**Advanced Operations Research**

**Prof. G. Srinivasan**

**Department of Management Studies**

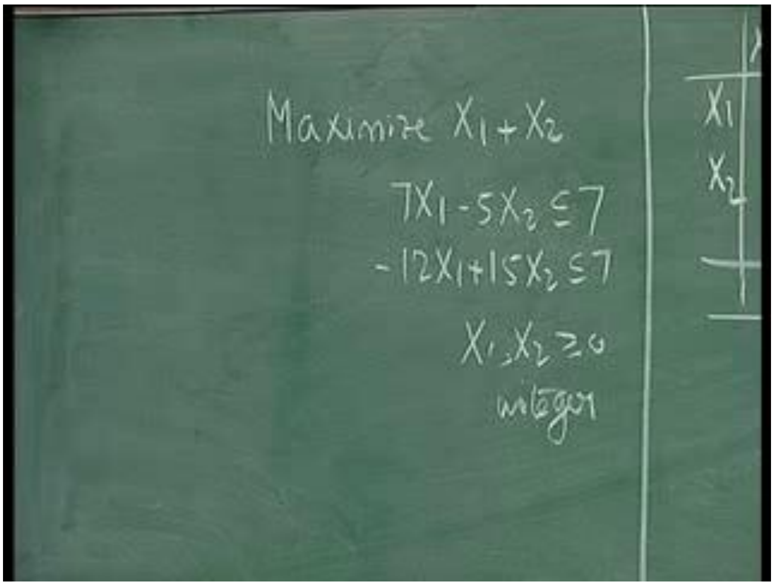
**Indian Institute of Technology, Madras**

**Lecture - 16**

**Cutting Plane Algorithm**

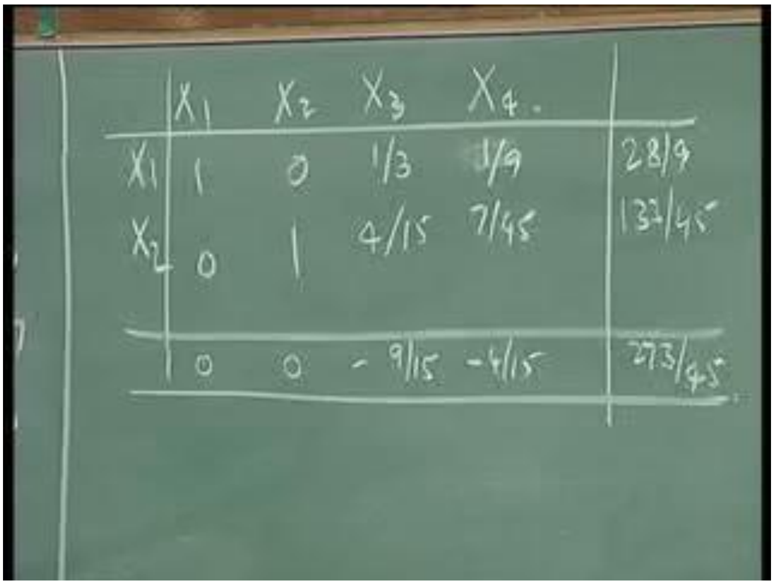
We shall continue the discussion on integer programming, so we take this example.

(Refer Slide Time: 00:20)



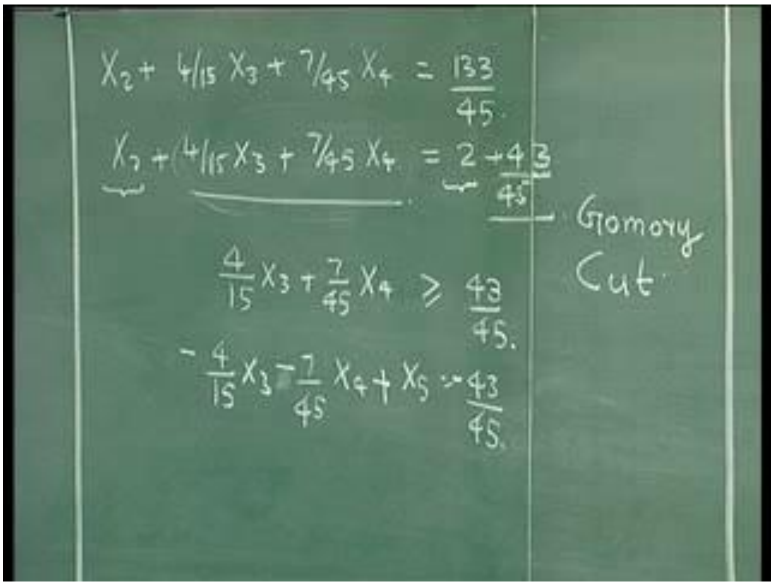

The first rule in solving any integer programming problem is to relax the integer restriction, treat it as a linear programming problem and solve it. If it turns out that the LP optimum is an integer valued, then it is also optimum to the IP. We try solving this problem by relaxing the integer restriction, making it a linear programming problem and we get this.

(Refer Slide Time: 00:49)




The optimum solution to the corresponding LP is $X_1$ equal to 28 by 9, $X_2$ equal to 133 by 45, Z equal to 273 by 45. Both $X_1$ and $X_2$ have non-integer value or fractional value; we can take any one of them for further consideration. We choose that variable, which has the largest fractional coefficient, as a general consistent guideline. So we take variable $X_2$ and write the constraint corresponding to the basic variable $X_2$, which is the second constraint, from the simplex table, and we get $X_2$ plus 4 by 15 $X_3$ plus 7 by 45 $X_4$ equals 133 by 45.

(Refer Slide Time: 01:28)



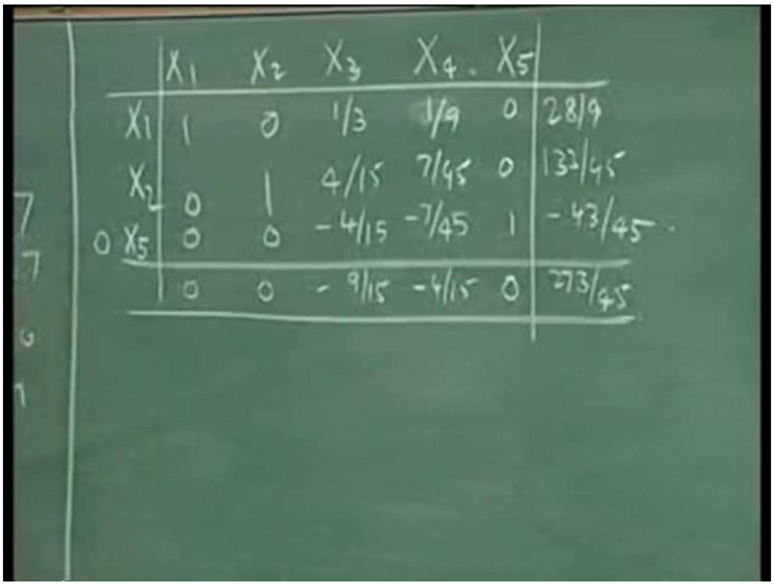
Now, this equation is rewritten in a certain form, wherein the right-hand side value, because, this is an optimal simplex table or an optimal table in a simplex iteration, this right-hand side should always be non-negative. This is written as an integer plus a fractional value less than 1. So this becomes 2 plus 43 by 45. Left-hand side, the coefficients are again written in the same form: an integer plus a positive fractional value. In this case, it turns out $X_2$ is an integer, so $X_2$ remains; 4 by 15 has a fractional value which is positive, which is less than 1; so it is straight away written as 4 by 15 $X_3$. For example, if this had been say 24 by 15, then we would write it as 1 plus 9 by 15; $X_3$ plus 9 by 15 $X_3$. This again becomes 7 by 45 $X_4$. For example if it had been say minus 7 by 45 then, we would write it as minus $X_4$ plus 38 by 45 $X_4$. So, it should always be written as an integer plus a positive coefficient less than 1. The integer could be negative; if the coefficient itself is negative, then the corresponding integer will be negative, but you will have a positive coefficient less than 1. So we have written this.

Now, we try to match the integer part on the left-hand side with the integer part on the right-hand side and the fractional part on the left-hand side with the fractional part on the right-hand side. So this is the only integer part, this is matched against this (Refer Slide Time: 03:34). This whole thing is to be matched against this. Now, we know that this part of the equation has strictly positive coefficients for each of these variables. This is clearly a positive term. The right-hand side has a fractional portion, which is also a positive term. From this, we can write an inequality which looks like this: 4 by 15 $X_3$ plus 7 by 45 $X_4$ is greater than or equal to 43 by 45. The greater than or equal comes because, this is an integer, this is an integer and this is an equation; so, the fractional portion will have to match. So 4 by 15 $X_3$ plus 7 by 45 $X_4$ will have a fractional portion which has to be 43 by 45, but this is a positive quantity; so this could be simply 43 by 45 or 1 and 43 by 45 or 2 and 43 by 45 and so on. Therefore, we make a general assumption here. We do not restrict it to an equation. We make it an inequality. If we make it to an equation, there can be a situation where we could be wrong, but then you make it an inequality, you are always right.

So you make this constraint from an existing equation (Refer Slide Time: 05:02). This is called a cut and this is called a Gomory Cut based on Ralph Gomory, whose algorithm we are right now seeing. Then we rewrite the cut. The cut is written as 4 by

15 $X_3$ plus 7 by 45 $X_4$ minus $X_5$ equal to 43 by 45; from which, $X_5$ minus 4 by 15 $X_3$ minus 7 by 45 $X_4$ equal to minus 43 by 45. Multiply this whole thing with a minus 1, you get minus 4 by 15 $X_3$ minus 7 by 45 $X_4$ plus $X_5$ equal to minus 43 by 45. Now write it exactly here with $X_5$ as a basic variable. Create another row with $X_5$ as basic variable.

(Refer Slide Time: 05:59)

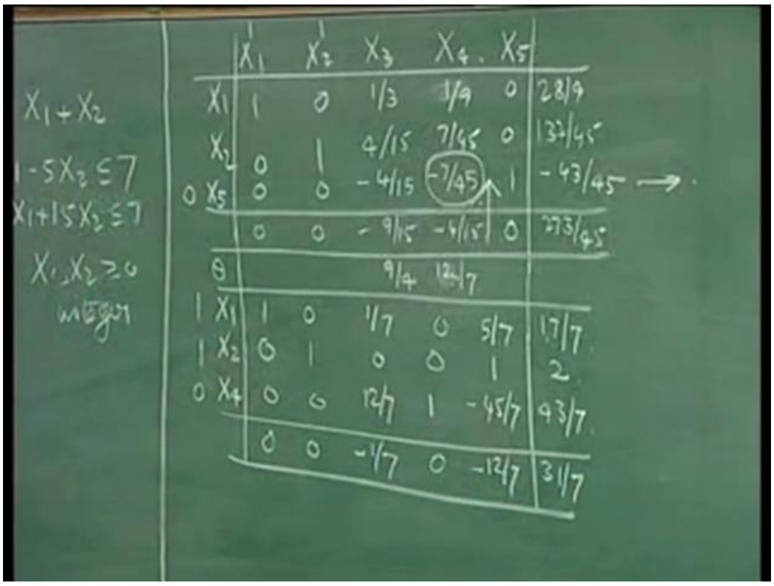



This becomes $0X_1$ plus $0X_2$ minus 4 by 15 $X_3$ minus 7 by 45 $X_4$ plus $X_5$ equal to minus 43 by 45. Introduce an $X_5$ again into the problem. So, a Gomory cut introduces an additional constraint and also introduces an additional variable. The additional variable is always a negative slack variable, because a Gomory cut is always a greater than or equal to cut. So you will get a negative slack and the negative slack will come into the basis; it will come into the basis because you multiply the whole thing with a minus 1; because, a negative slack comes into the basis, it comes into the basis with a plus 1 coefficient, the effect being right-hand side being negative. So the current solution will be infeasible; in all Gomory cuts the current solution will be infeasible, you will be adding a new constraint and the very fact that you should have a negative here, it indicates here you are doing it all right.

Now introducing variable $X_5$, $X_5$ has 0, 0, 1 that come here. $X_5$ becomes a basic variable with 0. Because, this is a basic variable with 0 and in the Gomory cut you always write the new variable in terms of existing non-basic variables, like what we have done, because you take it from an equation in a simplex table; any equation in a simplex table represents a basic variable in terms of the non-basic variables and because, we do it that way and we put this $X_5$ in, the $C_j$ minus $Z_j$ will not be affected by the addition of the new variable. Because this has a coefficient of 0, the values come only against the non-basic variables. The coefficient in objective function is 0, so $C_j$ minus $Z_j$ is not affected. Just because it is a negative slack with a contribution of 0, $C_j$ minus $Z_j$ is not affected. So optimality condition does not change, feasibility is affected; so you go ahead and do a dual simplex iteration on this.

Do a dual simplex iteration by pushing this out and calculate a theta. Right now, you have to calculate a theta, let us do that; since we are doing it first time, let us calculate a theta.

(Refer Slide Time: 08:40)




This is a maximization problem, so minimum theta straight away comes. In a dual simplex iteration, pivot has to be a negative number. So 9 by 15 divided by 4 by 15 is 9 by 4 and 4 by 7. So this enters and this is your pivot (Refer Slide Time: 09:02). Now go back and do one more iteration here. 9 by 4; I must multiply by 3. This is 12 by 7 and still smaller; so 12 by 7 enters, you have $X_1$, $X_2$ and you have $X_4$ and do a straightforward iteration.

The contribution will be 1, 1, 1, 1 and 0. Let me just write down the next table here, just to save time. You will get 1, 0, 1 by 7, 0, 5 by 7, 17 by 7; 0, 1, 0, 0, 1, 2 and $X_4$

has 0, 0, 12 by 7, 1, minus 45 by 7, 43 by 7 and this will be 0, 0, minus 1 by 7, 0, minus 12 by 7, 31 by 7. This is the table that we have after one iteration.

Yes, it happens in this case, but later you will see that you need not get the same thing in the branch and bound as well as in the Gomory. It can be different. We will see it a little later after we complete it.

The implied question is: does this reduce to $X_2$ less than or equal to 2, in which case we did exactly that in the branch and bound. So are we doing the same thing? The answer is: later we will see that this actually reduces to $X_2$ less than or equal to 2 and we will verify that. Whether it implies $X_2$, the same constraint or whatever, the Gomory cut will not and need not be the same as the branch and bound cut. You will realize later once we see. In the branch and bound algorithm, at every stage or at every node, the cut is always an upper bound or a lower bound on the variable. You will always get cuts of the form $x_k$ less than equal to some $l_k$ or greater than equal to $l_k$ or $x_k$ less than equal to $u_k$.

In a Gomory cut you could have a general constraint. This can reduce to, for example, some $3X_1$ plus $2X_2$ less than or equal to constant. It could reduce to something like that. Please remember the way this problem is written, it is also a very general assumption that in integer programming problems if the right-hand sides are integers, then the slack variables are also bound to be integers; otherwise, you are in a slightly different spot. For example, you may have a situation where decision variables are integers and slack variables need not be; you may get into that situation. In general terminology, when you say something is an all integer algorithm, it implies that decision variables are integers, right-hand sides are integers, which means slack variables are also integers. So decision variables and slack variables are integers.

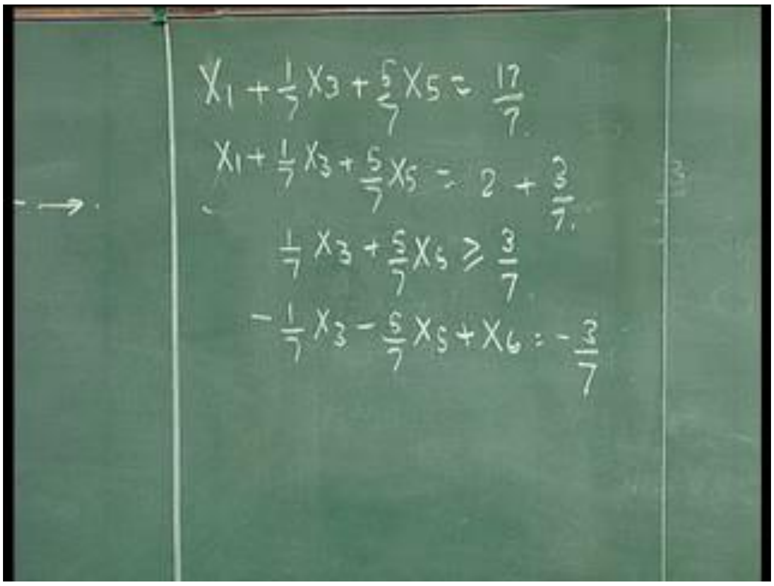
Now what happens to the slack variable corresponding to the Gomory cut? The slack variable corresponding to the Gomory cut should also be an integer, because, this is the governing constraint from which this equation is written. When you match the left-hand sides and the right-hand sides, you know that this quantity is either 43 by 45 or 43 by 45 plus an integer; it cannot be anything else. Therefore, the minus $X_5$ that comes in here it becomes plus $X_5$ after you multiply with a minus 1, should also become an integer greater than or equal to 0.

You will have all your variables, original slack variables in the problem plus every slack variable corresponding to every Gomory cut will be greater than or equal to 0 and integer. So, the moment we say we are solving an all integer problem, we are assuming all that. Now go back, we realize $X_1$ and $X_4$ which are meant to be integers are now having fractional values. Please remember, that at the optimum, all the basic variables will have to be integers. Non-basic variables are at 0; so we do not worry about them, they are having an integer value, but all basic variables have to be integers. For example, it may not occur, but if it happened in this iteration, that $X_1$ is 2 and $X_4$ has a fraction or if we had $X_1$, $X_4$ and $X_3$ with $X_4$ and $X_3$ having fractional values and $X_2$ having an integer value, we have still not reached the optimum. At the optimum, all basic variables will have to take integer values. So we look at these two and then find out the variable that has, once again, the largest fractional coefficient and generate another Gomory cut.

Yes, understood. The point I was trying to convey is the message that if you are looking at an all integer problem, then you will have; thinking along with the problem this statement is right because $X_3$ and $X_4$ are slack variables that come and only one constraint. Absolutely; it is right, but then the point I was trying to convey is that it is absolutely necessary that all basic variables have to be integers. So you take $X_4$ which comes here and write the corresponding.

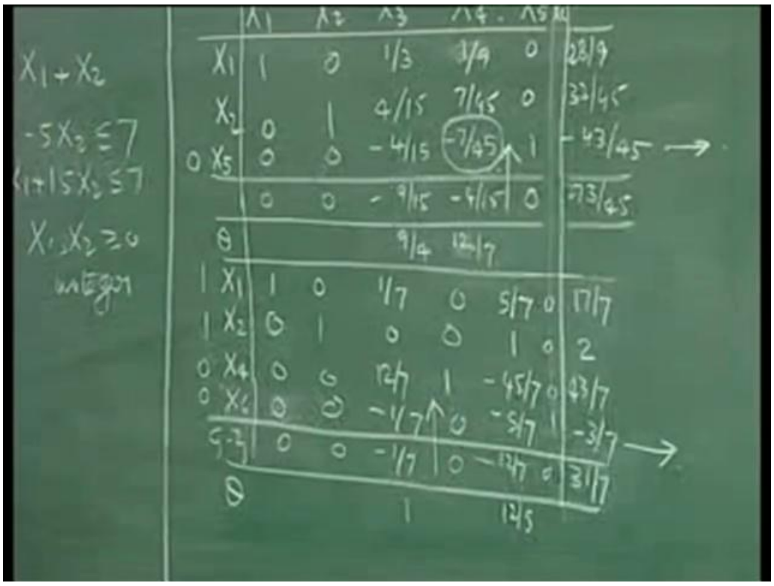
$X_1$ is 17 by 7, now this is 1 by 7, this is 3 by 7. So we will take this (Refer Slide Time: 15:47). This would help us in understanding something, but we will still be consistent in taking this. If you have taken this, then you have a minus term here. So when you write the Gomory cut, this will become minus 1 or minus 7 plus something. This would become something else.

(Refer Slide Time: 16:09)




So we will still go by $X_1$ and write $X_1$ plus 1 by 7 $X_3$ plus 5 by 7 $X_5$ equal to 17 by 7. This simply becomes $X_1$ plus 1 by 7 $X_3$ plus 5 by 7 $X_5$ is equal to 2 plus 3 by 7, from which the Gomory cut will be 1 by 7 $X_3$ plus 5 by 7 $X_5$ is greater than or equal to 3 by 7, which would give us minus 1 by 7 $X_3$ minus 5 by 7 $X_5$ plus $X_6$ equal to minus 3 by 7. This will be the Gomory cut corresponding to this solution.

(Refer Slide Time: 17:15)

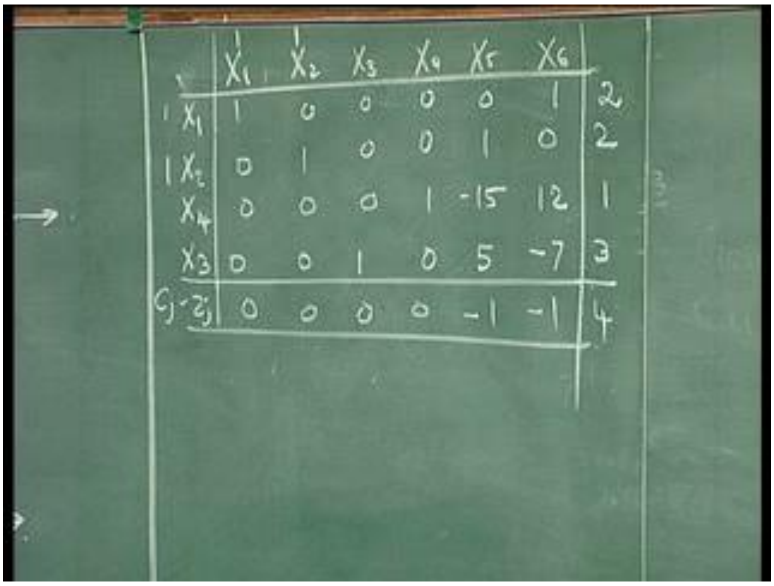



We have to write this again. I need to create some more things here. You need to create it little carefully. Just introducing an $X_6$ here. $X_6$ gets 0, 0, 0, 1. I have an $X_6$

here with a 0 here, 0, 0, minus 1 by 7, 0, minus 5 by 7, 1; you get 17 by 7 and what do you have here? 2, 43 by 7, minus 3 by 7. I have not written the $C_j$ minus $Z_{j;}$ we did not compute the $C_j$ minus $Z_j$. So 0, 0, 0; this is 0, 0, 0, 0 (Refer Slide Time: 18:20). This will be minus 1 by 7, 0, 5 by 7 plus 1 is 12 by 7. So I have a minus 12 by 7, 0; 2 plus 17 by 7 is 31 by 7.

Now we do one more iteration. Again, it is a dual simplex iteration. Every Gomory cut results in a dual simplex iteration, so this goes. We need to find a theta; so minus 12 by 7 divided by minus 5 by 7 is 12 by 5; 1 by 7 1 by 7 is 1. So this variable enters, so $X_3$ enters again.

(Refer Slide Time: 19:07)




The next simplex iteration, we will do it here. We will have $X_1$, $X_2$, $X_3$, $X_4$, $X_5$, $X_6$; with $X_3$ replacing $X_6$, so $X_1$, $X_2$, $X_4$, $X_3$. This is your pivot element; divide it by the pivot you get 0, 0, 1, 0, 5, minus 7 and 3 here; this minus 1 by 7 times this or simply this plus this. This $R_1$ minus 1 by 7 times this row is the same as simply adding this and this. You would get 1, 0, 0, 1 by 7 minus 1 by 7, again 0; 5 by 7 minus 5 by 7, 0; 0 plus 1, 1; 17 by 7 minus 3 by 7 is 14 by 7 which is 2.

This is all right, there is no problem here; there is already a 0; so write it as it is 0, 1, 0, 0, 1, 0, 2; this plus 12 times this will give me a 0. So I have 0, 0, 0; this plus 12 times this is still 1; minus 45 by 7 plus 12 times, so minus 105 by 7, which is minus 15. This plus 12 times this is 12. 43 by 7 minus 36 by 7 is 1 and you have reached the optimum. Because it is a dual simplex iteration, you do not have to do the $C_j$ minus $Z_j$ again. It is a dual simplex iteration, so optimality condition will anyway be satisfied. The moment you get your right-hand side's integer value, you have reached the optimum. For the sake of completion, you would still have 1, 1 coming in here; 1, 1 coming in here (Refer Slide Time: 21:48); 0, 0, 0, 0, minus 1, minus 1, 4. So, you have the same optimal solution with $X_1$ equal to $X_2$ equal to 2 and Z equal to 4.

What are the other issues in the Gomory cutting plane algorithm? One, in every iteration, when you have a non-optimal solution, when the LP optimum is non-optimal to IP, then you need a Gomory cut and every Gomory cut is going to result in an additional constraint and an additional variable. Therefore, the time taken per iteration is gradually going to increase, because, subsequent iterations are going to have more constraints. The other advantage perhaps is you still need to do an iteration, but you could think in terms of a sensitivity analysis type, because, every Gomory cut is like adding a binding constraint to an existing LP optimum; so you can treat it like sensitivity analysis.

Since the Gomory cut does not ensure that you have a constraint which is a bound, for example, a Gomory cut could become something like some $a_1x_1$ plus $a_2x_2$ less than or equal to $b_1$ or greater than or equal to something. It does not become a bound on the variable, therefore unlike in branch and bound, you cannot just impose an upper bounded simplex algorithm on it; you need to treat it like adding a constraint, which is like sensitivity analysis and then you proceed. This is called the Gomory's cutting plane algorithm and every cut is a Gomory cut.

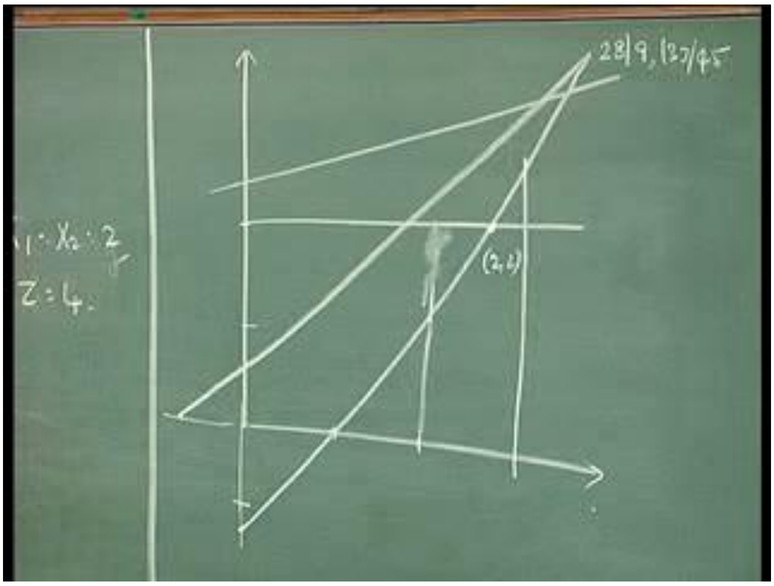
The Gomory cutting plane algorithm converges to the optimum, if an optimum exists in a finite number of iterations. But there is no proof or there is no result saying or having an upper limit on the number of iterations. When such a similar thing for simplex is exponential, you do not expect a nice polynomial bound on the number of iterations for the Gomory algorithm; you do not. The only nice thing about the algorithm is that it will eventually converge. That is all. There is no backtracking, there is no fathoming, there is nothing. In terms of storage, it is again a peculiar thing. You might get into a situation where, as far as storage is concerned, it may behave better than branch and bound. But as far as running time is concerned, depending on the problem, you may get into a situation where branch and bound is better, because, time per iteration or time per node would remain more or less constant in a branch and bound, whereas time per iteration varies in a Gomory cutting plane algorithm.

Next, there is absolutely no guarantee that choosing a variable that has the largest fractional coefficient would terminate faster; but both experience and a greedy approach to the problem would make us consistently choose that variable, which has the largest fractional coefficient; done more for uniformity and consistency and not for anything else. The most positive $C_j$ minus $Z_j$ is accepted based on experimentation; this is also accepted based on experimentation. You normally try to create a Gomory cut for a variable that has the largest fractional coefficient. You could branch or you could use any other variable also to create a Gomory cut.

A Gomory cut is always characterized by a negative value on the right-hand side. When you write your Gomory cut, you should have a negative value on the right-hand side. If you do not get one, then you have made a mistake somewhere. Now, let us go back and do a couple of other things.

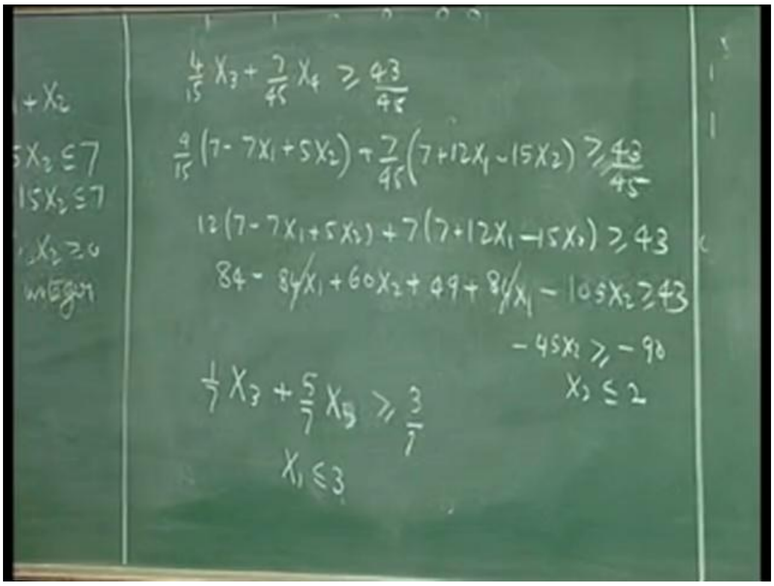
How does the algorithm go towards an integer solution? We will do that now by mapping it on the graphical method. We will just draw the graph corresponding to the original problem. We have seen two Gomory cuts, we will go back and see what the two Gomory cuts represent or how they are and what is the difference between a branch and bound cut and a Gomory cut? Let us draw the graph as it is here.

(Refer Slide Time: 26:53)



I think we had something like (1,0) here and the other one turned out to be minus 1.4, which could be here. So the first constraint was something like this. Second constraint is 7 by 15 on $X_2$. So this is somewhere here and minus 7 by 12 on $X_1$ is somewhere here. We got something like this; with the corner point 28 by 9 and 133 by 45. Now let us go to the first Gomory cut. What was the first Gomory cut in terms of $X_3$, $X_4$ and the right-hand side; you can leave out the $X_5$.

(Refer Slide Time: 28:25)



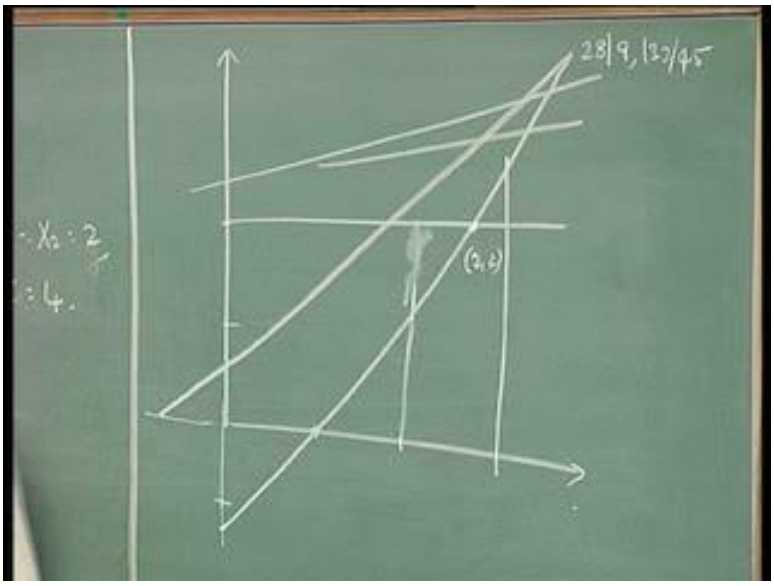

4 by 15 $X_3$ plus 7 by 45 $X_4$ is greater than or equal to 43 by 45. Now go back and write $X_3$ here. $X_3$ is 7 minus $7X_1$ plus $5X_2$. So 4 by 15 into 7 minus $7X_1$ plus $5X_2$ plus 7 by 45 into 7 plus $12X_1$ minus $15X_2$ is greater than or equal to 43 by 45. So multiply the first one, you would get 12 into 7 minus $7X_1$ plus $5X_2$ plus 7 into 7 plus $12X_1$ minus $15X_2$ is greater than or equal to 43. Very quickly simplifying, you get 84 minus $84X_1$ plus $60X_2$ plus 49 plus $84X_1$ minus $105X_2$ greater than or equal to 43. The $84X_1$ gets cancelled, 84 plus 49 is 133, 133 minus 43 is 90; so greater than or equal to minus 90. I have 60 plus 45 is 105, so minus $45X_2$ greater than or equal to 90, gives me $X_2$ less than or equal to 2.

In this case, it turns out that the first constraint is the same as the branch and bound constraint; normally, it will not happen, it need not happen. A Gomory cut need not finally result in this form - like an upper bound or a lower bound; you could have the other one.

We will look at the next one and see what happens. What did we do here? We just added another constraint $X_2$ less than equal to 2 and brought down the feasible region here. Now, what is the next Gomory cut? The next Gomory cut again turned out to be $X_1$ less than equal to 3. What was the Gomory cut? 1 by 7 $X_3$ plus 5 by 7 $X_5$ becomes greater than or equal to 3 by 7; $X_1$ less than or equal to 3. But it just happens again in our example that the Gomory cut is becoming a bound. Please remember that this may not be the best example to illustrate that or it may end up misleading you that Gomory cut is always a bound; it just does not happen. You could always have a Gomory cut for a two-variable problem; a Gomory cut of the form $a_1X_1$ plus $a_2X_2$ less than or equal to $b_1$.

We could have done that. What happens here is $X_1$ less than equal to 3 takes us here. This is here and this becomes the optimum which is (2,2). In a normal Gomory cut, it does not happen this way; this perhaps is not the correct example for it. It also removes areas as the branch and bound does; but what it does is, for example, if this is the optimum you could get a Gomory cut, which could behave like this. Instead of the first cut, you could get a Gomory cut, which is like this. It need not result in cuts which are parallel or perpendicular to the X and Y-axis.

(Refer Slide Time: 32:31)



You could get a cut like this. It kind of takes away areas from the feasible region, it makes the present optimum infeasible. It removes areas and then it progressively chips off areas. Branch and bound is more like a guillotine cut, it just cuts either horizontally or vertically. Gomory is like chipping it off; it just chips off a certain portion of the area, progressively you keep chipping off the areas till the LP optimum is integer value. Both of them try to do the same thing, but branch and bound does it in a certain systematic way, Gomory does it in a different way.

This could be your first cut, maybe this could be your second cut (Refer Slide Time: 33:27); it could happen that way. At times, it can make drastic cuts as it happened in this case, but after some iteration you will realize that the Gomory Cut progressively removes smaller and smaller area. If you work out on large problems, which might be difficult to demonstrate in a class, you will realize that one of the disadvantages of the cutting plane algorithm is that as the number of iterations increases, you will see that it does remove a certain area, as in two-dimension, in every iteration; but as the iterations increase, the extent of gain becomes less and less and Gomory's algorithm takes a little more number of iterations.

If you start mapping it on the graph, you will see that the area that is removed by the Gomory cut progressively becomes lesser and lesser and that is very important. In a branch and bound, what will happen is when you make a cut, this can happen both in Gomory as well as in branch and bound, a variable which at present has an integer value because of a cut in the next iteration, can take a non-integer value. That actually makes the whole thing interesting, because, if you put a restriction that the moment a variable takes an integer value, it cannot take non-integer, it automatically leads you to a way of writing a bound on the number of iterations. That does not happen in any of these. A variable that takes an integer value in a particular iteration, in a subsequent iteration can take non-integer values, both in the Gomory's cutting plane method as well as in the branch and bound method. So this is how the Gomory's cutting plane algorithm actually works.

As I said, there are proofs that this algorithm converges or terminates at the optimum in a finite number of iterations. For large problems, you will realize that the Gomory cuts become weak; weak in the sense that they remove smaller and smaller areas. You will realize that very quickly you are very close to the IP optimum. The gap between the present LP optimum and the IP optimum is very small, but in a single cut it is not able to get you the IP optimum. So it takes you to another cut and once again, you make a cut, you are still in between. As you keep proceeding, the gap just narrows down and finally it gives you the integer optimum. That is how the Gomory cutting plane algorithm works.

It is kind of advantageous to use a branch and bound when you have a mixed integer programming problem, because, the more continuous variables you have, the easier it is for a branch and bound algorithm. You need only to branch on those variables that has fractional values for the integer. You might be tempted to think on the same lines with the cutting plane. Remember that the Gomory cutting plane algorithm that we have seen is meant for all integers. It is not directly meant for a mixed integer, because, remember that every Gomory cut results in a slack variable, which is also an integer variable. It is an all-integer cutting plane algorithm that is what we have seen. This comes under the category of cutting plane algorithms. Cutting plane algorithm is an all-integer algorithm and the Gomory cut is an all-integer cut. It is for an integer algorithm.

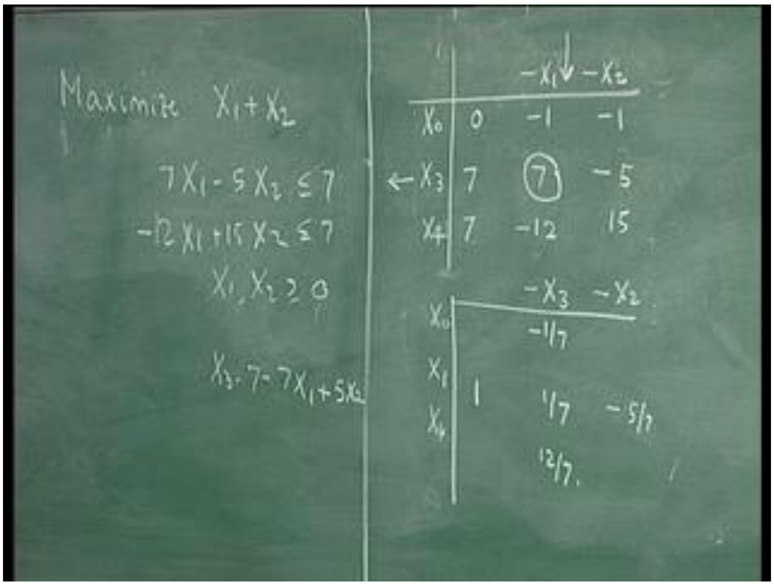
There are modifications of Gomory cuts for mixed integer problems. When you are working on a mixed integer problem and you have an optimum simplex iteration, you will not bother about that variable which can take a continuous value. But if an integer variable takes a continuous value, you cannot create a Gomory cut like what we did. That cut is meant under the all-integer assumption. There are some mixed integer cuts. Mixed integer cuts are slightly different from all-integer cuts. You may find them, but we will not go to that detail. As far as this course goes, we will say that we can use the cutting plane algorithm for all integer and we will use it. For mixed integers, there are modifications to the Gomory cutting plane algorithm, but we would recommend using branch and bound if you were to solve a mixed integer problem, because, branch and bound under the same algorithm and structure, you can solve an all-integer as well as mixed integer. Whereas, if you want to adapt the Gomory's method for the mixed integer, then you have to change the way the Gomory cut is written.

Branch and bound has some advantages when it comes to solving mixed integer. In every node of a branch and bound tree, you only create an upper bounded variable. As

I said earlier, the size does not increase with increase in the number of iterations. In general, for integer programming problems, branch and bound is preferred to the cutting plane, because branch and bound can handle both all integer and mixed integer under the same algorithm, whereas the cutting plane algorithm has to have different cuts for each of these. In branch and bound, somewhere a time taken per iteration is more or less fixed because every cut results in a bound. One can always have situations where the cutting plane algorithm is faster than the branch and bound and so on. Cutting plane may be better in terms of memory and branch and bound may be better in terms of time. So, depending on the problem, one needs to look at both, but as a general rule branch and bound is used more widely than cutting plane, simply because of its ability to handle both all-integer and mixed integer under the same algorithm.

Now, let us look at another thing here. There are two more types of all integer algorithms that we will see here as part of our integer programming. There is an all integer dual algorithm and there is an all integer primal algorithm. We will have to see both these, the all integer dual algorithm as well as the all integer primal algorithm. Before we see that, we will try to represent the Gomory cut or the Gomory cutting plane algorithm in a slightly different format. We will get into another way of representing a simplex table, which is slightly different from what we have been doing; we look at it very carefully. We had seen already different ways of making the simplex table: the simplex table in its regular tabular form that we saw in the first course and then we have seen the matrix form of the simplex using the $e_1e_2$ matrices. Then, in the column generation algorithm we saw one way of representing the simplex table and then, in the decomposition we saw another way of representing the simplex table. Now we look at third way of representing the simplex table. We will take the same example and we solve it. The example is as follows.

(Refer Slide Time: 41:13)




Maximize $X_1$ plus $X_2$; $7X_1$ minus $5X_2$ less than or equal to 7; minus $12X_1$ plus $15X_2$ less than or equal to 7; $X_1$, $X_2$ greater than or equal to 0. Let us take first the linear programming problem and solve it. Then we will look at the integer programming problem and solve it. We will create a much smaller table here. $X_3$ and $X_4$ are the slack variables to begin with and they will become the basic variables.

What we will do is we will have a column called $X_0$, we will then see what this column is or we will have a row called $X_0$. Then we will have here $X_3$ and $X_4$, which are the current basic variables. Then you have a right-hand side and then you have a minus $X_1$ and a minus $X_2$, where $X_1$ and $X_2$ represent the present non-basic variables. This maximize $X_1$ plus $X_2$ would be written as: 0, minus 1, minus 1. You can assume the standard format is for a maximization problem. So I have $X_1$ plus $X_2$ here. I already have a minus $X_1$ minus $X_2$. So the coefficients get multiplied to give me a plus $X_1$ and a plus $X_2$. Now, this will be written as $X_3$ is 7, the equation becomes $X_3$ equal to 7 minus $7X_1$ plus $5X_2$. So 7 minus $7X_1$ will become plus 7 and this will become a minus 5. To repeat, $X_3$ is equal to 7 minus $7X_1$ plus $5X_2$; 7 is written as 7 here; minus $7X_1$ will become plus 7 because of the minus 1 and plus 5 will become minus because of the minus. $X_4$ will become 7 minus 12 plus 15. Now, if you leave this out, this is like writing $Z_j$ minus $C_j$ or this is like writing the dual. Right now, the dual is infeasible here, primal is feasible. Again, this is like writing the dual as it is. We are used to a convention in the tabular form where we write $C_j$ minus $Z_j$, which is actually the negative of the dual. So we say a positive $C_j$ minus $Z_j$ enters for optimality.

You can always interpret it as a negative of the dual, so a positive $C_j$ minus $Z_j$ implies a negative dual. A negative dual implies infeasibility which is non-optimality; therefore, it enters. This implies dual negative, primal positive (Refer Slide Time: 45:13). So primal positive, dual negative, you will do a normal simplex iteration. When the primal is negative, you will do a dual simplex iteration. So you look at that variable where the most negative dual will come in, which is like saying the most positive $C_j$ minus $Z_j$ will enter.

In this case, there is a tie; you can break the tie arbitrarily. So you enter that one like this (Refer Slide Time: 45:37). Now the leaving variable is taken; there is only one leaving variable 7 by 7; here you cannot divide for theta, 7 by 7, so this is the leaving variable. You get into this. $X_1$ comes in. I get a minus $X_3$ here and minus $X_2$ here. I have an $X_0$, I have an $X_1$ and I have an $X_4$. Now, I am going to introduce a few rules with which you can directly make the simplex iteration.

Now this is going to be your pivot. Rule one is pivot always becomes 1 by pivot, so this becomes 1 by 7. Rule two is divide every other element of the pivot row by the pivot. Pivot alone will become 1 by pivot; divide every other element of the pivot row by the pivot; so this becomes 1, this becomes minus 5 by 7. Divide every element of the pivot column other than the pivot, which is anyway written, divide every other element by minus pivot; so, this will become minus 1 by 7, this will become plus 1 by 7 and this becomes 12 by 7. Sorry this becomes 1 by 7 and this becomes 12 by 7.

Now come the next set of rules. The value here will be: this is the pivot column, this is the pivot row (Refer Slide Time: 47:51); so this minus this into this, 0 minus minus 1 into 1 which is plus 1; 7 minus minus 12 into 1, 19. To repeat, 0 minus minus 1 into 1 plus 1, 7 minus minus 12 into 1, 19 and we have to do this similarly.

We will continue this in the next lecture.