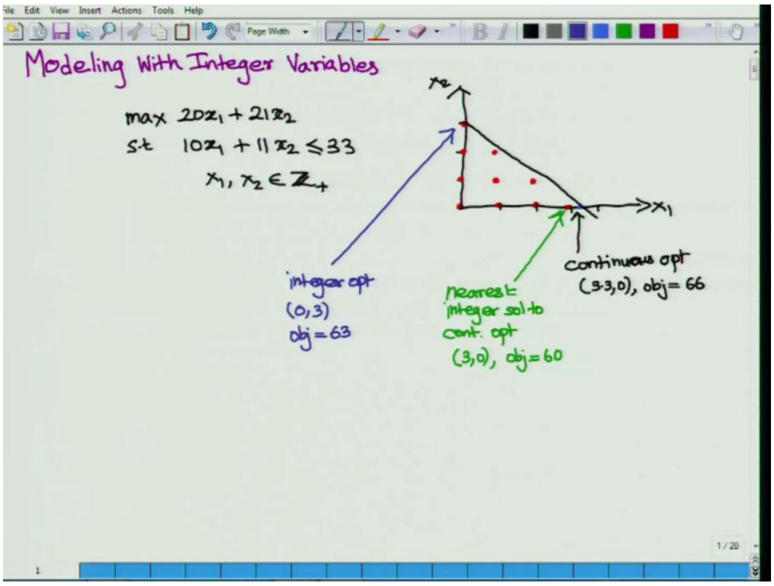
Foundation of Optimization
Prof. Dr. Vishnu Narayana
Department of Industrial Engineering and Operational Research
Indian Institute of Technology, Bombay

Lecture - 37
Special Lecture

Welcome to this Special Lecture on optimization, this is a part of the course, which I am giving on the Foundations Optimization. And these special lectures would come at the very last and it would be an addendum to what is not been told there, but you if you have learnt what has been there, you can understand what we will go on here. So, I would like to welcome professor Vishnu Narayana of IIT Bombay, industrial engineering operation and research, who is an expert in convex and discrete optimization. So, who to give this lecture on, how to model optimization problem with integer variables, thank you very much. Welcome everyone. So, as professor Joydeep Dutta said, today I will be talking about formulating certain problems as integer programming problems.

(Refer Slide Time: 01:07)



So, first of all let us see, so first of all let us see that integer programming is a different problem from linear programming, there are there are certain issues that are not resolved, that that are present in integer programming problem. So, let us look at a really simple two variable example, so what I want to do is to maximize 20 x 1 plus 21 x 2, subject to one constraint, which says that 10 x 1 plus 11 x 2 is less than or equal to 33, I also have x 1 and x 2 are non negative integers.

So, here is a really simple problem, 2 variables just one constraint with both variables, supposed to be non negative integers. So, if we at the feasible region of this integer programming problem, so we have x 1, x 2, and the feasible region or the constraint is going to look something like this. And the feasible points are all the integer points with inside this triangle, which I am going to mark in red.

So, these are the feasible integer points and the one at the corner, now, let us try solving so let us first of all let us ignore the integrality constraints and try solving this as a linear programming problem. Now, it is a very simple linear programming problem by whatever your favorite method is, you can see that the continuous optimal solution or the linear optimal solution is, this particular corner point.

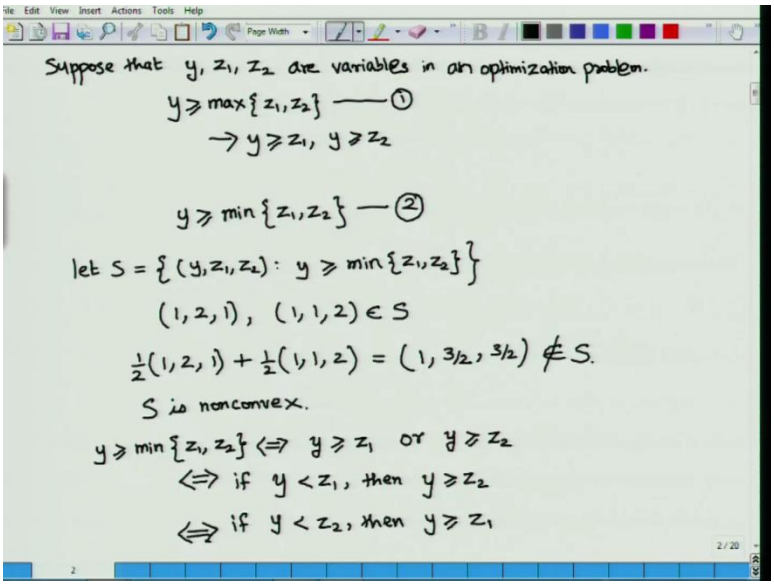
So, we have a corner point, here this is the continuous optimal solution, so the continuous optimal solution, which is the point 3.3 comma 0, with objective value 3.3 multiplied by 20 is going to be 66. Now, if we look at the nearest integer point in this continuous optimal solution, then it is this point 3 comma 0. So, this one is the nearest integer solution to the continuous optimum, whose which is 3 0 and this objective value, as you can compute is 60.

However, if you look at if you look at all integer points, you can see that the integer optimum does not lie any wherever near the continuous optimum solution, the integer optimum is this point 0 comma 3. So, this is the integer optimum 0 comma 3, with objective value being equal to 21 multiplied by 3, which is 63. So, now, one can also generalize this type of an example to see that, the integer optimal solution may be really far away from the continuous optimal solution.

So, that is why, you know integer programming requires separate treatment than just a linear programming problems in general. First, but today I am not going to talk mainly about you know the structure of these problems, I am going to talk about posing different problems as integer programming problems. So, here we saw an example of an optimization problem, where the variables are inherently integer however, there are lot of examples from real life situations, where you are not given any integer variables.

But in order to model problems effectively, you indeed have to use integer variables so let us look at some examples of those kind today.

(Refer Slide Time: 07:21)



So, first of all so suppose you have the following situations, suppose that y z 1 and z 2 are variables in an optimization problem. So, suppose y z 1 and z 2 are variables in a particular optimization problem and they are related by the following, y is greater than or equal to the maximum value of z 1 or z 2. If this is the case, then we can write or if you want to represent this constraint, an easy way of representing this constraint is the following.

I can simply write y is greater than or equal to z 1 and y is greater than or equal to z 2 so these are two different constraints. And this make sure that y is always greater than or equal to maximum of z 1 and z 2, that is not a issue at all. However, on the other hand suppose, so if we call this as, let us say one the constraint 1 can be modeled easily, fell easily. But, suppose if we have a different constraint, suppose what we want is, y is greater than or equal to the minimum value of the minimum of z 1 and z 2.

Now, here there is an issue so here first let me try to demonstrate, that this cannot be written as easily as writing these two constraints. So, the reason, suppose this is called 2 and let us look at the set s, let s be the set of all y z 1 z 2 such that, y is greater than or equal to the minimum of z 1 and z 2 right. So, what I am looking at is, I am looking at, all possible values of y z 1 and z 2, which satisfy this constraint 2. Now, let I am going to show you that, this set s is not convex.

So, if s is not convex, clearly it cannot be represented using a linear any qualities and linear constraints. So, so that is that is why, I can show you that we require something additional, in order to, write this constraint 2 effective. So, let us consider these 2 the following points, so I can look at the following points 1 2 1 and 1 1 2. So, here the first coordinate represents y z 1 and z 2 so here in the first point, the smallest of z 1 and z 2 is 1 and y is greater than or equal to 1.

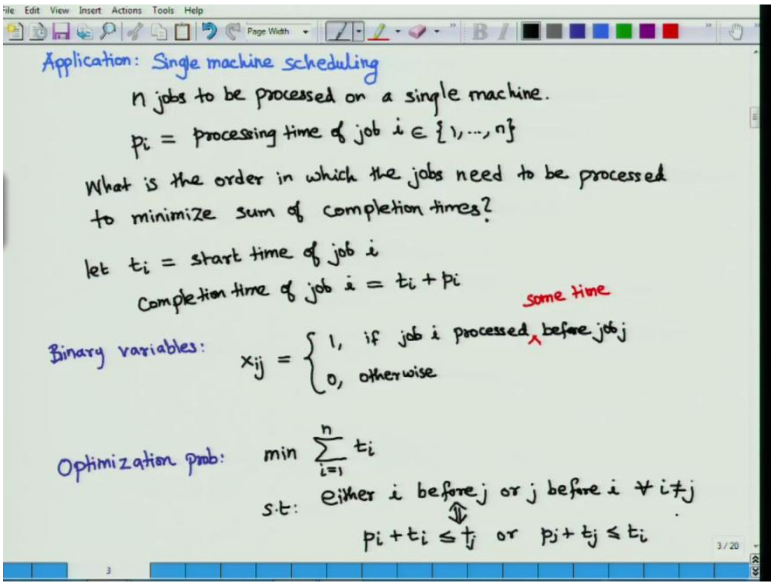
So, that is fine and similarly, I i have just permuted or exchanged z 1 and z 2 so that is also fine. Now, let us take a convex combination of this so let us now, these two points are clearly in s. Now, let us take the midpoint of the line segment joining these two points so half 1 2 1 plus a half multiplied by 1 1 2. Now, this is going to be the point 1 3 halves and 3 halves so clearly here in this case, y is smaller than both of z 1 and z 2. So, clearly this is not in s so s is non convex and therefore, we cannot just model this using polyhedral constraint.

So, we will see that we will require we will essentially see that we will require a binary variables, in order to, model this constraint effectively, the exact model I am not I am not saying right now. Because, we will we will see this in a much larger context. So, I hope I would have convinced you that you know there were a certain situations, where to represent some non convex feasible regions, we do require binary variables.

Let us see how we can do this so what we want is, y greater than or equal to the minimum value of z 1 and z 2, which I can write as y greater than or equal to z 1 or y greater than or equal to z 2. So, essentially what I want is, the disjunction of these logical of these two inequalities, I want either this inequality to be satisfied or this inequality to be satisfied. So, that is the whole idea and this I can again rewrite as, if y 1 is less than z 1, then y 2 is greater than or equal to z 2.

So, essentially you can either use write it as disjunction, or an if then else kind of a statement, (()) disjunction means the or the union essentially. So, let us look at an application, where this type of statements is logical, r type of statements appear naturally, and the application that we are going to talk sorry sorry that is y. So, either y, if y is less than z 1, then y must be less greater than or equal to z 2 or I can say, if y is less than z 2, then solve all the same, so let us look at the concrete application of, where this kind of constraints apply.

(Refer Slide Time: 14:54)



So, this application comes from scheduling single machine scheduling. So, the problem is as follows, now, what do you have some number of jobs, let say that there are n jobs to be processed on a single machine right. And the we have the following data, the data is the processing time so we know that, p i is the processing time of job I. So, we know, what what what is the time taken to execute this job now, what our objective is the following now.

So, we have these jobs, what we scheduling on a single machine means, to decide the sequence, in which the jobs are going to be processed. I mean, everything else is sort of fixed so what we want is, the question that we want to answer is the following. What is the order, in which the jobs need to be processed to minimize sum of completion times so this is a question. Now, this idea of, sum of completion times is a fairly standard objective function in in in scheduling so I am not going to going to more details on, why this is important but suppose I have this objective here.

So, now, clearly suppose if I start processing so let us say that, let t i be the start time of job I. So, we start processing job I at time t i, then the completion time is nothing but of job I is nothing but t i plus p i. And what we want to minimize is, the sum of all t i’s t i plus p i’s, which essentially means that, we want to minimize sum of all t i’s because p i’s are constants. So, how would you do that now, here is where we have some sort of a necessity to invent binary variables, in order to, formulate this problem as a mathematical optimization model.

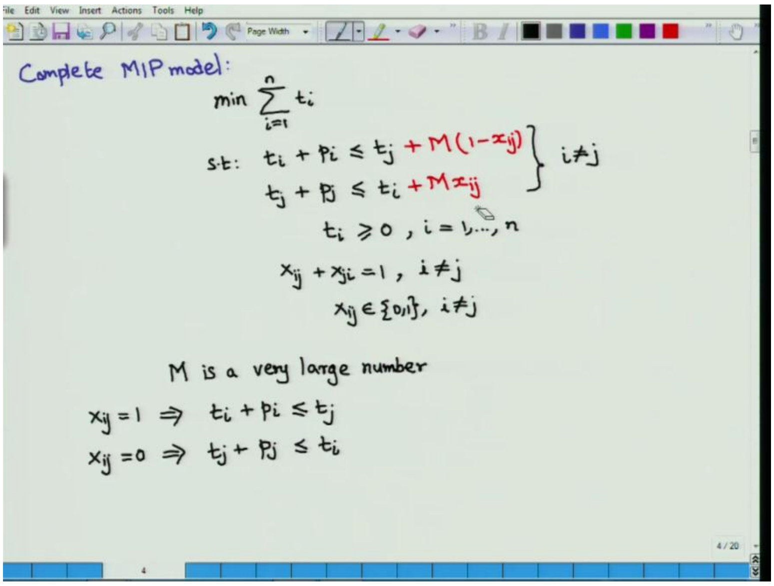
So, we are going to use the following binary variables, so let us use the following binary variables. So, I am going to create a binary variable x i j, which is going to mean the following so x i j is going to be 1, if job I is processed before job j, and 0 otherwise right. And just for emphasis, I am going to say that it is not necessary that job I is process immediately before job j, it is processed can be processed any time before job j.

Now, I i hope you are convinced that if I know these x i j values, I know the order in which, these jobs are processed. So, because I know if you give me job i and j, I know which one is processed first. So, so this gives me the complete order, in which the jobs are being processed so our optimization problem then becomes the following, then becomes the following.

So, what we want is, to minimize summation, I going from 1 to n t I, the remaining is a constant subject to now, what are the constraints, the constraints say that the constraints say that, I mean must have some some idea of the sequence, in which jobs are being processed. Now, if you think of it, I have job i, I have job j, only two things can happen, either job i is done before job j or job j is done before job i, that is the only. So, for every pair, I can I can say that and those are my constraints so my constraints are either i before j or j before i.

And this is true for all pairs i not equal to j and I can rewrite this essentially this is essentially same as same that so when is job I being when is job i process before job j that means, that job j can start only when job i is complete. So, this means that either p i plus t i is less than or equal to t j, or p j plus t j is less than or equal to t i. So, this is the this is the constraint that in need to model now, this is essentially same as s less than or equal to s, j have been completed before i, again for all pairs of jobs i and j. So, now once we know this, we can essentially write what is the complete mixed integer programming formulation.

(Refer Slide Time: 23:36)



So, this is the complete optimization problem looks like the following, so we have or objective that we had it in the last slide itself on minimize summation of t i's subject to s. So, the integer variables are going to tell you, which jobs are done before so it is it is going to give you the sequence essentially. So, t i plus p i less than or equal to t j, t j plus p j less than or equal to t i's, so I know that this is incomplete but we will come back and complete it later.

I just want to fill in the the standard constraints so also need to say that t i's are greater than or equal to 0, for all I going from 1 to n, then I also have the relation between x i, x i j's plus plus x j i is equal to 1. I need to say that, if i is processed before j, then j is processed after i so this is true for all, i not equal to j. And finally, x i j is r binary variables for all, i not equal to 0 so this is essentially, the trivial parts are filled in. But, now what we what we want is, if you look at these two constraints, we want to say that either this is true or this is true.

So, this is a very standard trick in mixed integer programming and here is what I am going to do that. So, what I am going to do is, I am going to add something, I want to say this is, I am going to add some 1 minus x i j. And here, I am going to add, plus m multiplied by x i j and again this holds for all, i not equal to j yes, that is the whole thing. Now, here m is a very large number now, let us examine what happens, suppose t i plus p i is greater than t j so if t i plus p i is greater than t j, the only way this constraint becomes feasible is, when x i j equal to 0.

If x i j equal to one, the this term vanishes so t i plus p i cannot be greater than t j so if t i plus p i is greater than t j, then x i j is 1 sorry x i j is 0. If x i j is 0 then t j plus p j must be less than or equal to t i so that must hold and similarly

Student: (( ))

x i j is 0 yes. So, the

Student: (( ))

Yes and that that is given by this constraint

Student: (( ))

If x i j is 0 then t j plus p j must be less than or equal to t i.

Student: (( ))

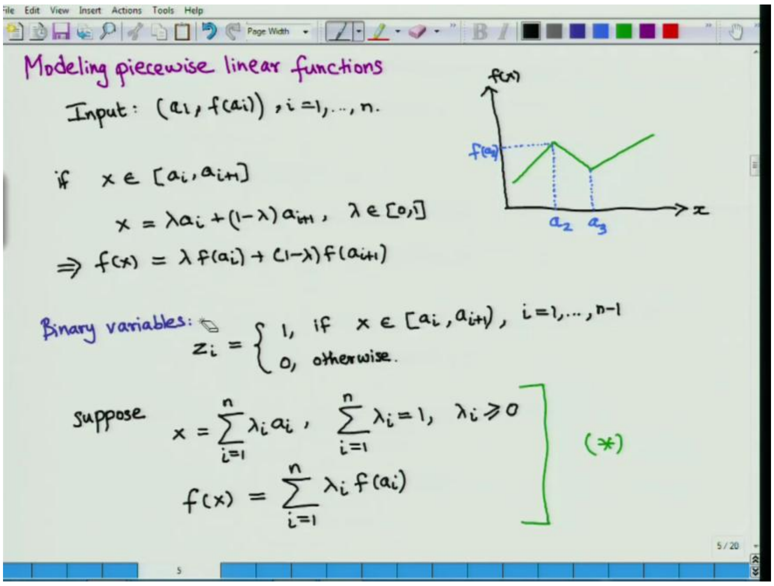
So, if x i j is one then t i plus p i is less than or equal to t j so so that is the the idea of of this population. So, this the using these binary variables, we can clearly get this distinction and let me say that, here m is a very large number and let me also mention this explicitly. So, if x i j is 1, then this implies that t i plus p i is less than or equal to t j and if x i j is 1, this this second constraint says t j plus p j less than or equal to t i plus some very larger number, which does not really say anything because because that is the that is the very weak bound.

And similarly, x i j is 0 will imply t j plus p j less than or equal to t i so this is exactly what we want. Now, before going to the next sort type of model that I want to talk about, let me just say that, although this formulation is correct, this is a correct formulation, this model situation correctly but it is not at all efficient. In fact I want to ask you what will happen if I just do not have that m there just add (( )).

So, then how can you, how do you make sure that so it will depend on what the values of p i's are, if the processing times are very large. So, in fact you can take m to be the sum of all processing type, that is the least that will make sure that you know this is going to work. Let me let me tell you that, this is not an efficient formulation at all, in fact this problem is an easy problem. So, if I want to solve this problem, I would just order the jobs by the shortest processing times and then go on doing that in.

And it can be proved that you know there is a very nice sub modular a poly meteoroid, which represents this situation, using sub modular functions. But, in order to, do in order to prove that it take some time so I am not I am not going to the details. But, there are other efficient models, so just because a model is correct does not mean that, it is a good model so that is the take home message from this part here.

(Refer Slide Time: 30:51)



Now, let us go to a very interesting type of models, what we want is, to model piecewise linear functions, what do I mean by that. So, suppose I have, so I have these are functions of one variable so x is fundamental object, f of x and suppose, we have some kind of piecewise linear function. Set non convex non arbitrary piecewise linear function so it happens that, you know in several practical applications you would have these piecewise linear functions.

So, for example, involving fixed causes things like that, where you need to model these and it is essentially important and we can and it is easy to see that without binary variables. So, we obviously, this cannot be written as a, if you want to somehow model the piecewise linear function, you cannot do it easily unless, it is convex or concave.

Otherwise, if it is arbitrary, it is not very easy to do it but with binary variables we essentially can do it and then that is what I want to show you now here.

So, suppose that what we have is, we are given this piecewise linear functions in the form of these break points. So, suppose, our input is the following is a data is of the following, we know points a i comma f of a i so what I mean is that, this point here suppose, we have this point. So, this is a 2 and here is f of it so we have given the break points and we are assuming that between a i and a i plus 1, the function is can be linearly extend.

So, this is the whole input that we can get now, let us look at the following so this is a 3 similarly, let us simplify the problem. Now, let us assume that we are between a 2 and a 3 or between a i and a i plus 1 so if x belongs to the interval a i comma a i plus 1 then I can write and suppose I can write x as lambda a i plus 1 minus lambda a i plus 1. Where, lambda lies between 0 suppose, I can do do that then I i know that f of x is going to be given by the linear approximation (( )).

So, between so if x is somewhere between a 2 and a 3, then I can look at f of a 2 f of a 3 and do linear interpolation and then see that, that is the value of a i plus 1.

Student: (( ))

Yeah, that is why

Student: (( ))

Yes.

So, so we know that so now, what we are going to do is, we are going to exploide this fact and then write particular mixed integer programming model for for doing. So, what we are going to do is, we are going to keep track of which piece is, does the point lie. So, if we know which piece the point lies in, then we know that from the end points we can in fact calculate the function value. So, we are going to define the following binary variables for that, so binary variables let me call them as z i.

Z i is going to be 1, if my x belongs to this interval, open in the right because I want to avoid confusions and up to n minus 1 and 0, otherwise. So, using these z i's we can keep track of which which piece we are in and then let us try to, suppose suppose, we know which piece we are in. Then then we can we can really write down what the constraints are so for example, now what we could do is, we could write the following. So, let us let us first write something and then see what is wrong with it and then we can try to, you know make it, in order to, correct it.

So, let us say that, let suppose, we write we write x as summation i going from 1 to n lambda i a i, summation or greater than or equal to zero. So, essentially what I am doing is, I am writing I am writing some my x has some convex combination of of all these x i's. What I want is the following, I want to write f of x as summation i going from 1 to n lambda i f of a i. However, I cannot do that directly because I don't know which piece my x lies.

If only I know that it lies in which piece, then then then I can make sure that this is that so in order to, do so in addition to this we need some extra conditions. So, let let me call these as, this is the part of formulation so let me call this as star then we will see what what additional things we need. So, what additional things we need, now, we also need to specify that, you know x lies exactly in 1 piece. So, we need to make sure that z i is 1 only for one particular i n, not for any other i's. So, in addition we need the following one, exactly one z i is 1, others are 0 and two, we also need to be show that if z i is 1 then x lies, so what what what is z i, z i is 1 if x lies between a i and a i plus 1. If if that is the case, we need to make sure that I need to write x as a convex combination of a i and a i plus 1 right.

So, I need to make sure that all the other corresponding lambdas are equal to 0 so here is what I need to make sure, if z i is 1 then lambda i and lambda i plus 1 can be positive but all other lambda j's are 0, this is what I want to enforce. And let us see how to do that so so enforce one, is fairly routine so to enforce one, I can easily write summation i goes from 1 to n minus 1, z i is equal to 1 and of course, this will make sure that exactly one of these z i's is is equal to 1. But, for the other part so I am adding all these, these are all integer variables they must sum to 1. So, exactly one of them is 1, others have to be 0 and for the other for the other one, there is a really need trick so let me tell you what it.

Student: (( ))

Because, I said that whether x is in the interval a i comma a i plus 1 so only upto n minus 1 so it is just a notation, nothing. So, for the other part here is here is how

Student: (( ))

Yes.

Student: (( ))

I do not understand your question.

Student: (( ))

Yes.

Student: (( ))

No, z i's are only defined upto n minus 1.

Student: (( ))

So, here is the is the way to do it so let me just state what it is and then we can discuss for a minute. So, lambda 1 is less than or equal to z 1, lambda i is less than equal to z i minus 1 plus z i where, i goes from 2 to n minus 1 and finally, lambda n is less than or equal to z n minus 1. Now, let us see what happening here now, suppose let us say, z 1 is 1 others are 0, what happens. So, z 1 is 1, lambda 1 can be positive, lambda 2 less than or equal to z 1 plus z 2 so lambda 2 can also be positive.

But, all other z i's are 0 so all other lambdas must be equals to zero similarly, that is why we we can make make sure that if z i is 1, then exactly lambda i and lambda i plus 1 can be positive. But all the remaining lambdas must be equal to 0, if we use these, so if I call these as say (( )), so all these together the piecewise linear function. So, I am stop so at least I hope I have been able to convince you that, there are some situations where, although the discrete variables are not part of the model, intrinsically part of the model. Usually we have to use them in order to, write these problems as optimization problems, so there are lots of things like that.

Thank you.