Advanced Operations Research

Prof. G. Srinivasan

Department of Management Studies

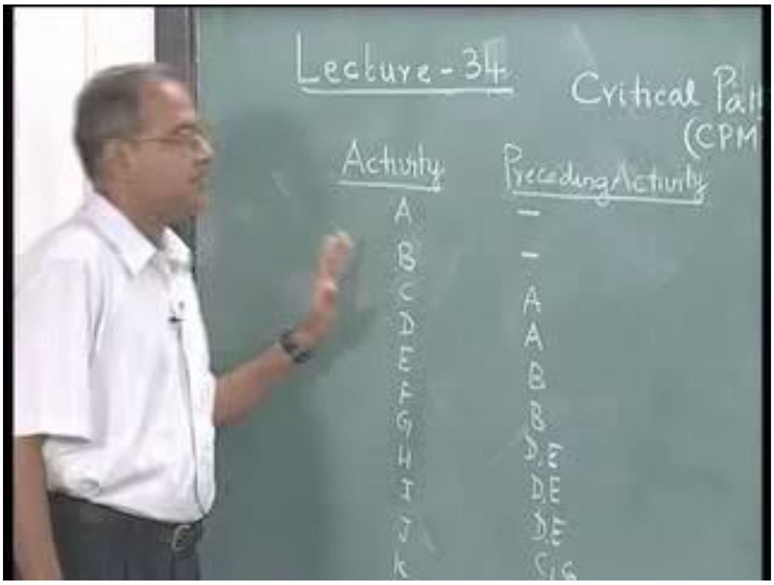
Indian Institute of Technology, Madras

Lecture - 34

Critical Path Method

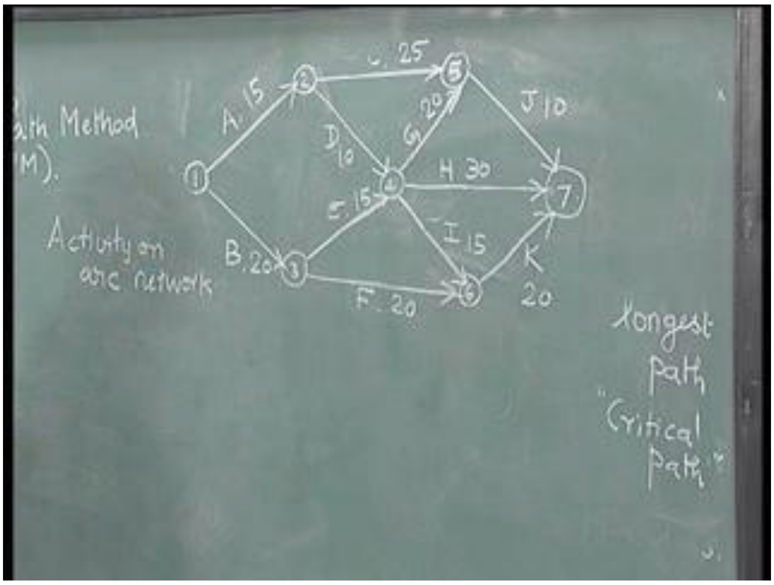
In this lecture, we continue our discussion on the critical path method. In the previous lecture, we introduced the critical path problem on the network by considering this example.

(Refer Slide Time: 00:28 min)



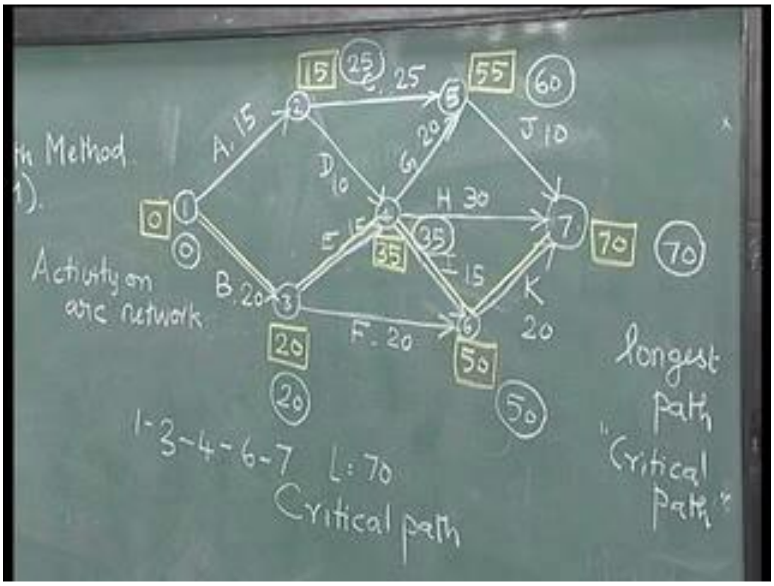
The network or the project network comprises of a set of activities and there are some precedence relationships among these activities. Based on these precedence relationships, we were able to draw this network and the procedure to draw was explained in the previous lecture.

(Refer Slide Time: 00:50 min)



After drawing this network and after writing down the durations associated with each activity, we would now like to find out when earliest this project can be completed. Now, to do that, we followed a labeling procedure, which was also described in the previous lecture, but we go through that procedure once again to understand a few more aspects.

(Refer Slide Time: 01:23 min)



Now, let us assume, we start here with time equal to zero. This takes duration of 15, so we reach this at 15. This takes duration of 20, so we reach this at 20. When we come here, we reach at 15 plus 10, that is, 25; 20 plus 15, that is, 35. This node 4 represents the point at which both D and E are completed and therefore, the maximum of the values would be the label for the node and we would get 35 here. When we come to 5, this indicates the completion of both C and G. So, it is 15 plus 25, 40; 35 plus 20, that is, 55; the maximum of that is 55. Similarly, 35 plus 15 is 50, 20 plus 20 is 40, the maximum is 50. For 7, 55 plus 10 is 65, 35 plus 30 is 65, 50 plus 20 is 70. So, based on this labeling procedure, we say that the earliest, it takes to complete this project, to complete all the activities along with the given precedence and the duration is 70 units of time.

Now, the labeling procedure was very similar to the Dijkstra's algorithm for the shortest path problem, except that the node labeled here was the maximum of the times as against the minimum of the times that we considered when we solved the shortest path problem. This labeling is a forward pass of the labeling procedure; we also go through a backward pass of the labeling procedure. So, we start with the same 70 here, but we use a different symbol; a circle for example and then, go through the backward pass. So, this is 70 minus 20, which will be 50; 70 minus 10, which will be 60. So, when it comes to 4, this will become 70 minus 30, which is 40, 50 minus 15, which is 35, and 55 minus 20, which is 35. So, we choose the minimum of them and we get 35 for this.

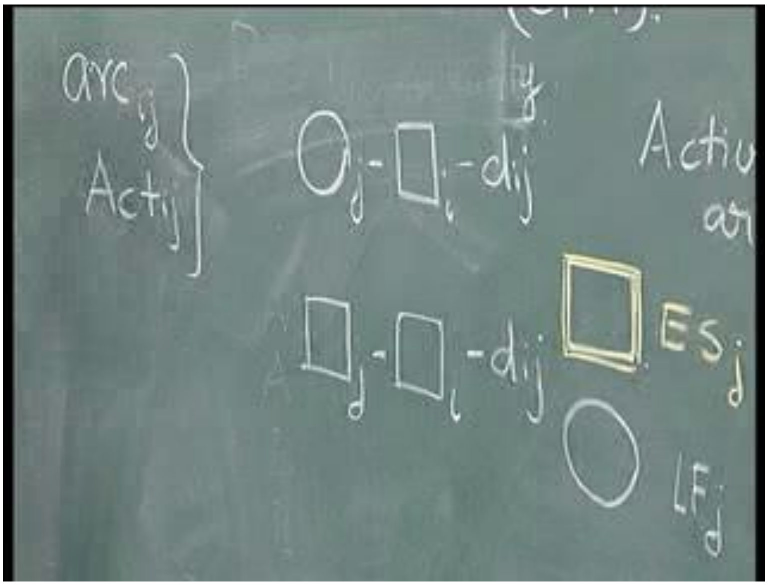
For this, it is 55 minus 25, which is 30; 35 minus 10, which is 25. So, we take the minimum of this. For this, it is 35 minus 15, which is 20; 50 minus 20, which is 30. The minimum was 20 and for this, 25 minus 15, that is, 10; 20 minus 20 is zero. Hence, the minimum is zero. So, we have completed the backward pass also associated with this and when we started the backward pass with this number 70 and we proceeded backwards, we were able to get zero, which was this. Now, when we look at both the labels that we have drawn the forward pass label, which is shown inside a square of a different color and the backward pass label, which is shown inside a circle and of a different color, we realize that some nodes have the same values of the labels like for example, node 1, node 3, whereas some nodes have different values for the labels like node 2 and node 5.

We also observe that if we have a path, we are able to get a path that starts from 1 and ends with 7 in this network, such that all the vertices in the path have equal values of the forward pass label and the backward pass label. Such a path, if you see carefully, is the path 1 to 3, 3 to 4, 4 to 6 and 6 to 7. So, this path, which is 1 to 2, 1 to 3, 3 to 4, 4 to 6, 6 to 7 with length equal to 70, which is 20 plus 15, that is, 35 plus 15 is 50 plus 20, that is, 70, is the longest path and it is also called the critical path of this network. So, the critical path is the path that goes from the first node to the last node, such that all the vertices that are there in the critical path have both labels equal.

Now, the other paths in this network, for example, 1 to 2, 2 to 5, 5 to 7 or 1 to 2, 2 to 4, 4 to 7, other paths that are there will have length less than or equal to that of the critical path. If they are equal, then they become alternate critical path; but, if they are strictly less, those paths are not critical. So, all the paths from 1 to 7 are either critical or non-critical and all non-critical paths will have length less than that of the critical path. For example, 1 to 2, 2 to 5, 5 to 7 would now have 15 plus 25, that is 40 plus 10 is 50, which is less than this 70, which is the critical path. So, we also understand that the critical path is extremely important in this network. For this network, to be completed at time equal to 70, then we have to do these activities sequentially and some of these activities, the non-critical activities, are those that are not in the critical path can have some more extra time and they can be fitted in such that the total project duration stays at 70. We also understand that if there is a delay in these critical paths, by which it takes more than 20 units of time, say, 21, then the length will become 71 and the time to complete the project will increase. So, to that extent, the critical path is extremely important and this labeling algorithm would help us to get the critical path.

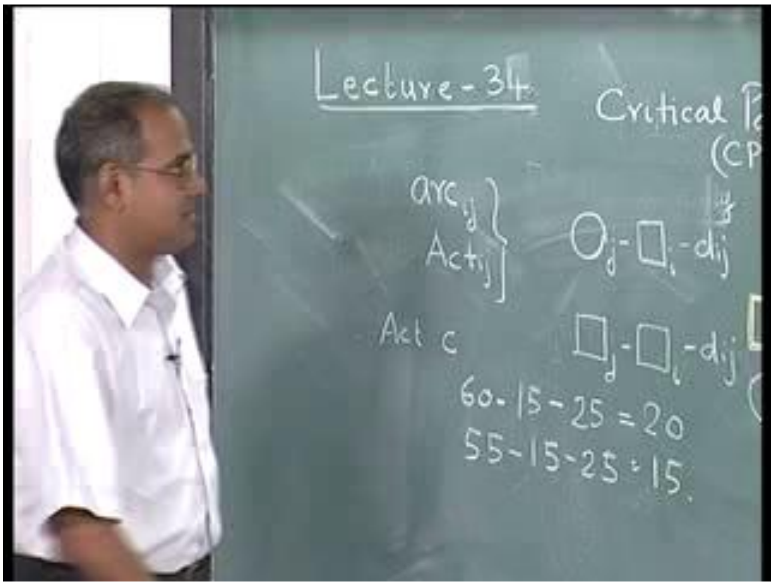
We also need to look at one more thing, what does this thing indicate? Here there is a difference, there is a 15 and a 25; here, there is a 55 and a 60 and so on. So, there are nodes, where the forward pass label and the backward pass labels are different. Do they convey something? What do they convey?

(Refer Slide Time: 08:52 min)



We try and define something like this here. If we take the forward pass, the box in the forward pass represents what is called the early start of that particular activity and whatever is shown in the backward pass represents the finish or late finish of that activity, j. We start defining some terms here. When we start defining a couple of terms, let us find out, for example, for a particular arc_{ij} or for an $activity_{ij,}$ let us define the backward pass quantity j minus forward pass quantity i minus dij. Let us also define the forward pass quantity j minus forward pass quantity i minus dij. For example, if we take this activity C, activity C is 2 to 5; so the backward pass label for j is the label corresponding to 5, which is 60. The forward pass label is the label corresponding to this, which is 15 minus the duration, which is 25. So, this will be 60 minus 15 is 45 minus 25 is 20. Now, the second one is 55 minus 15 minus 25, which is 15. What do these two numbers tell us?

(Refer Slide Time: 11:03 min)

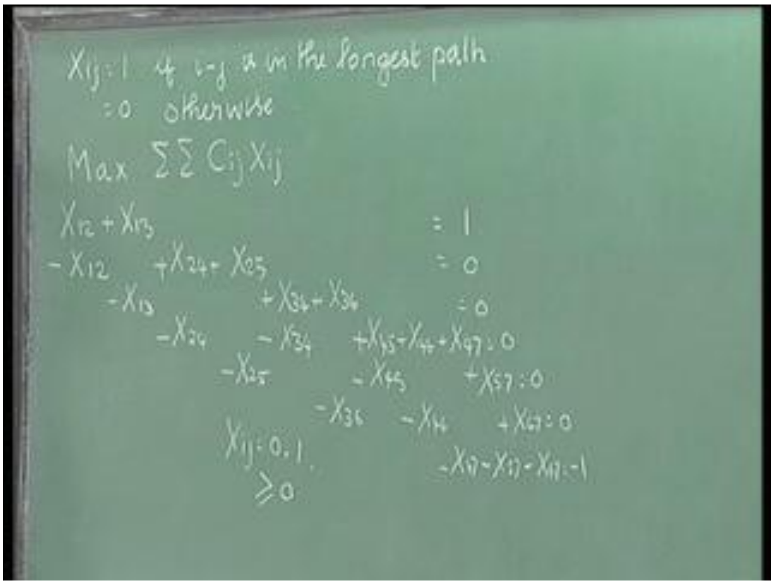


What it tells us is, if we look at this activity C, the earliest this can start is 15, the latest this can finish is 60, it’s duration is 25. So, it cannot start before 15 and it should end at 60, which means there is a 45 duration period, in which this activity which takes 25 units has to be completed. This activity cannot start before 15 and if it is scheduled in such a way that it ends after 60, then the critical path will be affected; but within this buffer time of 45 time units, 25 is the duration that is required for this. So, there is an excess buffer of 20, that is, available for this. So, this 20 is called the ‘total float’ associated with that activity. So, for this activity, there is a total float of 20. The earliest start is 15, and the earliest finish is 40. So, the latest finish is 60. Therefore, there is a float associated with this; that float is 20 units of duration that can be consumed. For example, if we look at this, the earliest it can start is 55, the earliest it can finish is 65. So, there is a float of 5 associated with this. The next thing is called a free float. The earliest it can start is 15, the earliest it should finish is 55. So, 15 plus 25 is 40 plus 15, which is 55. This is called free float.

Both these total float and free float in some sense, tell us the excess time that is available. By the way, these numbers are written down, the free float is always less than or equal to the total float. While the total float tries to tell us the excess buffer or total buffer, which can be used, which is 20, somewhere 15 can be used comfortably for this and if it stretches beyond 15, it will still try and affect something somewhere. So, both these floats give us a picture of some extra time that is available. The total float represents some kind of an extra time that is available across the entire path, containing these arcs and free floats are specific to the arcs. Nevertheless, without going very deep into the meaning of these two, we can understand that both these floats in a certain manner represent the extra time that is available, which has to be used and any poor scheduling, by which this exceeds the floats that are available for that activity, then this activity will become critical. So, this is how, the longest path on the network is computed. The critical path is the longest path in the network and the critical path computation is quite similar to the Dijkstra's label, except that instead of labeling the minimum, we label the maximum.

We need to show and understand why this kind of labeling is optimal for this problem, for which we write a linear programming formulation and try to understand the primal dual relationships and show that indeed this forward pass would give us the longest path on this network. So, to that extent, the CPM that we are looking at in this lecture series can be seen as an application of linear programming technique to solve real life project management problems. So, let us first, write down the primal and then, the dual and try to understand how this labeling is optimal.

(Refer Slide Time: 15:28 min)



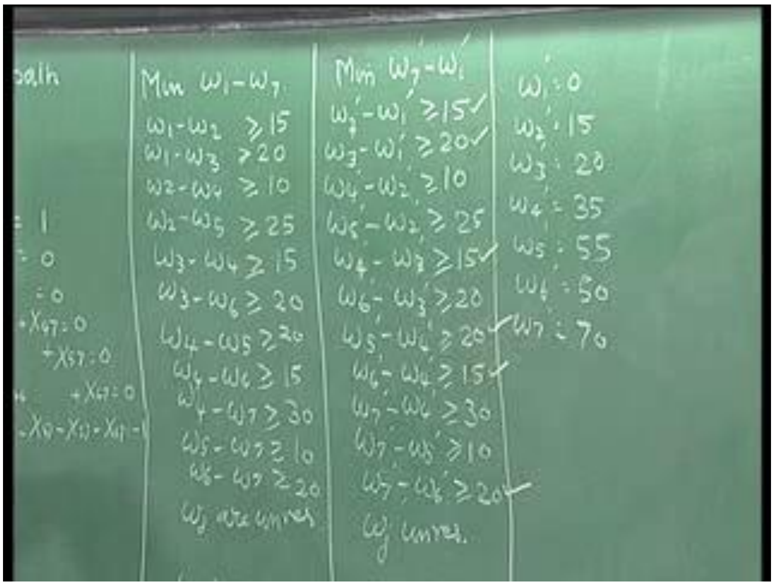
As we are always interested in finding the longest path in this network, the X_{ij} equal to 1, if ij is in the longest path and ij equal to zero, otherwise. The objective function will be to maximize sigma C_{ij} X_{ij}, where C_{ij} stands for these durations. This is the origin, so the longest path should start from here. Therefore, we have the usual X_{12} plus X_{13}

equal to 1. As far as node 2 is concerned, we have minus X_{12} plus X_{24} plus X_{25} equal to zero. 2 being an intermediate node, if the longest path goes through this, then 1 comes here; one of this has to be one. If it does not go through, everything will be zero. So, you have simply a flow balancing or conservation equation for every intermediate node. For 3, we would have minus X_{13} plus X_{34} plus X_{36} equal to zero. For 4, we have minus X_{24} minus X_{34} plus X_{45} plus X_{46} plus X_{47} equal to zero. For 5, we have minus X_{25} minus X_{45} plus X_{57} equal to zero. For 6, we have minus X_{36} minus X_{46} plus X_{67} equal to zero. For 7, which is the end node, we have minus X_{57} minus X_{47} minus X_{67} is equal to minus 1, because the longest path should end at 7. So, we get minus X_{47} minus X_{57} minus X_{67} is equal to minus one. So, these are the equations for this. Then, we have X_{ij} equal to 0, 1.

This formulation is very similar to the shortest path formulation, except that the shortest path will have a minimized C_{ij} X_{ij}. Here, we have a maximized C_{ij} $X_{ij.}$ We have already seen that the shortest path problem is unimodular and unimodularity has only to do with the constraints and it has nothing to do with the objective function. So, the longest path problem the way it is represented on this kind of a network, where you have arcs from i to j, j greater than i is unimodular. Therefore, we can treat this X_{ij} to be greater than or equal to zero and solve the resultant linear programming problem, which would give us the optimal solution to this.

Now, having understood that this is a linear programming problem, we will now go back and write the dual associated with this problem.

(Refer Slide Time: 19:03 min)



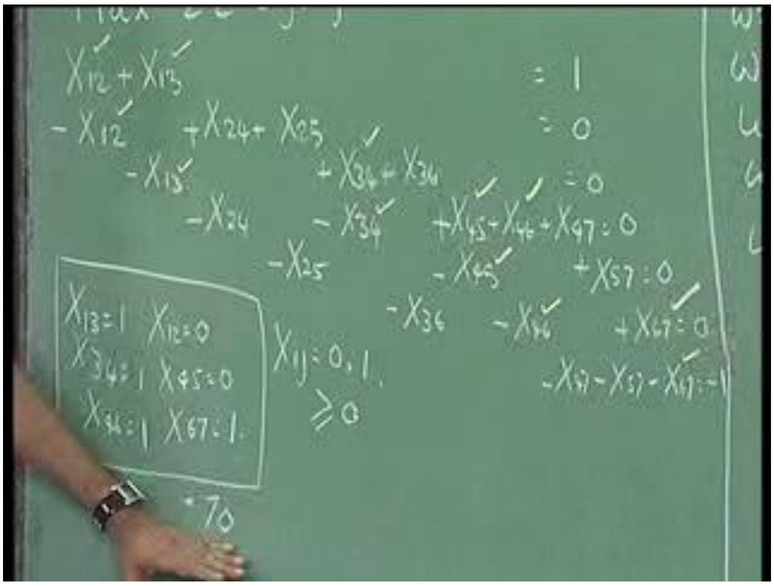
So, we define dual variables, w_1, w_2, w_3, w_4, w_5, w_6, w_7. The primal is a maximization problem, so the dual will be a minimization problem. Minimize w_1 minus w_7 that we have here, subject to every ij will be in the ith and the jth constraint. So, subject to w_1 minus w_2 it is a maximization problem with all greater than or equal to constraint. So, you will get a minimization problem with all greater than or equal to variable. So, w_1 minus w_2 is greater than or equal to 15. w_1 minus w_3 is greater than or equal to 20. w_2 minus w_4 is greater than or equal to 10. w_2 minus w_5 is greater than or equal to 25. w_3 minus w_4 is greater than or equal to 15. w_3 minus w_6 is greater than or equal to 20. w_4 minus w_5 is greater than or equal to 20. w_4 minus w_6 is greater than or equal to 15. w_4 minus w_7 is greater than or equal to 30. w_5 minus w_7 is greater than or equal to 10. w_6 minus w_7 is greater than or equal to 20. Importantly, all the w_j's are unrestricted in sign. The unrestricted comes because of these equations.

Now, we make another interesting change here, where, because these wjs are unrestricted in sign, we are now going to replace each w_j by say minus w_j dash and because, w_j is unrestricted minus w_j dash will also be unrestricted in sign. So, this will now be rewritten as minimize w_7 dash minus w_1 dash, subject to w_2 dash minus w_1 dash greater than or equal to 15; w_3 dash minus w_1 dash is greater than or equal to 20, w_4 dash minus w_2 dash greater than or equal to 10. w_5 dash minus w_2 dash greater than or equal to 25. This is w_4 dash minus w_3 dash is greater than or equal to 15. w_6 dash minus w_3 dash greater than or equal to 20. w_5 dash minus w_4 dash greater than or equal to 20. w_6 dash minus w_4 dash greater than or equal to 15. w_7 dash minus w_4 dash greater than or equal to 30. w_7 dash minus w_5 dash greater than or equal to 10. Finally, w_7 dash minus w_6 dash greater than or equal to 20. w_j dash unrestricted in sign.

Please remember that because we have replaced w_j by w_j dash, this will become minus w_1 dash plus w_2 is greater than or equal to 15. Remember, we are not multiplying this equation by minus 1; the right hand side will remain the same, we are only replacing the variable w_j by minus w_j dash, so that this becomes minus w_1 dash plus w_2 dash. It will still be greater than or equal to 15. So, we now have this as the dual, which comes from this and now, we can start with w_1 dash equal to zero. When w_1 dash is equal to zero, automatically w_2 dash is greater than or equal to 15 and we want to minimize somewhere, w_7 dash minus w_1 dash. So, w_2 dash will become 15. w_3 dash will become 20. Now, from this, w_4 dash will become 15 plus 10, that is, 25 and w_4 dash is 20 plus 15, that is, 35. So, w_4 dash is greater than or equal to 25. w_4 dash is greater than or equal to 35. So, w_4 dash will become equal to 35.

Now, for w_5 dash, you get 25 plus 15, so, greater than or equal to 40. This is greater than or equal to 55; so w_5 dash will become 55. For w_6 dash, you get 20 plus 20, that is, 40 and then, 35 plus 15, 50. So, w_6 dash will become 50 and w_7 dash will become this is 30 plus 35, 65. This is 55 plus 10, 65 plus 20, that is, 70. So, starting with w_1 dash equal to zero, we can go through this set and by inspection, we can get a solution here with 70. Now, we go back and find out, apply complimentary slackness and see which are the ones that are satisfied as an equation. So, this is satisfied as an equation and this is also satisfied as an equation. Then, w_4 dash comes from 15 plus 20 and this is an equation. w_5 dash comes from 35 plus 20, 55. So, this is an equation. w_6 dash comes from 35 plus 50 and w_7 dash 50 plus 20, 70. Now, from these, we have a fixed w_1 equal to zero. So, six of them give us the equations. Now, we apply complimentary slackness. Wherever it is satisfied as an equation the corresponding variable is a basic variable. So X_{12} is a basic variable, X_{13} is a basic variable, X_{34} is a basic variable, X_{45} is a basic variable, X_{46} is a basic variable and X_{67} is a basic variable. So, these are the six basic variables that we have and then, we realize that the solution will be X_{12} plus X_{13} equal to 1.

(Refer Slide Time: 26:53 min)



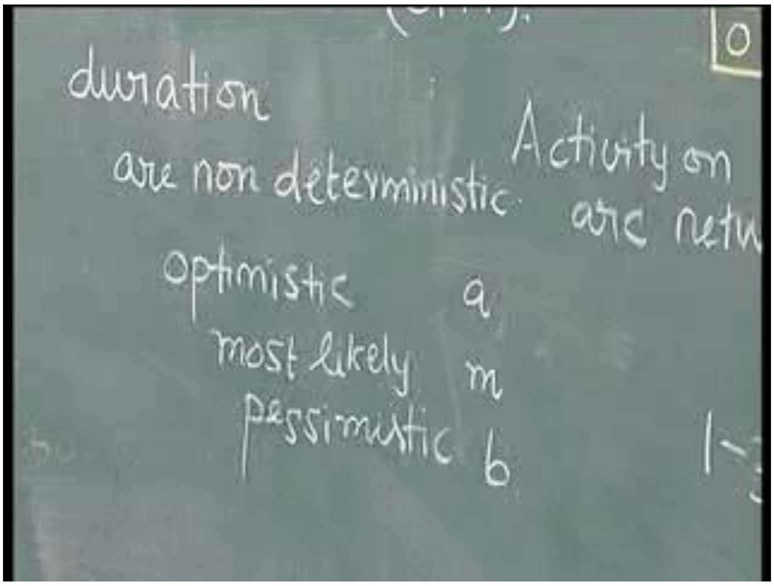
So, now we treat X_{13} equal to 1, X_{12} equal to 0. So, this is satisfied. Now, from this, X_{12} is basic; so zero equal to zero. From this constraint, it is satisfied, these are non-basic. This is a basic variable, but these are non-basic at zero. This is a basic variable at zero; so, you get zero equal to zero. As far as this constraint is concerned, X_{13} is equal to 1. So, this contributes a -1; therefore, X_{34} will be equal to plus 1. So, this is satisfied. As far as this is concerned, here X_{34} has 1; so, this is contributing a -1, then, we have X_{45} which is zero, X_{46} is 1. So X_{45} is zero, X_{46} equal to 1, so that minus one plus zero plus one is zero. X_{45} is a basic variable with zero. Now, this is a basic variable, so we get zero equal to zero. These two are non-basic. Now, X_{46} equal to 1 would give us X_{67} equal to 1, because you have a minus sign here. So, you get X_{67} equal to 1. Now, as far as this is concerned, X_{67} is 1, so minus 1 equal to minus 1. So, we have a set of six basic variables, which correspond to these six equations coming from this solution. These six are satisfied as equations. So, these six are the basic variables and with this set of basic variables, we are able to get a basic feasible solution, which satisfies the primal. We could get degenerate, when we are in situations like here, we get zero equal to zero and somewhere here also we get zero equal to zero; otherwise, it is a degenerate basic feasible solution, which is obtained by applying complimentary slackness from here. The objective function value associated with this is 1, 3; 3, 4; 4, 6 and 6, 7, which would give us 70. So, we have a dual feasible solution, we have applied complimentary slackness, we have got a corresponding primal feasible solution with the same value of the objective function. Therefore, it is optimal to both the primal and the dual respectively.

Now, this also shows that whatever we computed a 0, 15, 20, 35, 55, 50, 70 are the labels that we have here; 0, 15, 20, 35, 55, 50, and 70. The algorithm that we developed here, which was actually the modification, where we updated the largest value, instead of the smallest is optimal to this particular problem. Wherever we write the primal and the dual and when we make this change and then, we can start with w_1 equal to zero, we will get the forward pass labels. On the other hand, if we use this and if we started with w_7 equal to zero or 70, as the case maybe, we could come back and get the backward pass labels. In fact, if you remember very carefully, we did a very similar analysis for the shortest path problem. We also mentioned that we could write the dual this way and then, we mentioned that if we write the dual this way, we get the forward pass, whereas if we write the dual this way and move from w_7 equal to 0 or 70, we would get the backward pass labels. So, this is the relationship between the primal and dual of the longest path problem.

The critical path problem can always be formulated as the longest path and a simple forward pass and a backward pass of the Dijkstra's algorithm suitably modified for the longest path would give optimal under these circumstances. The circumstances are very important; the network is made up of arcs such that arcs go from i to j, j greater than i. If we are able to do that, then this problem can be formulated as a longest path problem and this particular longest path problem can be solved optimally using a polynomially bounded algorithm, whose optimality can be proved by the primal dual relationships that we have shown here. So, this is how, we solve the longest path problem.

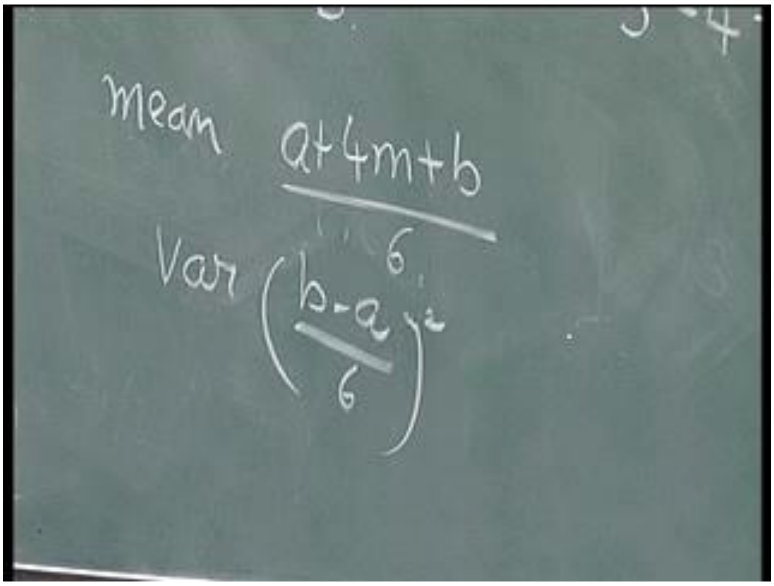
There are a couple of other things that we might see in the critical path problem. One of the things that is very common, which we will also do for the sake of completion is trying to analyze the critical path particularly, when the durations are non deterministic.

(Refer Slide Time: 32:02 min)



When we did the critical path, we have assumed that these durations are known and deterministic. There could be situations, where these cannot be estimated accurately. So, they will follow certain distributions. So, whenever they cannot be estimated accurately, then it is customary to look at distributions and instead of fitting a distribution, it is also customary to fit three estimates of the demand, which is called the optimistic estimate, the most likely estimate and the pessimistic estimate. Now, these are usually called as a, m, and b respectively for every one of these activities. It is obvious that m is greater than or equal to a and b is greater than or equal to m, because the pessimistic estimate is higher than that.

(Refer Slide Time: 33:09 min)



These kind of estimates are drawn for each of these activities and then, the mean time is calculated assuming a beta distribution as a plus 4m plus b by 6 and the variance is given by b minus a by 6 the whole square. Based on this distribution, the expected value and the variance are computed. Each of these will have an expected value and the variance. With these expected values, one can substitute these with the expected values and then, do one pass of the CPM or the critical path method to get the estimated or expected longest path and along with that, the duration will be the expected duration. Since variance is additive, we could add the variances and say that this is the expected critical path with a certain expected variance. So, the analysis shifts from a purely deterministic analysis to more of a probabilistic analysis. Such a thing is called PERT, which is called 'Program Evaluation and Review Technique'.

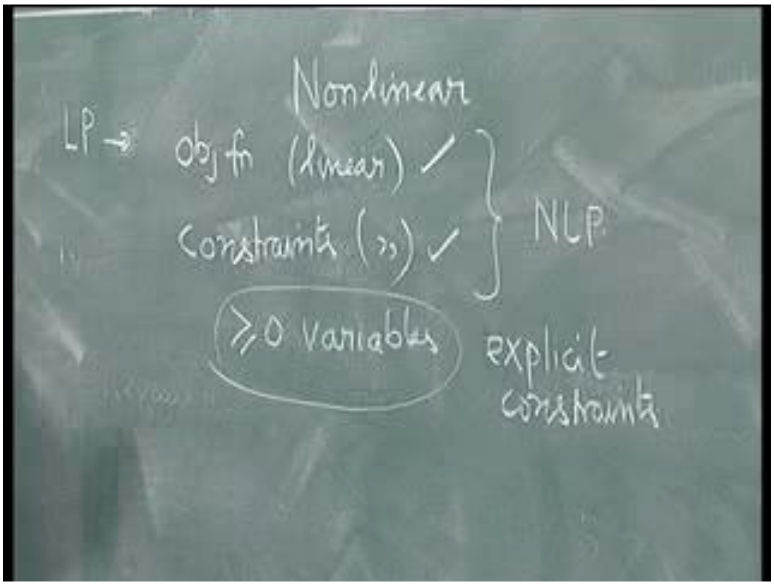
This technique represents a probabilistic analysis. One can also do certain simulations of this network to try and find out the expected longest paths; but, those things are slightly beyond the scope of this lecture series. So, we are not proceeding in that direction. We only wish to inform that these problems are close to the OR problems that we looked at and while the CPM, the critical path method can be treated or seen as an OR application of finding the longest path on a particular type of a network, PERT offers a probabilistic analysis of the network. We will see one more aspect of the critical path problem, before we complete our discussion on it.

When we come back to the critical path method, we characterized each activity by its duration. There will be situations, where certain resources are required to carry out these activities. For example, if this is from a construction project network, we need material, we need people, etc. Many times, we will not have an unlimited number of these resources. So, we will be constrained by resources and that leads to what are called 'resource constrained project scheduling problem'. We could have single resource, we could have multiple resource, and so on. Just to give an example, if we are looking at material as a resource or if you are looking at labor as a resource, people as a resource, now, this will become for activity 'A' we could say, 15 and 2, which means 2 people are required to carry out this 15.

For example if we say B, it would mean 20 and 3, which means 3 people are required to carry out B. Then, if we have a restriction that we have only 4 people available, then we cannot do A and B parallely, because to do them parallely we require 5 people. As we cannot do them parallely, the longest path in the network will increase and the earliest time that we can complete a project will also increase. So, the resource constrained project scheduling problem will now become a very similar formulation of the CPM problem plus resource constraints. In fact, the actual formulation of the resource constrained project scheduling problem will be slightly different from what we show here; but in principle, it will be the longest path problem with resource constraints. So, the moment we add the resource constraints into the problem, the problem will lose its unimodular structure and the problem cannot be solved as a linear programming problem; the problem will become an integer programming problem. It becomes a very hard problem and it becomes an application of integer programming, a topic that we have covered in this lecture series.

There are popular branch and bound algorithms as well as heuristic algorithms to solve the hard problem, which is the resource constrained project scheduling problem. However, the normal critical path problem, which does not include the resource constrained or to state that this assumes that an infinite amount of resource is available so that resource is not a constraint, then, that problem becomes the longest path on this network. It can be solved by an adaptation of the labeling procedure that we applied to solve the shortest path problem. So, this brings us to the end of our discussion on the critical path method. Then, we go to the next topic, which is quadratic programming.

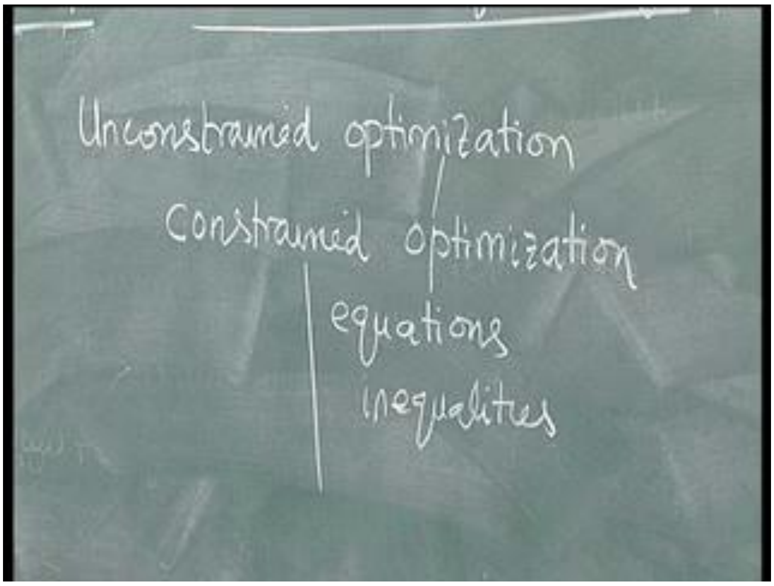
(Refer Slide Time: 38:49 min)



Quadratic programming is a part of nonlinear programming. In this lecture series, we will be restricting ourselves to understand only one algorithm to solve the quadratic programming problem. So, before we introduce a quadratic programming problem, we have to introduce a nonlinear programming problem and then show where the quadratic programming comes under the general umbrella of nonlinear programming or nonlinear optimization. We are very comfortable, because we have addressed the linear programming problem extensively in this lecture series.

Linear programming problem has an objective function, which is linear. It has a set of constraints and all these are linear. It has an explicit restriction of greater than or equal to for the variables, which is called the non-negativity restriction and all these three will constitute a linear programming problem. Now, what is a nonlinear programming problem? If there is a non-linearity in the objective functions or if there is a non-linearity in the constraints, then the problem becomes an NLP or nonlinear programming problem. One of the important things in nonlinear programming problems are that, we do not have the explicit mention of variables to be greater than or equal to zero. The variables can even take negative values at the optimum or at the solution. If we wish to say that the variables have to be greater than or equal to zero, these become explicit constraints in the NLP, instead of being implicitly given, as they were given in the linear programming problems. When we have nonlinear functions that we wish to optimize, then we have these classifications.

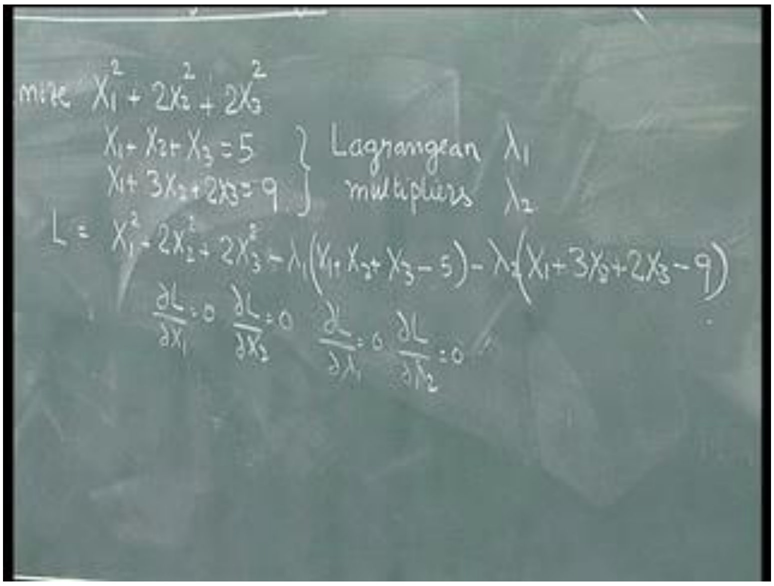
(Refer Slide Time: 41:27min)



So, we have unconstrained optimization problems and then, we have constrained optimization problems. These constraints can be equations or these can be inequalities. We also have single variable optimization problems and multiple variable optimization problems. Now, when we solve these types of optimization problems, we have necessary conditions and we have sufficient conditions for optimality. We are quite aware that the first derivative is equal to zero from calculus of a single variable optimization. First derivative equal to zero would give the optimum and the second derivative at the optimum will qualify whether it is a maximum or a minimum. If we have an unconstrained optimization problem, then we simply have if it is a single variable df by dx equal to zero gives a maximum or a minimum; the second derivative would tell us whether it is a maximum or whether it is a minimum.

If we have unconstrained optimization problem with multiple variables, then dow f by dow x equal to zero would give us the maximum or the minimum. Then, the dow square f by dow x square will have to be computed as a matrix. We need to do dow square f by dow x_1 square dow square f by dow x_2 square, and so on. So, there are these sufficient conditions for sufficiency for whether it is a minimum or whether it is a maximum. Here, because we are restricting ourselves to quadratic programming, we are not going to concentrate that much on the sufficiency condition. We are only going to concentrate on trying to get the maximum or the minimum, when it actually exists, which means trying to put the first derivative equal to zero and trying to solve them. We describe an unconstrained optimization problem; first, without we have seen that, then we start describing constrained optimization problems. Then, show equations and inequalities and then, present a way to solve them and then, come back to quadratic programming.

(Refer Slide Time: 44:10 min)



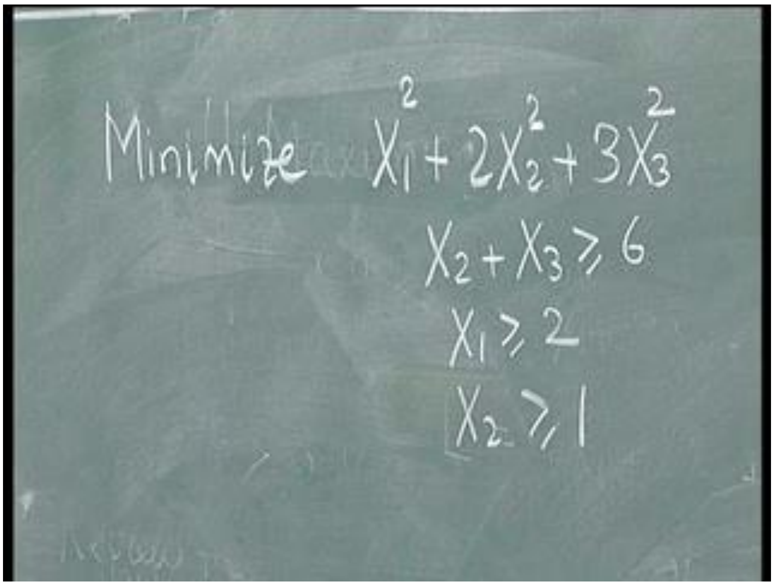
Now, let us look at a problem like this; maximize X_1 square plus $2X_2$ square plus $2X_3$ square subject to X_1 plus X_2 plus X_3 equal to 5 and X_1 plus $3X_2$ plus $2X_3$ equal to 9. In this example, we do not have an explicit mention of X being greater than or equal to zero. If we did not have this constraint, then we would have simply taken the partial derivatives equal to zero and then, we would have got the minimum or the maximum point; first derivative equal to zero gives the optimum. If it were unconstrained, then it would only be a minimum and it would not be a maximum. This is a maximization problem with constraints. Now, let us assume that we know to solve this kind of problem, when there are no constraints, that is, dow f by dow x equal to zero; first derivative equal to zero gives us the solution. But, the moment we have constraints, how do we handle that.

One of the ways of handling constraints, particularly, when the constraints are equations, is to use Lagrangean multipliers. Take this to the objective function by introducing as many Lagrangean multipliers as the number of constraints. So, in this case, we would introduce Lagrangean multipliers lambda_1 and lambda_2. So, this problem will become L, which is the Lagrangean function will be X_1 square plus $2X_2$ square plus $2X_3$ square minus lambda_1 into X_1 plus X_2 plus X_3 minus 5 minus lambda_2 into X_1 plus $3X_2$ plus $2X_3$ minus 9. We need to describe why we put a minus for this lambda_1 and lambda_2. The reason is that we always write this of the form lambda into ax minus b, where the constraint is of the form ax equal to b. If we remove this constraint, then it becomes an unconstrained problem and if we add this constraint, it becomes a constrained problem. So, any time, when we add a constraint to a maximization problem, the objective function value comes down and therefore, we subtract. We put a minus lambda_1 and then write a x minus b; so minus lambda_2 to do this.

Then, we can go back and take partial derivatives; partial derivates equal to zero which would give us the corner point. Then, we have to look at the sufficiency condition and verify that whatever we got by the partial derivatives is actually a maximum. So, we just go only up to the partial derivatives, so you could do dow L by dow X_1 equal to zero, dow L by dow X_2 equal to zero, dow L by dow lambda_1 equal to zero, dow L by dow lambda_2 equal to zero would give us the solution. In this case the whole thing is now a second degree, a quadratic expression, there is a square, square, square; lambda_1 X_1 is quadratic, and so on. So, first derivative of all these would give us linear equations. Solving these linear equations would give us the corner point associated with the optimum. Then, we need to go back and show that the corner point that we obtained is indeed a maximum. Whenever we have equations, we can use the method of Lagrangean multipliers. Introduce as many Lagrangean multipliers, take it into the objective function, then, use a dow L by dow X equal to zero dow L by dow lambda equal to zero and solving the resultant system would give us the corner point, except that, we need to know how to solve the resultant system. This whole thing being a quadratic function, all the derivatives would give us a linear equation. If this were cubic, then the resultant equation will have a quadratic term and then, we may have to resort to a suitable method to solve such a thing. Nevertheless, Lagrangean multipliers present us with a framework with which we can attempt problems of this size.

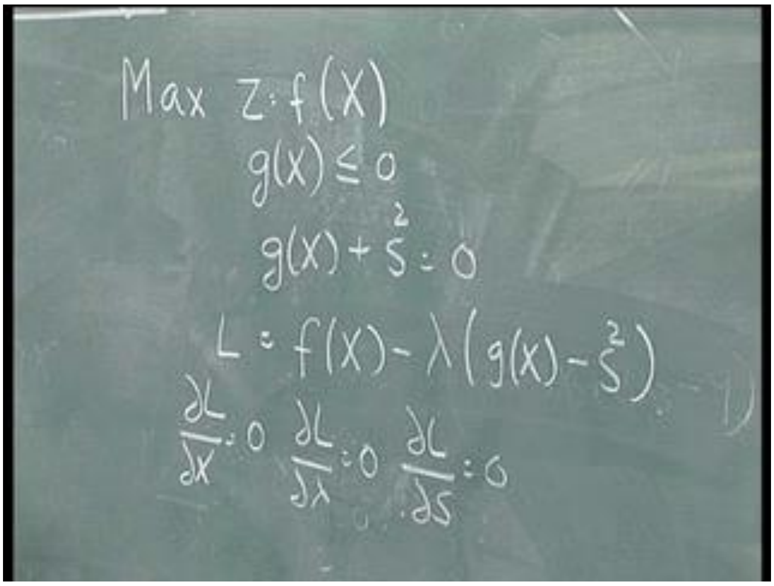
Now, we get into another situation. What happens when these constraints are inequalities, instead of equations? When these constraints are inequalities, let us take another problem.

(Refer Slide Time: 49:30 min)



So, let us consider a problem like this, which is minimize X_1 square plus $2X_2$ square plus $3X_3$ square subject to X_2 plus X_3 greater than or equal to 6. X_1 greater than or equal to 2, X_2 greater than or equal to 1. This does not have an explicit restriction that X_1, X_2 should be greater than or equal to zero. When we have this, then we cannot directly use the Lagrangean multipliers, because the Lagrangean multiplier method is meant for equations. So, whenever we have inequalities, the first thing that we have to do is to convert these inequalities into equations.

(Refer Slide Time: 50:25 min)

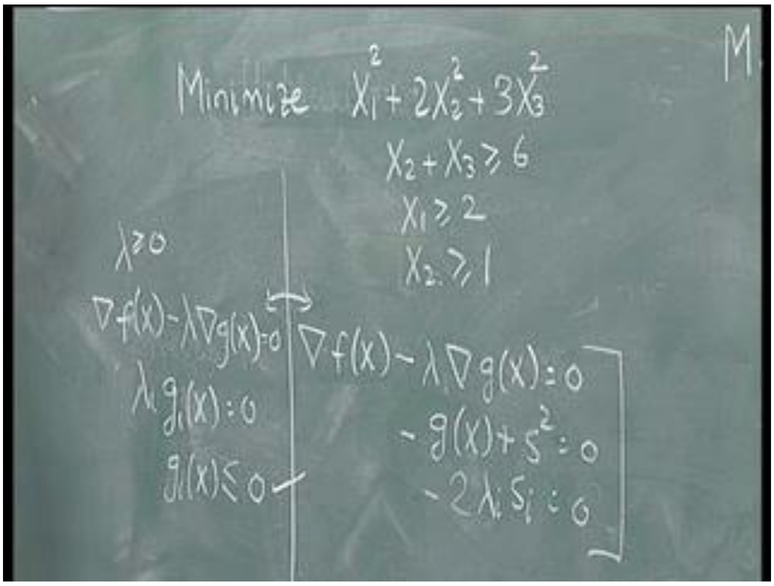


We write a general expression that we have here. In a general maximization problem, we have maximize Z is equal to f of X, where this X is variable, X_1, X_2 to X_n subject to g of X less than or equal to zero. g of X less than or equal to zero has three constraints here. It is always possible to multiply this with -1 to convert it into g of X less than or equal to zero. So, this is a general form of a constrained optimization nonlinear optimization problem, where the constraints are of the inequalities. Now, in linear programming, we used to write this as, for example, X_2 minus X_5 equal to 1 and put X_5 greater than or equal to zero. Because linear programming has this explicit restriction that every variable is greater than or equal to zero, it was easy to write this as X_2 minus X_5 equal to 1 and then, say, X_5 greater than or equal to zero.

In nonlinear programming, we do not have this explicit thing. So, what we normally do is to convert this as g of X plus S square equal to zero, where we introduce a variable S associated with every constraint and that S can be negative or positive. Therefore, since this is less than or equal to zero, we would write g of X plus S square is equal to zero. So, this quantity is less than or equal to zero. Whether S is positive or negative, S square will always be positive, so you get g of X plus S square equal to zero. Now, we introduce a Lagrangean multiplier, as many multipliers as the number of constraints here and then we write maximize L is equal to - you just set up the Lagrangean function L is equal to f of X minus lambda into g of X minus S square.

So, we already explained why we put a minus for a maximization problem, because a constrained problem will only bring down the value of the objective function. So, we bring this in. This lambda is called the multiplier or a dual variable associated with this. Now, we can apply the principles of optimization to get dow L by dow X equal to zero, dow L by dow lambda equal to zero and dow L by dow S equal to zero. When we do this, we get this form. So dow L by dow X equal to zero would give us del f of X minus lambda del g of X equal to zero because, this is dow by dow X del f of X. This will not be there, because we are partially differentiating with respect to X. So, minus lamda del g of X equal to zero. Dow L by dow lambda would give us minus g of X plus S square equal to zero from here and dow L by dow S would give us lambda minus two times lambda$_i$ S$_i$ equal to zero. Now, these three are typically the equations that we get, when we partially differentiate L with respect to X, with respect to lambda and with respect to S. Now, we can always write this as lambda greater than or equal to zero, del f of X minus lambda del g X equal to zero. We write this, lambda$_i$ g$_i$ of X equal to zero and g$_i$ of X less than or equal to zero. So, we write this as these four important things. Now, what are they? What is the relationship between this and this?

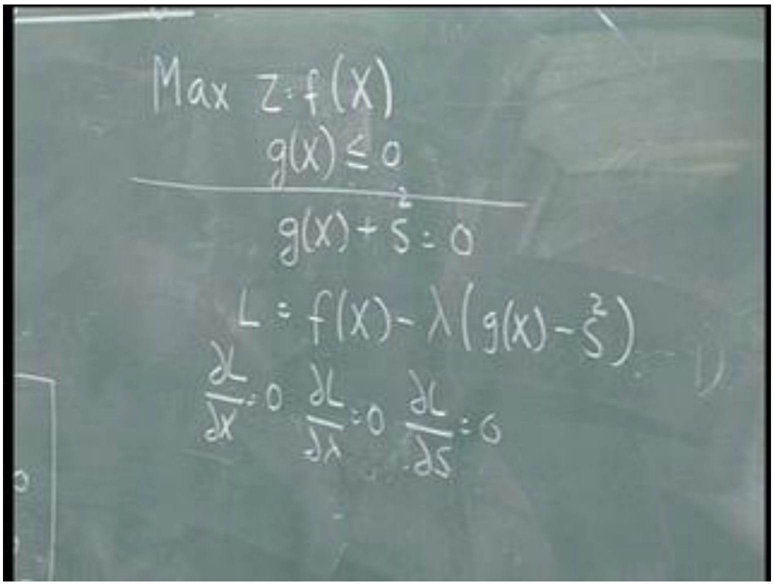
(Refer Slide Time: 54:40 min)



This is exactly the same as this, there is no problem. Now, g of X plus S square equal to zero is the same as g of X less than or equal to zero. So, now, we write minus 2 lambda$_i$ S$_i$ equal to zero is now written as lambda$_i$ g$_i$ of X equal to zero and lambda greater than or equal to zero. So, lambda greater than or equal to zero comes only then we have this quantity, which will reduce the objective function. So, when we put a minus lambda, lambda has to be greater than or equal to zero for the inequality that we have. So, lambda is greater than or equal to zero. So, the only other thing that we have actually left out is lambda and this would tell us either lambda is equal to zero or S of i equal to zero or both equal to zero. So, this is lambda greater than or equal to zero.

So, when lambda is greater than zero strictly, then we have lambda is greater than zero, g of X is equal to zero. When lambda is zero, then g of X is less than or equal to zero. Therefore, we get lambda_i into g_i of X equal to zero. So, this is now replaced by lambda greater than zero and lambda_i g_i of X is equal to zero. This is a general form that we will use to solve problems of this type.

(Refer Slide Time: 56:10 min)



Now, these conditions are very well known Kuhn Tucker conditions, which form the basis for solving nonlinear optimization problems. Sometimes, these are called KKT conditions, Karush Kuhn Tucker conditions; but, we use the term Kuhn Tucker conditions for this. So, the Kuhn Tucker conditions are the actual conditions that we can generalize and we need not derive this every time. We can simply generalize this and then, for every nonlinear problem, which is described in this form, maximize Z equal to f of X g of X less than or equal to zero, we can simply write the Kuhn Tucker conditions and depending on the resultant system that we get, we can solve those equations and inequalities. Some of them may be equations, some of them may be inequalities, and some of them may have higher degree, which all depends on what happens with this f of X and with this g of X.

If f of X is cubic or has a higher power, then del f of X will be quadratic or more and same with g of X. So, once we write the Kuhn Tucker conditions, we simply solve the resultant equations and inequalities to get the maximum or the minimum. Sufficiency will have to follow, but we are not looking at sufficiency in this lecture series. In the next lecture, we will see the application of Kuhn Tucker conditions to a quadratic programming problem.