Business Analytics for Management Decision
Prof. Rudra P Pradhan
Vinod Gupta School of Management
Indian Institute of Technology, Kharagpur

Lecture – 51
Prescriptive Analytics 3

Hello everybody, this is Rudra Pradhan here. Welcome to BMD lecture series. We will continue today with prescriptive analytics and that too coverage is on interior programming. And in fact, we have already discussed this integer programming problem. It is not something different; it is similar structure of linear programming problem that to like primal and dual, and here the requirement is the values of the decision variable at the optimal stage should be integer type. In the first instance it will get the values of the decision variable that you know that is the integer type, then there is know further requirement or there is no further processing or you know more number of kind of you know iterations through which you can actually address the business problem.

But if the values of the decision variable at the optimal stage is not coming integer type then we look for the a further kind of you know iterations by different kind of you know structuring and restructuring through which you can get the optimal solution, and that is a you know integer type and that too address the business problem as per the particular management requirement.

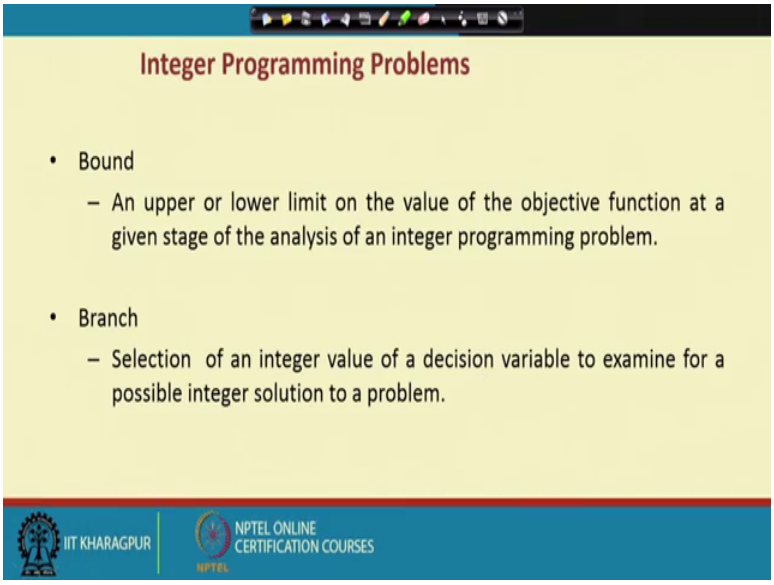
So, what we have already discussed in the last lecture. So, the integer programming can be 3 types. So, it can be pure integer type, it can mixed integer type, or it can be 0 1 integer type. So, pure integer type is a situation where all the values of the decision variable must be integer type. And in the case of you know mixed integer programming. So, some variables, means, some of the values of the decision variable will be integer type and some are not integer type. But in the case of you know 0 0 1 integer programming, all the values of the decision variable either 0 or 1.

So, that means, so, these are all typically you know means typical structure of integer programming, and that too theory integer programming, and then mixed integer programming, and then the kind of you know 0 1 integer programming. So, in this lectures, we like to start with a kind of you know problem, and a you know and then we discuss the kind of you know requirement whether it is a kind of you know mixed integer

type or theory integer type, and then how we can actually come with a kind of you know solution which is exclusively theory integer type a situations. So, as usual the typical structure is the we must have a linear programming problem, that to must have objective function constants and condition. Then the usual procedure we will first start and then look for the optimal solution.

And once we reach the optimal solution, then we like to take a decision whether will stop there or will proceed further till you get another optimal solution as per the particular business requirement.

(Refer Slide Time: 03:30)

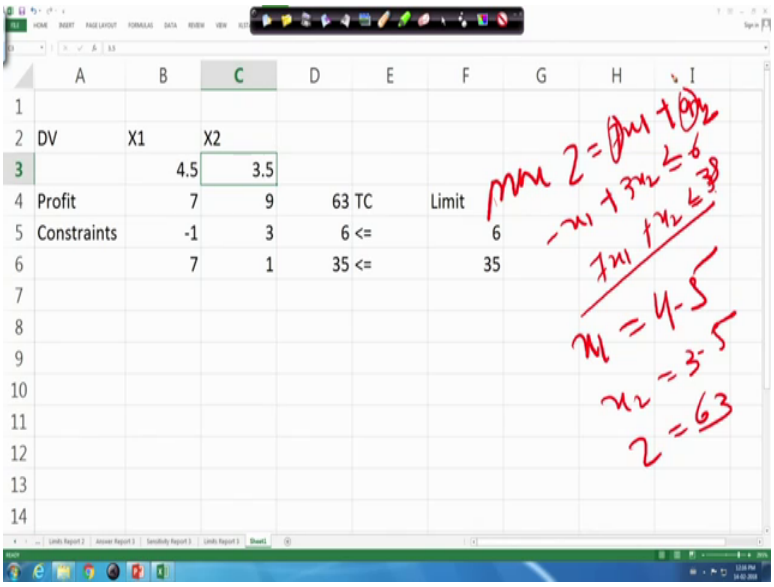


So, accordingly, what we have already discussed earlier. So, if the optimal solution is not integer type, as per the particular you know management requirement. So, then we we like to you know reevaluate the particular process, and the reevaluation of this particular process can be through CPM or through BBM coating plan mechanisms or branch and bound mechanisms. In both the occasions, it is like you know sensitivity analytics structure. So, that means, after reaching the optimal solution which is not integer type, then the final optimal tables can be used and in that particular you know process, we like to add a another constant by using some of the you know values of decision variable, through which again we can actually restart the process, till you get the optimal solutions where the values of the decision variable will be integer type. So, that means, whether you use coating plan methods or BBM branch and bound method.

So, the procedure to get the integer programming solution is just to add a another constants, and then you know then follow the iterative process procedure, against you know as usual check the kind of you know optimality, then you know get the optimal values and again check the integer type whether the values of the decision variable are integer type or not.

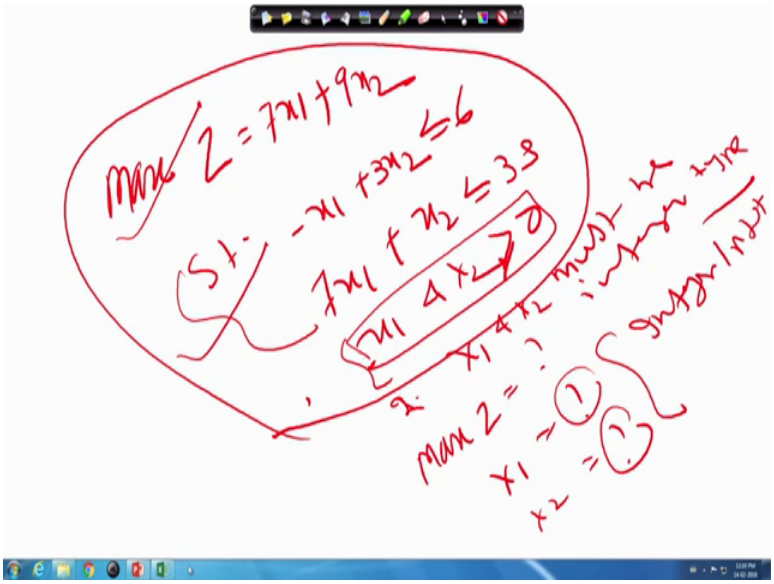
So, that means, it is a extra kind of you know check we have to do here in each and every stage. Then finally, we will have a solution which is the a final which we have a kind of you know where you know all the values of the decision variable will be positive and that too it is exclusively on integer type. Either 0 1 or 0 1 2 something like that only so, it cannot be the a ratio type. So, let us start with you know you know some examples then we can discuss. So, I will take you to the a solver.

(Refer Slide Time: 05:34)



So, let us say this is a problem. And the problem is here like this so, we have actually objective functions. And the objective function is the a so, let me a highlight the particular you know structure again. So, here the idea is the just to know how it is actually happening.

(Refer Slide Time: 05:58)



So, the whole procedure is the we have a problems like this, you know, let us maximize maximize z equal to 7 x 1s, plus 9 x 2 and subject to minus x 1s plus 3 x 2 less than equal to 6 and a 7 x 1s plus x 2 less than equal to 35, and x 1 and x 2 greater than equal to 0. So, this is what the original problem, and here there are 3 parts. The first part is the objective function; the second part is the constraints. And the third part is the conditions here the first-hand condition whatever we have discussed till now is actually both will be positive in nature. Sometimes, we may have the kind of you know solution having unrestricted sign. But here we are putting the restriction where the values of decision variable must be positive.

And against we are adding another kind of you know options condition that is both x 1 and x 2 must be must be integer type ok, must be integer types. So, this is how the a particular you know requirement. So, that means, so, first you know first step of the process to solve the problem, and must have maximum value of you know z subject to x 1 value and x 2 value.

Once we get x 1 value and x 2 value we will check whether it is a integer type or not integer type, right, this is how the kind of you know requirement. So now, having the kind of you know problem let us first we start with the simple structure, then look for the a optimal solution where the values of the decision variable will be a integer types. So, let us go to the a particular you know structure here. So, structure will be here let us we

will go to the solver and in the solver. So, here the problem is a maximize z equal to 7 x 1s plus 9 x 2 and minus x 1s plus 3 x 2 less than equal to 6 and 7 x 1s plus x 2 less than equal to 35 this is the original problem.

And now so, accordingly we have said yes, and the requirement is here that you know we have a dependent variables x 1 and x 2, and the profit function is actually putting 7 and 9. And then here the constraints are you know minus 1 and 3 third means minus x 1s and 3 x 2 that is a less than equal to 6 that is the limit here, and then 7 x 1s plus x 2 that is you know less than equal to 35. So now, as usual the particular structures, we will look for the optimal solutions, and the optimal solutions ok. So, what will we do? So, we will reset the particular you know process.

So, this is how we have actually operated, and go to the solver and then start operating the particular you know process. So, this is already the kind of you know operations. So now, in the solver (Refer Time: 09:05) case these are all having actually entry. So, what will you do? So, we can you know refresh. So, that means, technically we reset all the particular you know requirement. So, after in the case of you know reset. So, this is first you know set the objective. So, that objective is the, this one. So, that is you know default.

And then we will come to the kind of you know objective function structures. So, this is what actually x 1 and x 2, and then against we come to the constraints. So, the first-hand constraint will be. So, that means, technically you have to go on add on add on. So, this is the first constraints and this is the less than already less than type, and the limit is here 6, and then follow you know it can be followed by another constraints and as a result. So, you can go to the second constraints, and this is again less than type and the limit is here 35, and then put . So, the moment you will put.

So, that means, technically so, we have actually here a the kind of you know condition. So, in that means, technically what I have done here. So, I just set the objective, and set the constraints and look for the here we are putting the option of you know simple x. And we in the first hand we are not actually setting the you know integer type. So, obviously, we can know get to know what is the difference between the original solutions, and the kind of you know integer type of you know solution

So now you know in the solvers I do not give any kind of you know restriction to integer. So, in the first hand so, adding the constraint these 2 constraints and their you know limits. So now, the process is you know ready to operate. So now, just you know put click solve the situation then we will have a you know answer here and then put ok. So, that means, technically we have here solutions and in the solution. So, what we have here a x 1 equal to 4.5 and x 2 equal to 3.5.

So, that means, corresponding to these particular you know problem. So now, we have here, we have here now the kind of you know x 1 equal to 4.5, and x 2 equal to 3.5 and then z equal to 63. So, that means, typically it is since our profit target is 7 for x 1 and 9 for x 2. So, obviously, 7 into 4.5 plus 9 into 3.5 will give you the optimum value of z that is 63. And once you get 63, and where the values of decision variable you know are 4.5 for x 1 and 3.5 for x 2.

But the particular problem requirement is that, you know, values of the decision variable will be positive that is the first-hand requirement. So, this is satisfied; however, values of the decision variable again should be integer type. So, in this you know solutions, we have reached the optimality as per the objective function constraints and condition; however, the in the conditions the second restriction is not actually fulfilled here. So, that means, x 1 is not integer type and x 2 is not integer type. So, so, means all the variables are actually non-integer.

So, that that means, technically we have not reached the final optimality depending upon our business requirement or management requirement. Although, we have reached optimal solution, but this solution is not effective for our, you know management requirement. So, as a result what we will do we have to you know reevaluate the particular process. So, that means, say we need some kind of you know clue through which actually we continue the iterative process. So, that is how you know integer programming coming into the picture.

So, against a after getting the optimal results here. Since, these are all not integer type. So now, we can use either coating plan mechanisms or you know branch and bound mechanisms through which actually you can reevaluate the process and continue till you get another optimal solutions, where the values of decision variables you know against you know optimal, and then we will stop at a particular point where you know the values

of decision variable not only optimal not only positive, but also integer types. So, that means, in the coating plan mechanisms, we need to create a new constraint depending upon the final you know optimal table of the original problem.

Similarly, in the case of you know branch and bound mechanisms, we also create you know with a given kind of you know situations we can you know a you know branch the particular program into 2 parts, and one is the less than type of you know constraint and one is the greater than type of you know constraint depending upon a particular you know value of the decision variable which is not integer type.

If a if only a variable is you know non-integer, then that particular variable can be clustered into 2 different branch. But if all the variables are you know non-integer, then a we like to find out which variable should be used for the kind of you know branching. So, will discuss in the later stage in the meantime, awe start with the simple structure of you know coating plan mechanism in the coating plan mechanism the idea is to create a new constraint after reaching the optimals optimality and where the values of the decision variable is not optimal, means, values of the decision variable not optimal means it is not integer type. And where we can actually use a particular mechanism called as you know gomarian constraint.

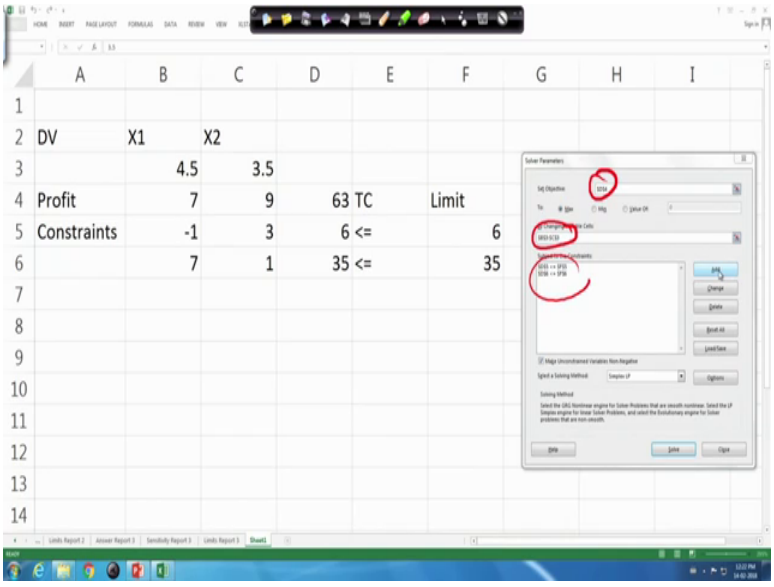
So, that means, there is kind of you know in the coating plan you know methods. So, it is a mathematician who developed this particular you know technique and according to his guideline. So, we like to create a constraints new constraint that is through a gomarian constraint, and then a this is what the sensitive analytics is all about; so, we are now adding a new constraint. Of course, the criteria to you know crate a new constraint on the optimal solution is the kind of you know trick business.

And there is a procedure through which you can actually derive a new constraint, and again a reevaluate the process or do the iteration again look for the optimal solution and then check the optimality where the values of the decision variable not only positive, but also integer type. So now, since it is actually here you know we have reached the optimal solution and which is not coming integer type and ah, you know. We are not discussing details about the particular mechanisms, but the procedure is the just you know after reaching the optimal solution, then you know create a new constraint which can help you to go for the processing of that original optimal solutions, and take you to another you

know kind of you know corner point where will reach again optimality and the values of decision variable will be you know optimum and that too it is positive and integer type.

So, in the solver package, there is no need to no need to go to the mechanism. So, what will you do here? So, we go to the actually solver package again, and then what will what will you do here.

(Refer Slide Time: 16:43)

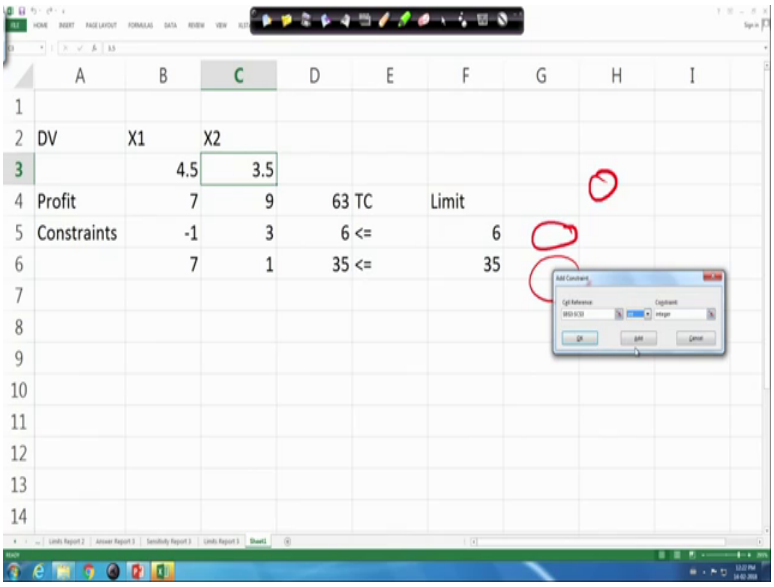


So, till now we have not given any kind of you know integer restriction. Of course, manually you know go step by step process the initial basic feasible solution then you know first improved basic feasible solution second improved basic feasible solution and then. So, on and finally, will go the final optimal basic feasible solution. And infact in the integer programming the route will start from the final basic feasible solutions. So, where you know we have reached the tech means technically we have reached the optimality, where the values of the decision variable will be positive, but not integer types. Then a through cutting plan mechanism and branch and bound mechanism, we do further processing by adding a new constraint and then look for the optimality, where the values of the decision variable will be again integer type.

So, that means, technically there is the, another restriction or another constraint we have to add here ah. So, that you know particular optimal solution will you know will change, and then we have a new optimal solution where the values of decision variable will again you know integer type. For that what will you do? Infact we have already set the

objectives here, and we have we have here constraint; that means, technically this is what the objective we have set and this is what the kind if you know ah, you know, the kind of you know linear combination of you know 2 products objective functions.

(Refer Slide Time: 18:42)



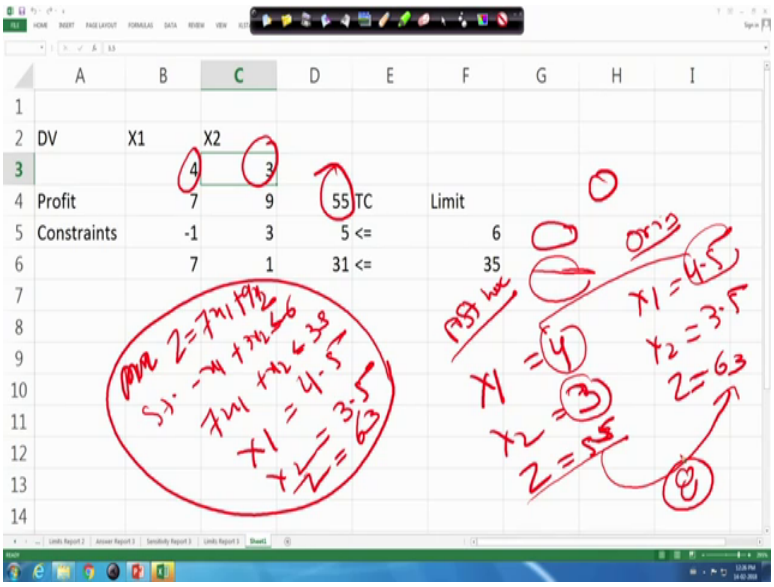
Then these are all 2 constraints and; obviously, by default we are taking care of the non-negativity restriction. Now what is happening, you know, for the integer type of you know requirement so, that means, technically we need to add a a you know another restriction. So, as a result so, what will you do here? So, we will we like to go for you know a add on. So, put you know a add on here uh, and then will find here the kind of you know requirement that is the condition. So, that means, the condition both x for x 1 and x 2 means we are going for you know pure integer type case.

So, in that case so, here the option is actually we have here less than type equal to greater than then integer type. So, you just choose the option of you know integer type, and then put ok. So, that means, you will you see here so, the new restriction is imposed here. So, that means, technically this new restriction is imposed here, and after you know imposing the new restriction. So now, you again allow the solver to you know solve the problem, and then if you put we know in the solver this is actually a ask the subtract solve the problem.

So now, we have a another kind of you know solution. So, in the previous solutions we have a different kind of you know output in the previous solution; that means,

corresponding to our original problem. So, our original problem is here, our original problem is the maximize a maximize a z equal to z equal to 7 x 1s plus 9 x 2 and subject to minus x 1s plus 3 x 2 less than equal to 6 and 7 x 1 plus x 2 less than equal to 35.

(Refer Slide Time: 19:46)



And then we have solution x 1 equal to 4.5 and x 2 equal to 3.5. So now, so, this is the first hand you know solutions without putting the integer restriction. Now after putting the integer restrictions, through you know cutting plan mechanism, then we have the solution here. And in fact, z is here a z is here coming 63, right, then in the case of you know you know cutting plan mechanism. So, we have another solution.

So, where the values of you know a x 1 equal to 4 and x 2 equal to 3 and z equal to 55. So, this is what the, you know z value. So, 55 so, that means so, this is x 1 value this is x 2 value. So, that means new x 1 value new x 2 value and new objective function value. So, that is coming actually 55 so, that means, corresponding to original problem. This is a post huc output , this is post huc output where we are putting the integer restrictions, and the original output so, we have x 1 equal to 4.5, and x 2 equal to 3.5 and z equal to 63 .

Now, if you compare the post huc analysis where, you know, we have put the integer restriction. So, in that case, in that case the 4.5 reduce to 4. So, that means, we remove the 0.5, and in the case of you know round off structure. So now, ultimately so, x 1 equal to 4 and x 2 equal to 3 in the new solution means new feasible solution and that is

optimal again. And where x 1 is positive x 2 is positive and x 1 is also integer type x 2 is also integer type; that means, it is a now pure integer type solution because we ask the solver to give the solution which is actually pure integer type. So, corresponding to x 1 equal to 4 x 2 equal to 3, our objective function is coming a 55, now if you compare actually objective function 55 and the original objective function original optimal output where z equal to 63. So, that means, technically a there is difference of you know 8. So, original objective function which gives you know 63 you know kind of you know profit level, and here the profit level is come down to 55.

And because of you know the integer restriction, but ultimately this is what the our you know business requirement, you cannot just you know have the optimal solution, and then you may not in a position to you know address the you know business problem as per the management requirement. Since, these are all production kind of you know requirement. So, this should be in numbers only.

So, no ratio so, ultimately the solution requirement for this business problem is the integer type solution, and a in the first end you know with respect to the original problems, we have optimal solutions and that too the values of decision variable where both the variables are not integer type as a results you know we create you know a gomarian constraint and through which actually reevaluate the process and then we reach the optimality again. And in the new optimality a we have the a values of decision variable and that too x 1 is the 4 here and x 2 is 3 here both are positive and both are integer types.

But ultimately, you know, the value of objective function is actually somehow reduced corresponding to original optimal solution. And obviously, when you put one after another constraints and condition, there is the high chance that you know the value of objective functions you know will be reduced or you know; that means, that means technically you need to compromise, if you look something else and then; obviously, other side we need to sacrifice something else.

That is what the concept of you know the concept called as you know opportunity cost in the kind of you know business environment. So, looking for a particular requirement so, somewhere we have to compromise otherwise, we may not in a position to get the optimality or the optimum values of the decision variable where we can address the

business problem more effectively more efficiently more accurately, and that too as per the particular you know requirement.

So, so, this is how the solver can help you to get the kind of you know integer solution. In fact, manually you know it will take you know long times to reach the optimal solution, of course, since it is a kind of you know sensitivity structure after the optimal solution original optimal solution. So, there is high chance that you know, it will be you know in the immediate next step you may get the optimal solution, then the issue is whether the particular you know optimal solution is again integer type or not.

So, that means, original you know optimal solution will give you the indication, whether we like to stop or we like to continue. So, if the values of the decision variable will be integer type, then we can stop here then address the problem, and then take the management decision. If not then against you know derive again new constraint through the gomarian mechanism, then continue the process, then you will go for another you know iterative process. And again, by default you will reach the optimality, and again check whether the particular you know solution is optimal, and whether the values of decision variable again integer type if it is again you know optimal and values of the integer values of the decision variable are integer type then that is the stop point.

And then address the business problem effectively and then take the management decision, if not then against you know derive another new gomarian constraint corresponding to the second optimal solution, right? And then you continue again till you get the optimum result that is the optimality, and again check the values of the decision variable whether you know they are positive and integer type a.

So, once you know you know have the optimal solution where the values of the variables are both positive and integer type, then you can stop and address the problem business problem a. Again, if the values of the decision variable is not actually integer type, again you have to you know create a new constraints and then again continue and uh; obviously, we will reach a further you know optimal solution. And by the previous you know steps you can also check the optimality, you know, negativity and the integer type, then you know if you can reach out you know the kind of you know final optimal solutions where all the conditions are satisfied then you can stop and address the problem.

If not, then again you can continue with you know again the case of you know new constraint. So, that means, this is a typical you know continuous process you know till you get the results as per the particular you know business requirement and the kind of you know management requirement . Technically, it is a very interesting kind of you know component, because some of the business is a very ah, you know, you know key kind of you know requirement, and specific means technically some business has a specific requirement.

If you could not fulfill this specific requirement, then you cannot actually come with a kind of you know strategy or we will not come in a kind of you know position to take some kind of you know decision. So, that is why integer programming will help you as per the business requirement the kind of you know management requirement. So, that you know we can actually address the problem more effectively. So, that means, this is a extra flexibility or extra advantage in the prescriptive analytics structure, which is highly required as per the various business dynamics and the various business requirement.

Like we have discussed you know in the predictive analytics, lots of you know (Refer Time: 28:37) checks lots of flexibility by creating different kind of you know options. So, here in the case of you know prescriptive analytics, we have also different kind of you know options through which actually you can you know analyze the particular you know problems, and look for the optimal solutions, and the values of decision variables and then address the problem without any kind of you know further restriction or the kind of you know conditions.

And against it will give you the flexibility to have a another kind of you know solution depending upon the kind of you know more conditions more kind of you know restrictions and as per the particular you know business requirement; that means, it is a in a kind of you know technique which can have a kind of you know solid structures , you know, typically means what I can say that you know it is a it is a kind of you know specific requirement all together, in a specific business requirement. depending upon the specific business requirement, this particular technique is usefuls ok. So, I am very sure you know the integer programming is actually a problem specific business specific or you know management specific.

So, if there no such you know requirement in the business or kind of you know management. So, then no need to go for integer value, because it is a it will add you know extra kind of you know steps or you know extra iterative process where we can have a some kind of you know compromise with respect to values of the decision variable and values of the objective functions in 99 percent instances. So, you know any restriction ah, you know, further restriction like you know integer type solutions or something like that.

So, every time there is compromise of you know value of the objective functions. So, that is why when there is no specific requirement of a particular business or a particular management. So, it is not actually look for the a solution where it is integer type, but if a problem is actually it needs an kind of integer solution, then you can apply otherwise there is you know high requirement. Because the original problem can have the optimal solution without integer kind of you know restriction, and that will give you high object you know you know objective function value and now putting any restrictions like you know integer type and that too may be mixed integers and the kind of you know pure integers.

That means technically if you put pure integer kind of you know structure, again there is more compromise on you know value of the objective function. So, that is why so, if the if there is no typical you know business requirement about the integer type solution corresponding to all the values of the decision variable.

So, no point to go to the integer kind of integer programming a that too any mechanism through which actually you look for the another optimal solution and address the business problem as per the particular you know requirement. So, that means, in some we like to you know submit that you know. So, this is a business specific you know kind of you know requirement, and this technique is very useful the these occasions where you know the values of the decision variable will be only integer type.

So now, with this you know I like to take you to the particular, you know, structures so, the here, the particular requirement is you know the cutting plane mechanism. So, that means, the problem which we have solved is a kind of you know cutting plane mechanisms, where every stage you know every stage we will have a new constraints that is the, that is derived through a gomarian constraint.

And then look for optimal solutions, and again] every stage we can we can the reach the optimality and then check the values of the decision variable integer type. If you if you if you reach the optimal solution, and then the values of the decision variable integer type, then you can stop and conclude and then take the, you know business decision accordingly.

And if not, you will continue you know in the subsequent stage. I am very sure it will take you to a particular, you know, routes; there is high chance that, you know, because of this negative you know means integer restrictions the optimal solution also may not reach that we can actually give little bit indication means more clarity or more clear cut you know view in the case of you know branch and bound because in the branch and bound compared to cutting plane mechanism a it will.

It will give you 2 different, you know, 2 different constraints all together corresponding to you know target of a particular you know decision variable which is not integer type. So, in the cutting plane mechanisms so, a every next step, you will have a particular new constraint and then look for the optimal solution.

But in the case of you know branch, and bound for a particular decision variables which is not integer type it will add you 2 different new constraints. So, that means, technically you have 2 sensitive a sensitivity case. So, then we like to check you know both the case both the situations, and then see which one is more practical and more valuable to the particular you know business. And where, obviously, you know if there are 2 additional new constraints.

So, we have a 2 different you know problems a corresponding to the original problem. So, that means, a corresponding to original problems. So, we will add one constraint here and another constraint here. So, that means, initially if you have 2 constraints so now, there is a 3 constraints, and the third constraint you know in both the sides will be you know little bit different.

That is different means with respect to sign only, and that to the kind of you know coefficients of the left-hand side of the constraints. And then against we look for the optimal solution the first branching and look for the solution for the second branching. Then we check the optimality again where the values of the decision variable will be optimal, positive and then integer type against you know in the second case we look for

the optimality, check the values of the decision variable which are which must be positive again which must be integer type.

In there are you know different situation all together. So, in the first situation that is the most effective. So, here the optimal we have reached the optimal solution in the branch you know first branching, and where the values of the decision variable will be positive and integer type, and then you report the objective function value, and against in the second case we have reached the optimality and the values of the decision variable again positive and integer type and then report the objective function value.

So now, since both are you know uh kind of you know different solutions, and that too same business problem with a different kind of you know setups. And here ultimately between the 2 the particular you know solution can be a you know finally, choosens depending upon the you know highest value of the objective functions. Or sometimes you know, there is the you know some kind of you know multi objective kind of you know concepts through which you know sustainability, feasibility all this things it will take into considerations, then you know it is you to decide which one should be the final choice for addressing the business problem and for the particular you know management requirement.

So, technically what I like to say that you know so, we have a different kind of you know flexibility. So, the original problem, which is optimal, the values of the decision variable you know positive can be can be used, but ultimately depending upon the business requirement may not you know directly use because of you know the values of decision variable are not integer type. So now, corresponding to the original solution we have now 2 different ulternative solutions; that is how we have a multiple solutions now.

So, in the first ends we have again optimal solutions and then the values of decision variable integer type and the values of the variable integer type, and also value of the objective (Refer Time: 36:47) then a in this kind of you know requirement since our requirement is the integer type for this particular you know business requirement. So, ultimately the first or you know first and original solution will be discarded, then finally, the between the 2 branching problems. So, the choice will be in between the 2.

So, then which one is the highest value depending upon the particular business requirement, you choose that combination. And there are several or other alternatives,

several other alternative means in the first end branching's you know let us say we have reached the optimality and all the values of decision variable are integer type, but in the second case, you have reached the optimality, but all the values are not integer type. So, that means so, in the first branching, you can stop and conclude, but in the second branching you can just you cannot just you know stop and conclude. We will continue with different branching's.

And against this will give you 2 different you know branching, and against you look for optimal solution for both the branching. And then you know check which one is the optimal and then final comparison will be the final optimal solutions which you know satisfy non negativity then integer type; that means, the typical integer programming, it is actually solid structure of you know sensitivity analysis, where you know it will by default give you lots of you know various alternatives corresponding to the original problem corresponding to the business requirement corresponding to the management requirement.

Now, ultimately a you know you just follow up the you know steps and then find out the various alternatives, and then with respect to the business requirement and the kind of you know problem requirement you pick up which one is the good and which one is the best through which you can you know apply here you know management strategy and then take a management decision. you know, with this uh means we can actually a you know understand the kind of you know concept called as you know integer programming, that is the cutting plane mechanism and branch and bound mechanism. In fact, the problem which we have solved through solver you know package is through is a kind of you know cutting plane mechanism.

And in fact, it will give you more different kind of structure in the case of you know branch and bound mechanism, where we have actually more number of flexibility and since a there are more number of flexibility; obviously, it is good for business good for management, because it will give you various, you know, alternatives; then you have the option to choose which one is the best as per your, you know, particular you know business requirement. And we will discuss all this details in the next class. And by the way we will stop here.

Thank you very much, have a nice day.