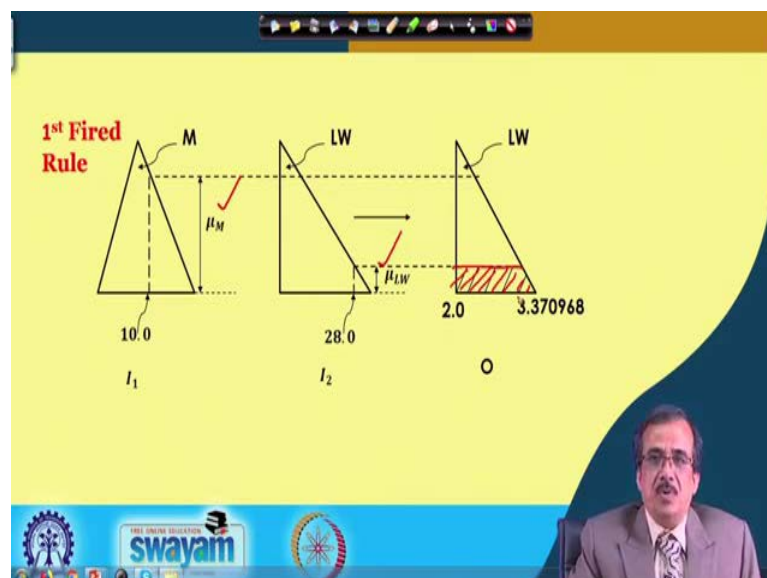
Now this I have already discussed that by using the principle of the similar triangle, very easily you can find out what should be the membership function distribution corresponding to this medium and if you calculate, you will be getting  $\mu_m$  is nothing, but 0.83.

(Refer Slide Time: 22:17)



Now, if you follow the same principle, just to find out what should be the  $\mu$  value corresponding to  $I_2$  equals to your that 28. So, corresponding to 28, I can also find out what is your  $\mu_{low}$  following the principle of the similar triangle and I can calculate the  $\mu_{low}$  is nothing, but 0.13.

(Refer Slide Time: 22:49)

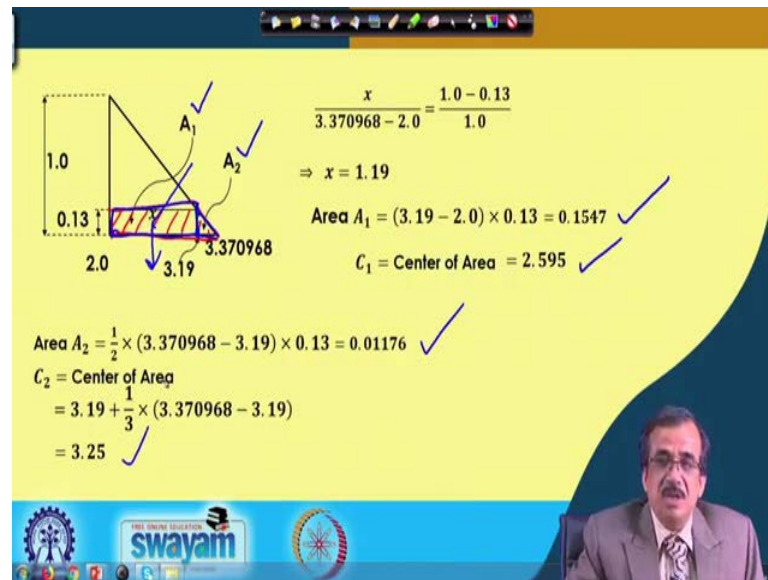


Now, if you see, you can find out the  $\mu$  values. Now, if I concentrate on your the 1st fired rule, that is nothing, but if  $I_1$  is medium AND  $I_2$  is low then output is low, this is a membership function distribution for the low. Now, corresponding to this medium, we



have already determine  $\mu_m$ , corresponding to low we have determined  $\mu_{low}$ . And, we will have to find out the minimum value, as we have already discussed and corresponding to the low value of  $\mu$ , this will be the output, the fuzzified output.

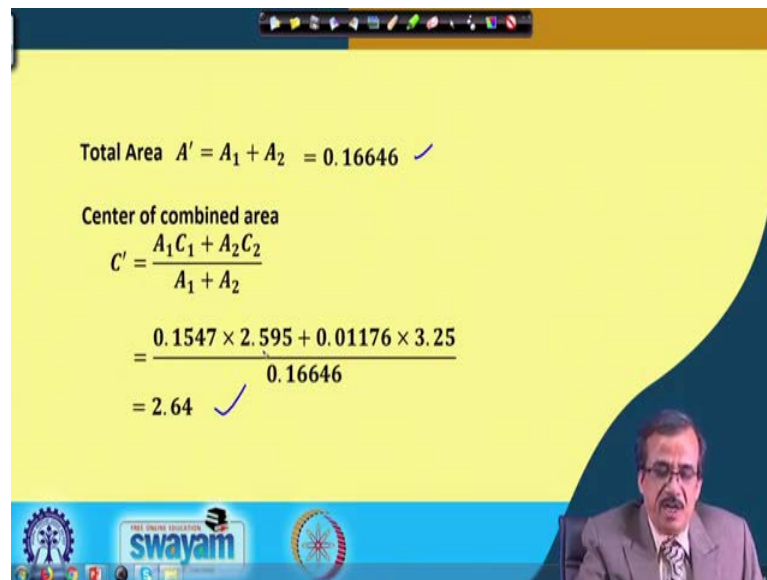
(Refer Slide Time: 23:33)



Now, following the similar procedure, I can also find out, what should be the output for the 2nd fired rule, but before that let me concentrate a little bit more on how to determine the area and center of area, corresponding to this fuzzified output, which we have got corresponding to the 1st fired rule. Now, I will have to find out the area and center of area of this particular the shaded portion. Now, to determine the area and center of area, so what we do is.

So, this is divided into two parts, I have got one rectangle sort of thing and I have got one triangle sort of thing, ok. Now, what you can do is, I can find out the area for this rectangle, that is  $A_1$  and area for this triangle, that is  $A_2$  and I can also find out the center of area. Now, this I have already discussed in much more details. So, I am just going to skip the detailed discussion on this and because I have already discussed. So, this area  $A_1$  can be determined like this, the center of the area the  $C_1$  can be determined like this. Similarly, the area for the triangle that is  $A_2$  can be determined like this and the center of area can be calculated like this.

(Refer Slide Time: 25:07)



Total Area  $A' = A_1 + A_2 = 0.16646$  ✓

Center of combined area

$$C' = \frac{A_1 C_1 + A_2 C_2}{A_1 + A_2}$$

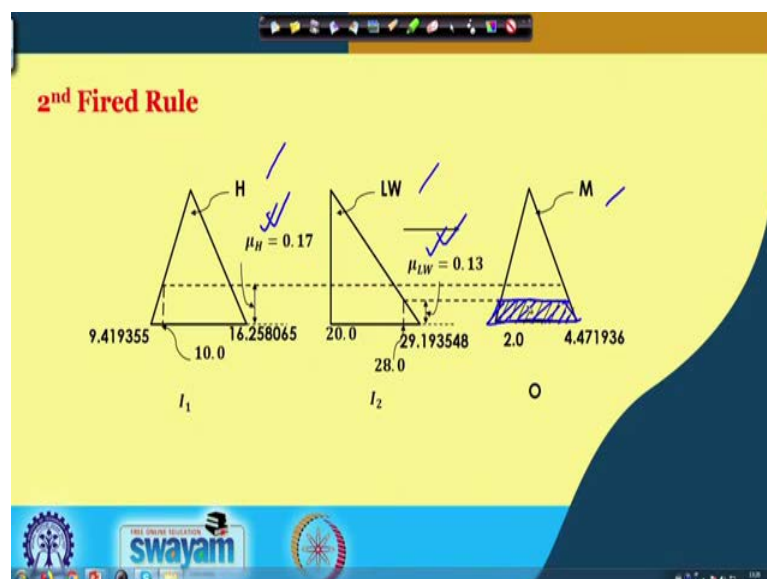
$$= \frac{0.1547 \times 2.595 + 0.01176 \times 3.25}{0.16646}$$

$$= 2.64 \quad \checkmark$$

The slide also features a Swamy logo and a small video inset of a man in a suit.

And, once you have got this area and center of area. So, now, I am in a position to find out corresponding to the 1st fired rule. So, what should be your the center of combined area. So, I can find out corresponding the 1st fired rule like what should be your the area and what should be your the center of area. Now, this area and center of area I can find out. So, this area is nothing, but this and center of area is nothing, but this, corresponding to the 1st fired rule.

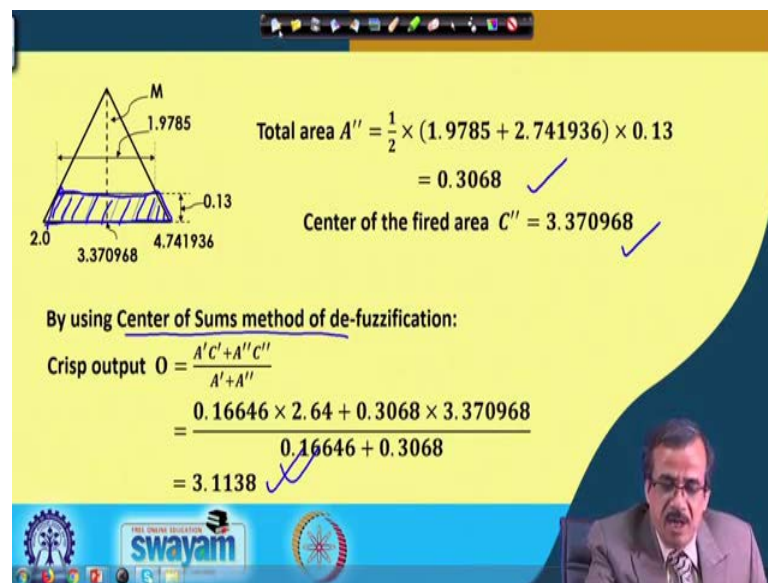
(Refer Slide Time: 25:41)



And, once you have got this, now we can concentrate on the 2nd fired rule. The 2nd fired rule states, if I<sub>1</sub> is high and I<sub>2</sub> is low then output is medium, and I can find out what is  $\mu_H$ , I can also find out what is  $\mu_{low}$  I can compare, so this is the minimum.

So, corresponding to this in fact, I can find out. So, this will be the area, the shaded area and now, I will have to find out the area and center of area of this particular shaded portion.

(Refer Slide Time: 26:23)




Now, this area I can find out, now this is the area of the shaded portion and center of area I can find out. So, if I know the area and center of area for this particular fuzzified output of the 2nd rule and by knowing the same for the first rule; now we are in a position so that we can find out using the center of sums method, what should be the crisp output.

Now, if you follow the principle of center of sums method very easily you can find out that this will be the crisp output and once you have got the crisp output and that is nothing, but is your the calculated output. Now, if I know the calculated output.

(Refer Slide Time: 27:09)

Deviation  $d_1 = |3.5 - 3.1168| = 0.3832$   
corresponding to the first training case.  
Pass all the training cases and determine the deviation values as follows:

Sl. No.	$I_1$	$I_2$	O	Deviation in prediction
1	10.0	28.0	3.5	$d_1$
2	14.0	15.0	4.0	$d_2$
⋮	⋮	⋮	⋮	⋮
T	17.0	31.0	4.6	$d_T$




So, this is the calculated output, we compare with the target output, find out the deviation and the deviation could be either positive or negative, take the mod value and this is actually the deviation. Now, this particular deviation is corresponding to the first training scenario. So, this is the first training scenario, now we will have to follow the same procedure for all the training scenarios and we will be able to find out  $d_2$ ,  $d_3$  up to  $d_T$ .

(Refer Slide Time: 27:51)

We calculate:  $\bar{d} = \frac{\sum_{i=1}^T d_i}{T}$

Sl. No.	GA-string	Fitness
1	10110011011101110001010111001	$f_1$
2	011001011011010001010111010010	$f_2$
⋮	⋮	⋮
N	1010001110101110100100110111011	$f_N$

$\frac{1}{f}$   
Min-f



And, after that actually, what we do is, we try to find out the total deviation and that is divided by the number of training scenarios and that is nothing, but the average deviation. Now this particular average deviation is nothing, but the fitness of this particular the first GA-string.

So, this is the GA-string for which we have got the fitness and this is  $f_1$ , but is your  $d$  bar that is the mean deviation. Now, similarly you can find out the fitness for the second GA-string the fitness for the  $n$ -th GA-string and this is a minimization problem. Now, if use the binary-coded GA and if you want to convert, we can convert the objective function to minimize this particular fitness. So, what we will have to do is, I will have to maximize say  $1$  divided by the fitness. So, I can maximize  $1$  divided by fitness just to minimize your the fitness, that is,  $f$ .

Now, you are going to use the operators like reproduction, crossover and mutation. Now, GA through a large number of iterations will try to find out the optimal knowledge base of the fuzzy logic controller; that means, your optimal database and rule base for the fuzzy logic controller so that this particular fuzzy logic controller can predict the output for a set of inputs as accurately as possible.

Thank you.