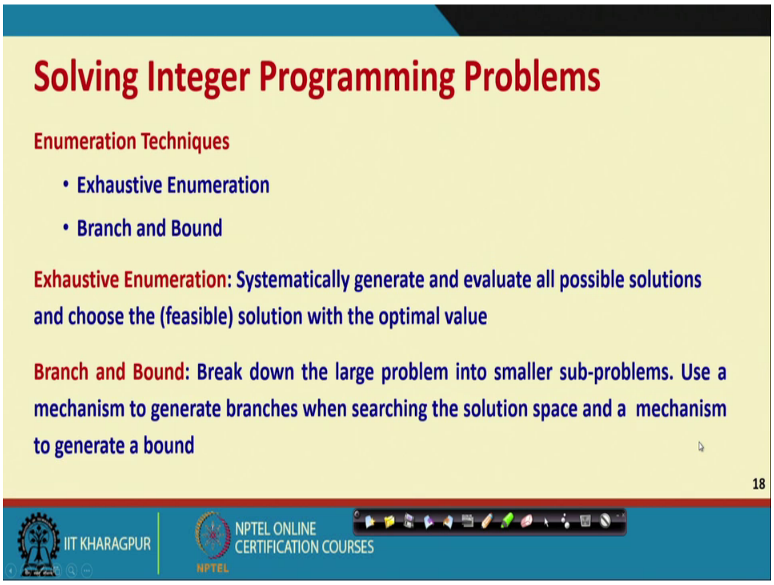
Selected Topics in Decision Modeling

Prof. Biswajit Mahanty
Department of Industrial and Systems Engineering
Indian Institute of Technology, Kharagpur

Lecture - 16
Exhaustive Enumeration and Branch and Bound Techniques

So, in our course Selected Topics in Decision Modeling, today we shall discuss the Exhaustive Enumeration and Branch and Bound Techniques as applied to integer linear programming problems, is it alright. So, what we do here in the exhaustive enumeration and branch and bound techniques, they are two different techniques; and both the techniques are essentially can be called as Enumeration Techniques right.

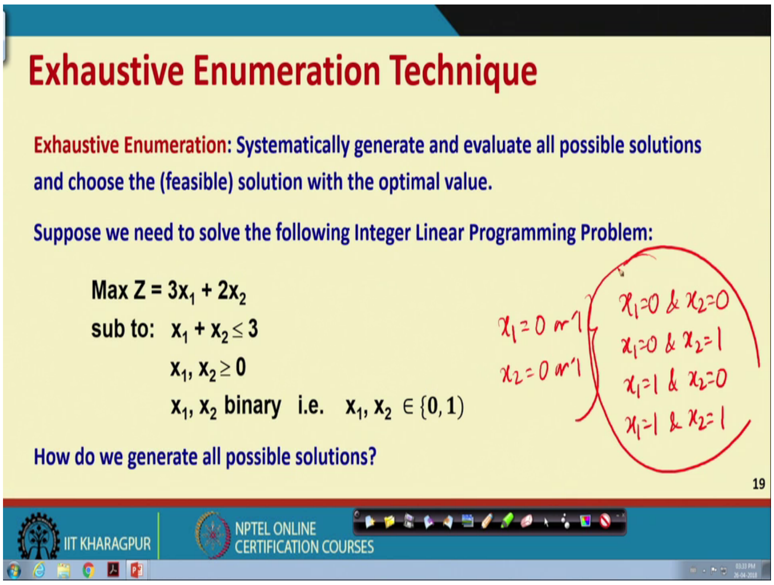
(Refer Slide Time: 00:43)



So, both are enumeration techniques. And what happens in enumeration techniques? Systematically we generate and evaluate all possible solutions and choose that those feasible solutions and you know with the optimal value. So, obviously, out of many feasible solutions one that will give us the best solution will be our optimal solution, so that is how we do exhaustive enumeration. So, we look at all possible solutions.

Whereas, in branch and bound we divide the large problems into smaller sub-problems you know the sub-problems should be divided in a particular way. So, that they are mutually exclusive and collectively exhaustive I will explain that. And then there should be two mechanisms; one mechanism to branch out those sub systems that means break them into smaller subsystems, and a mechanism to generate a bound. The bound means this possible value at you know that sub-problem level and then the examining the bounds you know we try to find out the optimal solution. So, it is very interesting method the branch and bound. So, we will explain that also. But before that let us look at in detail what is this exhaustive enumeration method right.

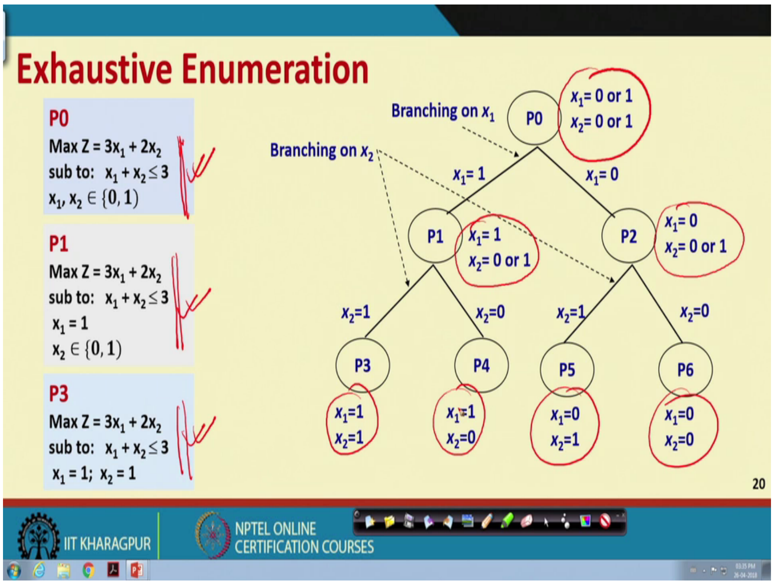
(Refer Slide Time: 02:09)



So, let us take a simple problem systematically how do we generate a linear programming problem? What is that problem? Maximize Z equal to 3 x 1 plus 2 x 2 subject to x 1 plus x 3 less than equal to 3, and x 1 plus x 2 greater than equal to 0, and x 1 and x 2 are binary, is it alright. So, how do we solve this particular problem? What are the all possible solutions? Can you think and tell, very simple problem, x 1 and x 2 are binary.

So, what are all possible solutions? I just think over that since you know it is all binary, so x 1 could be 0 or 1, x 2 also could be 0 or 1 right. So, that means, you know x 1 equal to 0 and x 2 equal to 0, x 1 equal to 0 and x 2 equal to 1, x 1 equal to 1 and x 2 equal to 0, x 1 equal to 1 and x 2 equal to 1. They could be our all possible solutions, so very easy really, so because it is a very simple problem.

(Refer Slide Time: 03:38)

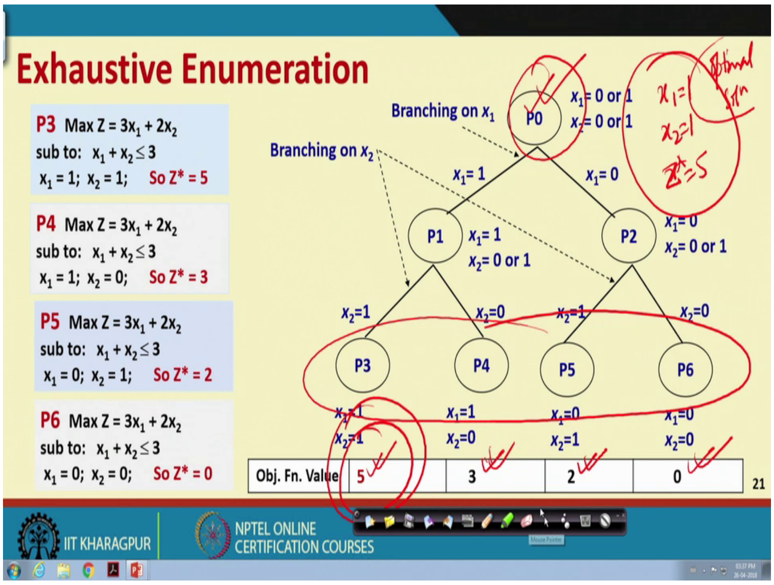


So, that is what we put it in this manner. So, you see at P 0 level x 1 is that is the initial level, we put x 1 equal to 0 or 1, x 2 equal to 0 or 1. Then we branch you know x 1 could be 0 and x 1 could be 1 that will give us a problem which is may be called as P 1. At these x 1 equal to 1, and x 2 is 0 or 1, here x 1 equal to 0 and x 2 could be 0 or 1. Then P 1 is again divided x 2 equal to 0 and x 2 equal to 1 then we get P 3 and P 4 P 5 and P 6.

So, what is P 0? P 0 is the original problem you know very clear. What is P 1? P 1 is a problem which is you know these with x 1 equal to 1. P 3 is a problem which is x 1 equal to 1 and x 2 equal to 1. Similarly, P 4, P 5, P 6. Now, look here the original problem P 0, we would like to solve by enumeration we found that basically the solution of P 0 is the base possible solution of P 3, P 4, P 5, P 6.

So, where we had only one problem, we now have 4 different problems with 4 different solutions, the best ones among them going to be our optimal solution right. So, this is what exactly we do in exhaustive enumeration. We have enumerated all possible n nodes are found; every n node is a problem right that is the original problem with some additional conditions right. But collectively all of these small problems you know not really small conditional problems, they are collectively define the total problem that is what is exhaustive enumeration does right.

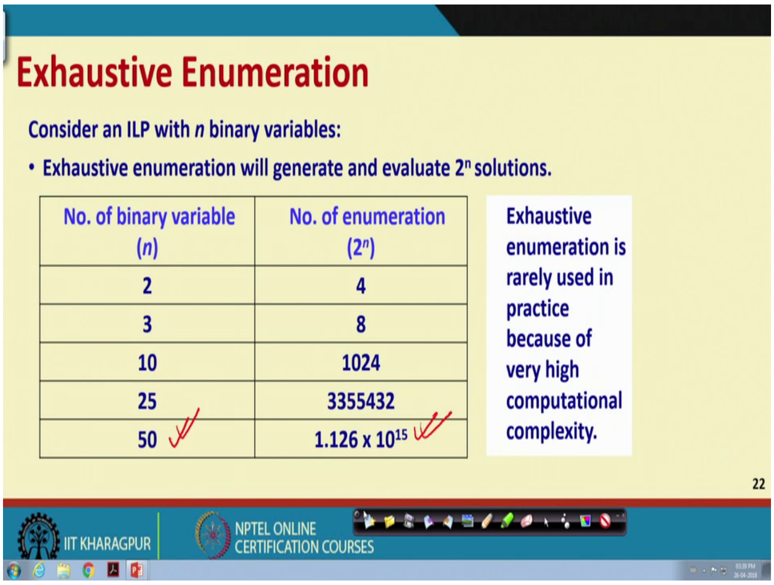
(Refer Slide Time: 05:56)



So, if that is so then you know we can now find out the solutions for these four problems there is easy if x 1 equal to 1 and x 2 equal to 1, z star will be 5. If this is 1 and 0, it will 3; 0 and 1, it will be 2; 0 and 0, it will be 0. So, this is the objective function values written. As I said what is the objective function value for P 0? It is the best possible value out of these four because these 4 problems collectively define all the possible cases of P 0.

Actually we are trying to solve P 0, not P 3, P 4, P 5, P 6. The best solution for P 3 is 5, P 4 is 3, P 5 is 2 and P 6 is 0 alright. Out of that 5 is the best. Since, 5 is the best, now it is clear the best solution for P 0 is x 1 equal to 1, x 2 equal to 1 and z star equal to 5 say our we are not interested in solving P 3, we are interested in solving P 0 right. P 0 is P 3 is 1 of the cases of all possible enumerations and the solution of P 3 comes out to be the best, so that is the optimal solution for our original problem that is P 0. I hope you understood that is how by exhausted enumeration we can solve such problems anyways it is fairly easy its simple problem we took. So, there is nothing much to worry.

(Refer Slide Time: 07:46)

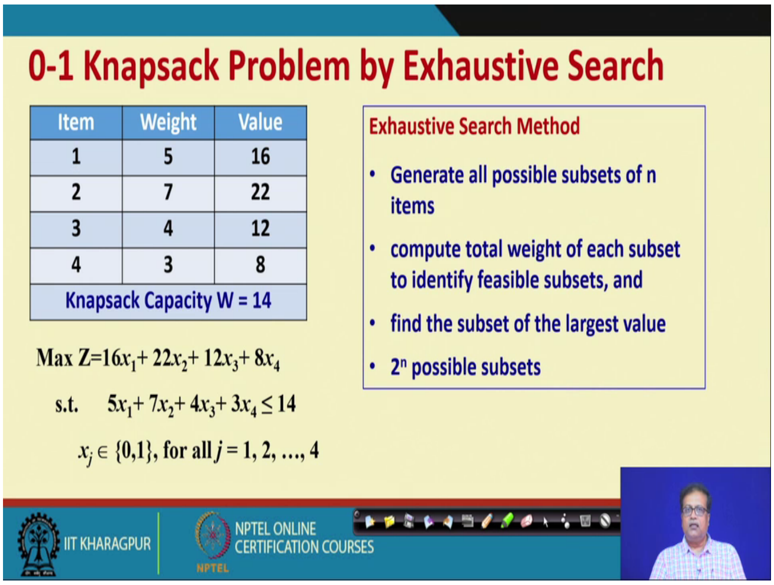


Let us see slightly other issues. The problem is that if we are really doing binary bifurcation. So, if number of binary variables are 2, number of enumerations are 4. If we had instead of 2 variables as 3 variables, number of enumerations would be 8. If we had 10 variables enumerations will be 1024. And when you go to 50, you know if we had 50 variables, then the binary enumerations would have been 1.126 into 10 to the power 15. You can imagine it is a high number right.

So, if you do exhaustive enumeration, you know it might lead to very high computational complexity, is it ok. You know you may have to really evaluate a very large number of problems. You can say I have big computers what is the difficulty, but how big is the big there could be problems in reality which may have 1000 binary variables. And then the problems size would be so large that you may require years, 10 years, 20, 30 years to solve a large problem.

We do not have that much time. You need to solve a problem in a reasonable time. So, exhaustive enumeration or a kind of brute search is a method, but not always applicable because it makes the problem computationally so large so complex that we simply cannot think about it. We must have efficient method right. And one such efficient method is branch and bound.

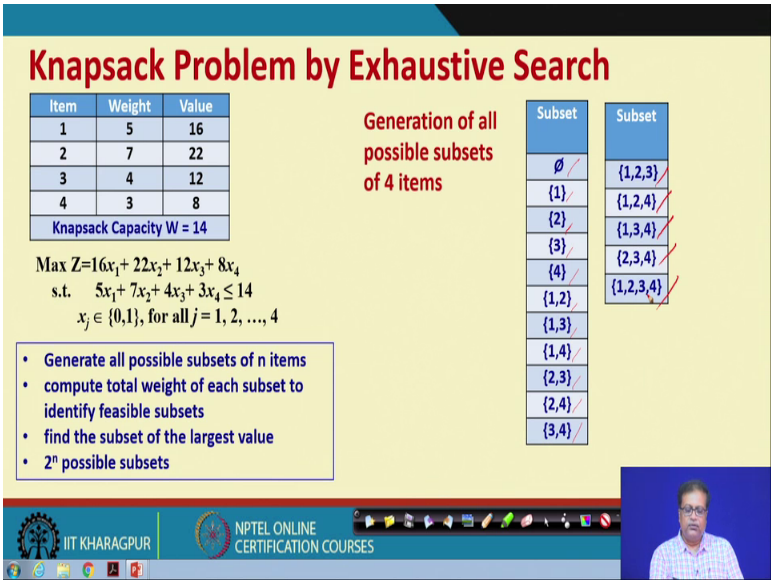
(Refer Slide Time: 09:45)



But before going to branch and bound let us see one more example. So, let us say 0-1 knapsack problem by exhaustive search. Let us say we have 4 items item 1, item 2, item 3 and item 4 with a weight and a value. And this weight and value you know we need to maximize the value within a certain knapsack capacity. How do we go about, obviously the formulation is maximize Z equal to the total value 16 x 1 22 x 2 12 x 3 and 8 x 4 subject to the weight restriction that is 5 x 1 7 x 2 4 x 3 plus 3 x 4 less than equal to 14.

And additionally this is 0-1 knapsack; that means, each item can be either there or not there as simple as that right. So, generate all possible subsets of n items that is how we shall go about that all possible combinations right. And compute total weight of each subset and see whether it is visible or not visible. Then find the subset of the largest value 2 to the power n possible subsets that will be generated, so that is how we go head with this 0-1 knapsack problems right.

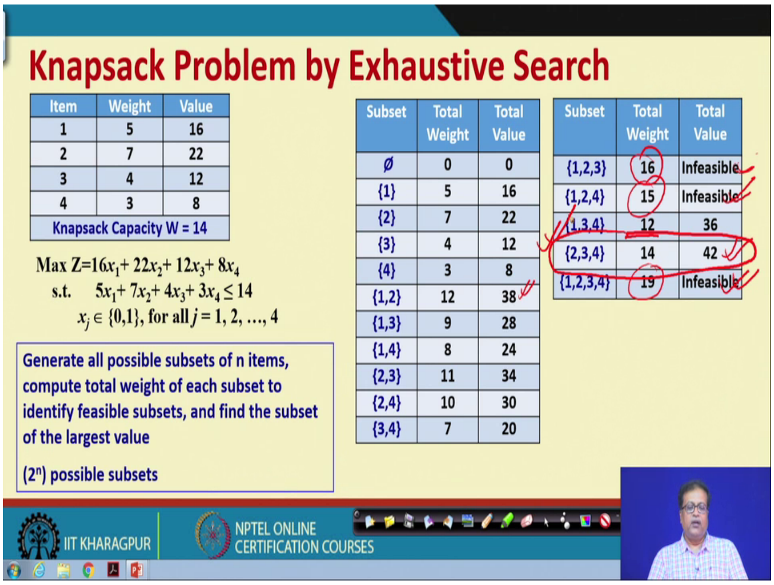
(Refer Slide Time: 11:08)



So, let us see what are some possible generations? So, look here these are the generations, first of all you can have null set then you can take only 1; we can take 2; we can take only 3 or 4. Then we can take combinations like, 1 and 2; 1 and 3; 1 and 4; 2 and 3; 2 and 4, and then 3 and 4 right, 6 combinations. And then also we can have 1, 2, 3; 1, 2, 4; 1, 3, 4, and 2, 3, 4.

And finally, all 4 of them 1, 2, 3, 4 so that is 1, 2, 3, 4, 5 then 611; then 415, and 116 is it not. So, that is what 2 to the power n that is 16 possible combinations that we could have under the situation that is 2 to the power 4. So, 16 possible combinations are there. Now, what we can do for each combination we can find out what is the weight, and what is the value. And based on the weight whether it is becoming you know feasible or infeasible, because if the weight is more than 14, then it will become infeasible fine.

(Refer Slide Time: 12:31)

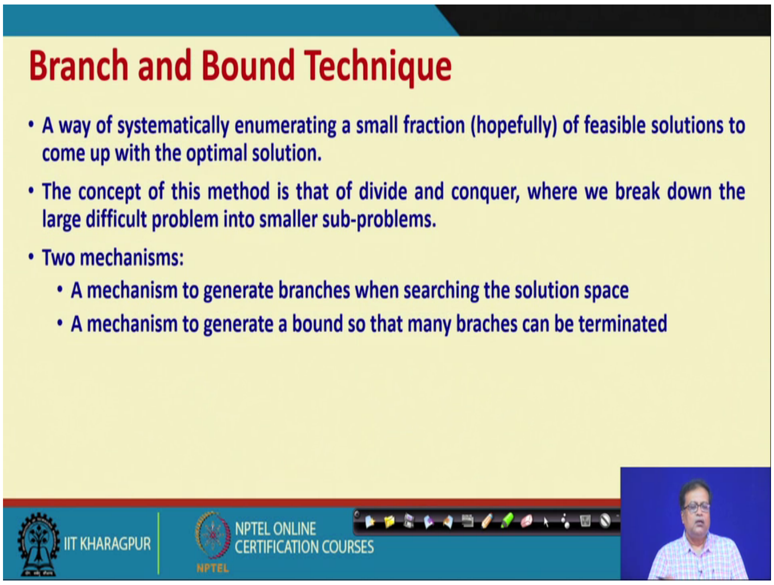


So, then we see what where we reach. So, we will see that as we evaluate then we find for null set both are 0 for 1, 2, 3, 4 it is only the weight and value as given. If we take 1 and 2, then will become 5 plus 7 12 and value is 38 like this for each of these we can write down the weight and the value. We see that total weight crosses in some situations for example, here and here the total weight is crossed.

So, this since they are crossed they are all infeasible right, these solutions are not practical. So, out of these remaining solutions which one shows the maximum possible value, just look at carefully this 38 is good you know, but this 42 is the best. So, the best solution comes in the combination where we take the items 2, 3, and 4 with a total weight of 14 that is within the knapsack, and with the value of 42 is it alright. So, that is our best possible solution.

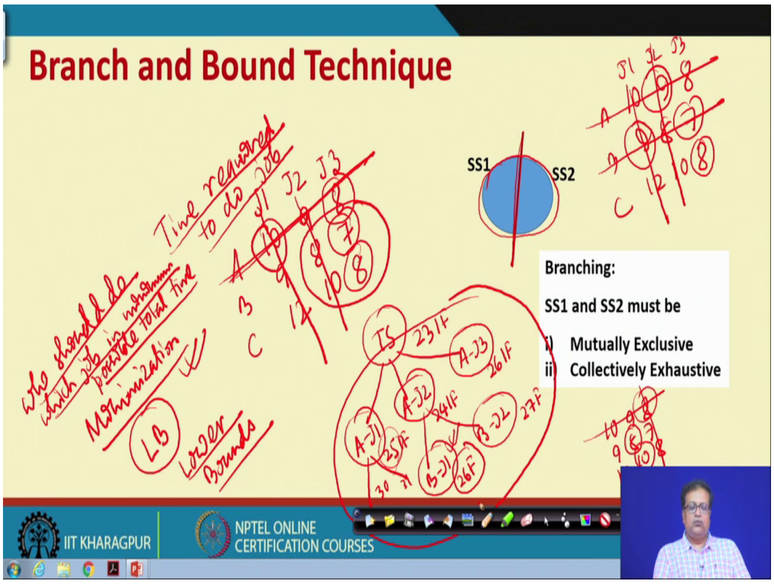
So, by exhaustive search we can find out the problem of the knapsack problem, but for only four items we had 2 to the power 4 different searches think of 10, then it will 2 to the power 10 right, 20 items 2 to the power 20. So, there are lots and lots of such you know combinations, which we have to search, and it could be complex very complex and computationally challenging. So, may not be possible to solve right. So, let us see then another way of solving these problem which may be called as branch and bound technique.

(Refer Slide Time: 14:33)



So, how this branch and bound technique is and how we go about it. Again I repeat how what is really a branch and bound method is that you know, specifically a divide and conquer method, so we break a large problem into a smaller sub problems. And you need as I said two mechanism a mechanism to generate branches; mechanism to generate a bound. So, that many branches can be terminated. So, how do we go about it?

(Refer Slide Time: 15:10)



So, that is our branch and bound technique. So, as you see you know this look at these example what we have done supposing this is our total problem the total problem is you know this is our total problem. The total problem is divided into two sub-problems S S 1 and S S 2, and S S 1 and S S 2 must be mutually exclusive, but collectively exhaustive right, mutually exclusive, but collectively exhaustive.

See it is not necessary that we have to divide into only. Let us take a simple situation, supposing I have three people A B C and these three people are doing three jobs, J 1, J 2 and J 3. Now these are the times which are taken by A B C alright. So, who should do which job? So, this is the time required to do job right. A does the job 1 in 10 minutes or 10 hours; B does in 9 hours; C does in 8 hours sorry. A does the job 1 in 10 hours; job 2 in 9 hours job 3 in 8 hours; B does in 9, 8, and 7; and C does in 12, 10, and 8 hours. Now who should do which job, so that we can do all the three jobs in the minimum possible time.

So, first of all we have to do if we have to solve this problem, by branch and bound technique. First of all we should be able to quickly find a bind a bound. How do I found bound? So, this is a minimization problem, because we need to find out who should do what minimum possible total time. This is what we want, who should do which job in minimum possible total time. So, it is a minimization problem and we really find out then the lower bounds LB, what is LB lower bounds.

So, how do I find out the lower bounds, so can you quickly find out a lower bound from this matrix. Suppose if we find out row wise then 8 mean hours, that is lowest; 7 hour that is lowest; and 8 hour that is lowest. So, 8, 7, 8, 23 hours is the lower bound you know. So, you see if you know only each only job 3 would have been done by A and B and C the job could have been done in 23 hours, but then we need to do all the jobs.

So, just doing J 3 is not enough. So, these 23 is a lower bound no problem at all, but these 23 is infeasible right. So, we have found a lower bound 23 which is an infeasible solution. Now, how do I make branches, so supposing we make three branches. What are the branches, A dash J 1; A dash J 2; and A dash J 3. You see A has to do either J 1 or J 2 or J 3, there are three people three jobs. A has to do either one, supposing A dash J 1 then this line is gone, that means, A has to do J 1. So, out of the remaining this is these matrix, which is the lower bound again we have 7 and 8, so 15 and this 10, so 25. So, this 25 is the lower bound again it is infeasible. Why, because B and C both are doing job 3, so, it is an infeasible solution is it alright.

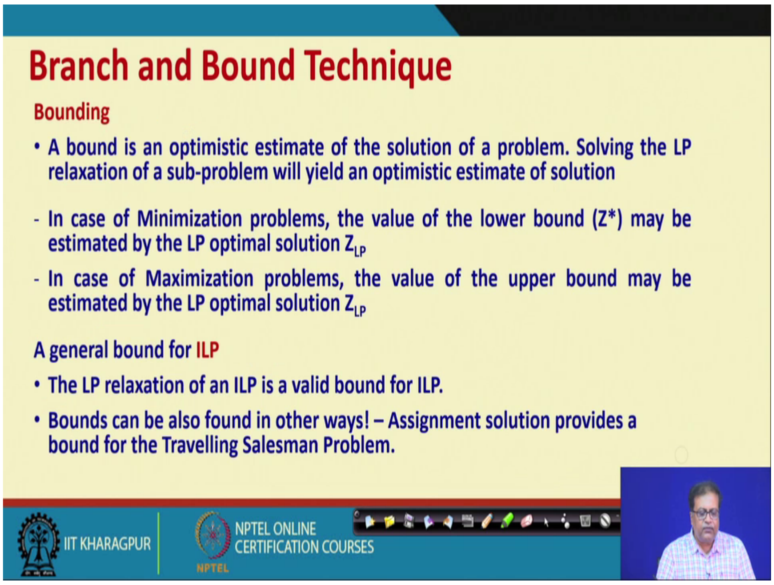
So, you see what we have done, the original problem we have branched into three problems, and for each problem we are finding the bounds alright. If A does J 2, then you know these two lines will be gone. So, let us write once again 10, 9, 8; 9 8, 7; 12 10 8. So, J 1, J 2, J 3, A, B, C, so A does J 2, so this is gone, so A will be 9. Now, which is the lowest again these are lowest, 7 and 8. So, what is the bound bound is 24 24, but it is infeasible again. But if A does J 3, so again let us see 10, 9, 8; 9, 8, 7; 12, 10, 8, A does J 3. So, then you see them again the lowest is 18 here, and this 8 here, so this is 26 infeasible.

So, we are getting all infeasible solution, but then you see A 2, J 2 looks promising, because this is 24 infeasible. Then you know if we go further then again now B has to do either J 1 or J 2. So, look here then we find B to J 1 and B to J 2. So, we did it further right. So, if now B does J 1 look, so this is also gone, so B does J 1. Then C has got only one choice, C has to do J 3 only. So, you see that time is 26 infeasible feasible, it is a feasible solution. Because A 2 you know this A to J 2, B to J 1, and C to J 3 is a feasible solution. But then if B does J 3, then C has to do J 1 then that time is 27 again feasible.

So, you see now if we look here, something interesting has come up that you know, B to J 1 26 feasible; B to J 2 27 feasible; A to J 3 26 infeasible; and A to J 1 25 infeasible right. Now, which is the lowest, lowest is 25. So, we have to explore A to J 1 further. Supposing arbitrarily I am saying now, if we explore this further, you know we might find a value which is higher than 26. Supposing arbitrarily I am not doing it, supposing I find 30 here, and 31 here suppose. Then you see out of 30, 31, 26, 27 and 26 infeasible. This is the 26 feasible is the lowest also, and feasible also. That means, the solution obtained in these branch is the lowest amongst all the different branch values, and at the time feasible.

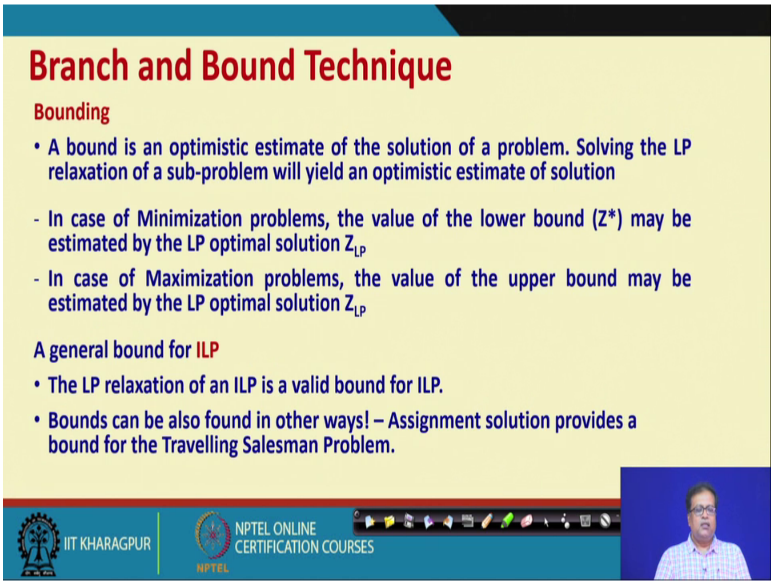
So, when that happens you know we find an optimal solution right. So, this is how it goes that we divide the problem into different subsystems, and those subsystems should be mutually exclusive and collectively exhaustive, is it alright. And in each we find a bind we found a bound, it could be lower bound for minimization and upper bound for maximization. Once we found the bounds then you know keep on branching till you know you get to a point where a particular branch is fathomed, that means we cannot go further into that branch anymore. So, when that happens; you know we have found the best possible solution by observing all the N nodes, and finding the best possible bound values. As we see more and more examples our concept will be clear.

(Refer Slide Time: 25:04)



So, let us look at a little more theory and let us see some of the things. So, these are three basic steps. So, first one branching, bounding, and fathoming right. So, what we did each problem we branched and then we found bound for each branch. And accordingly each branch we should know where to stop that is called fathoming. Branching choose one of the variable whose value will be fixed to create n new sub-problems. So, you see the problem that we took we fixed that A to J 1 or A to J 2 or A to J 3. So, when you fix then you created certain sub-problems which are obviously, mutually exclusive and collectively. If you add all of them, then they should make the total problem. So, that is how we do the branching, that is the first step.

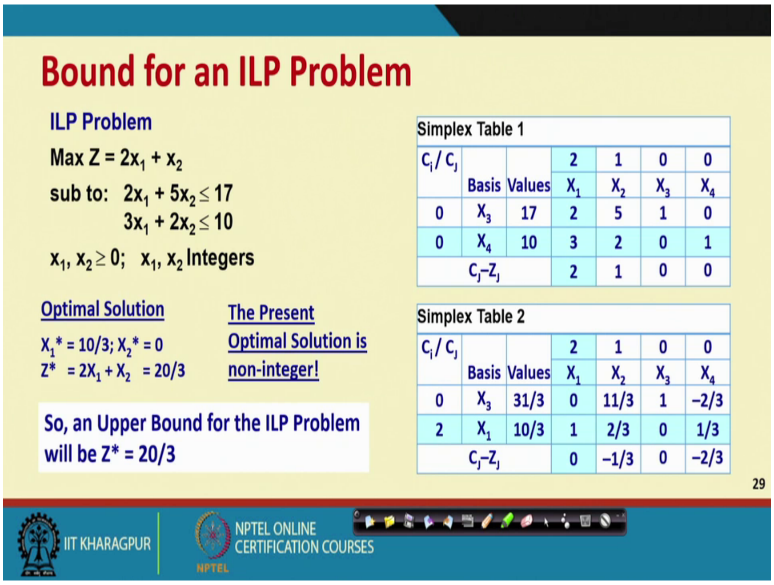
(Refer Slide Time: 26:02)



Now, after branching next stuff is bounding. Now, I have shown you one type of finding binding bound values, but there could be other methods. So, bound is an optimistic estimate of the solution of a problem, solving the LP relaxation of a sub-problem will yield an optimistic estimate of solution, so, you see if it is an integer problem, and we can solve an LP then LP can give a bound value. It is not necessary that all the time you have to solve an LP only like the example, I gave you can find bound by certain other methods also, but then typically if we are solving a non-linear sorry non integer linear programming problem, and we have a non-integer problem solution.

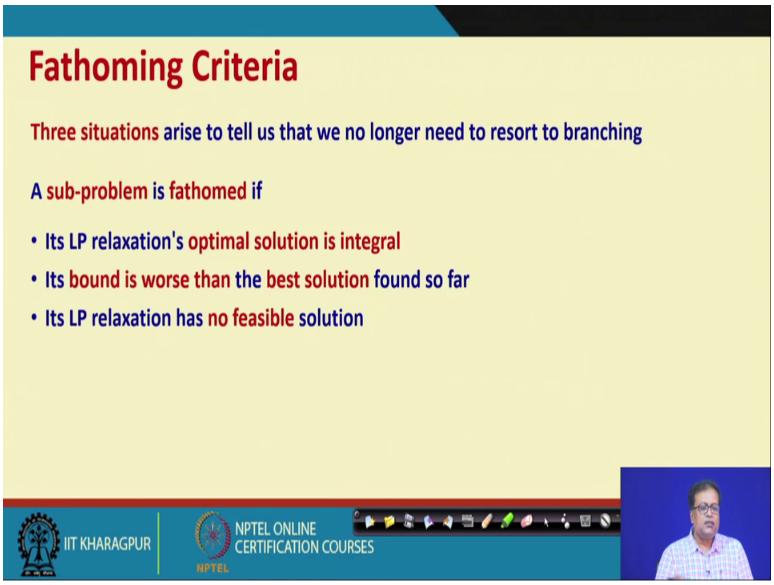
Then the LP solution can give us a lower bound for minimization, and an upper bound for maximization right. So, LP relaxation of an ILP is a valid bound, and bounds can also be found in other ways like, assignment solution provides a bound for the travelling salesman problem, later on we shall see. If you are trying to find out travelling salesman problems solution, then we can solve an assignment problem, and the solution; we find then we use it as a bound for the travelling salesman problem.

(Refer Slide Time: 27:28)



Like these ILP that we solved in our previous lecture right. So, this problem we have solved by simplex, and then we found the optimal solution was X 1 star equal to 10 by 3, X 2 star equal to 0, and Z star was 2 X 1 plus X 2 equal to 20 by 3 the present optimal solution is non integer right. So, an upper bound for the ILP problem will be 20 by 3 right. So, the LP solution would actually be used as a bound all right, so that is how it is.

(Refer Slide Time: 28:10)



And when it comes to fathoming three situations may arise, when we no longer need to resort to branching. What are those three solutions? A sub-problem is fathomed if it is LP relaxations optimal solution is integral that means we find an optimal solution and we find it is an integral solution. It is bound is worse when the best solution found so far so you see something becomes fathomed that we find a bound. But then there are some other bound at a different problem which is providing a better value, so lower than the lower bounds lowest of the lower bounds, or elsewhere.

So, we found a lower bound here, which is higher than a lower bound elsewhere, so, that means this one is fathomed. And the LP relaxation has no feasible solution by LP relaxation we found an infeasible solution that means, also we can stop there right, so by making use of the bounds the branching and the fathoming right.

We can solve the branch and bound algorithm we can make use of, and in our next class we shall take proper you know examples of how to solve branch and bound. You know use branch and bound techniques to solve general purpose problem is it alright. And specifically further we shall spend more time in solving travelling salesman problem by branch and bound technique right in further classes right. So, we stop here.

Thank you very much.