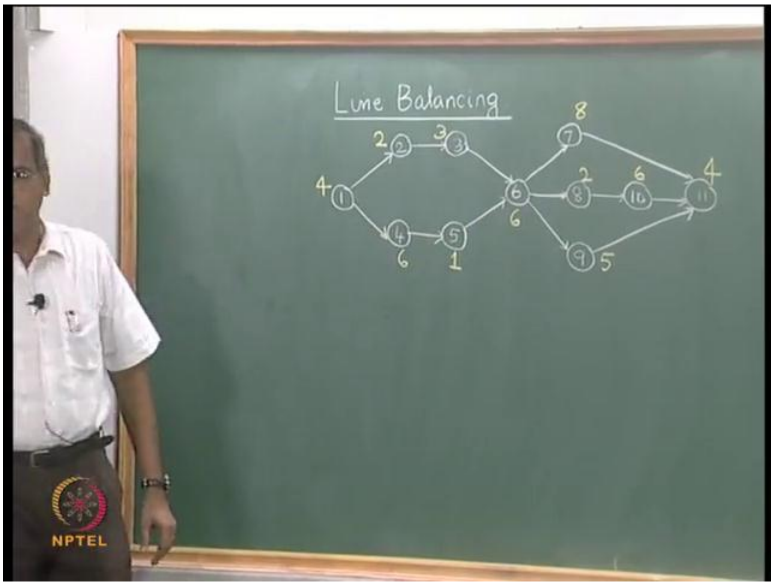
Operations and Supply Chain Management

Prof. G. Srinivasan
Department of Management Studies
Indian Institute of Technology, Madras

Lecture - 31
Line Balancing

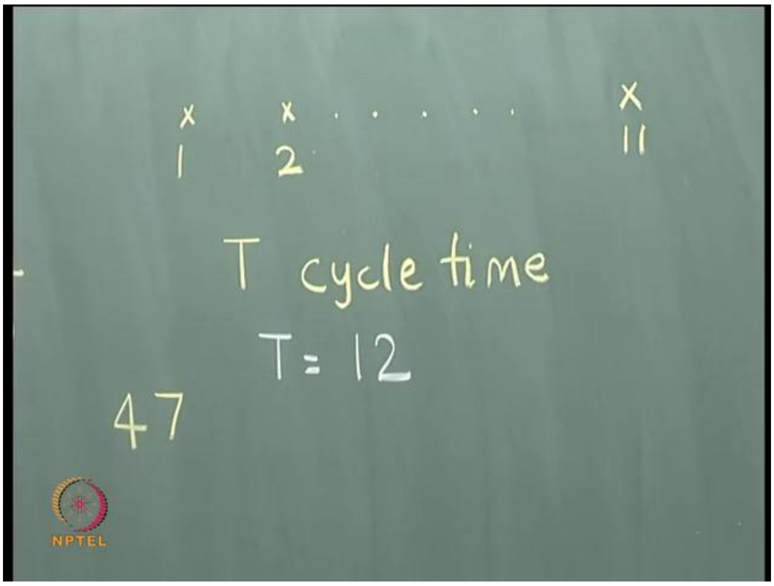
(Refer Slide Time: 00:18)



In this lecture, we continue the discussion on Line Balancing. So, this network represents the activities in a typical line balancing situation. So there are 11 activities that have to be performed, one could assume that 11 components are subassemblies are assembled to form a final product. Now, there is an activity time associated with each of these activities, which are given in yellow, the 4 units, 2 units, etcetera; units could be in minutes or seconds depending on the assembly. There is also a precedence relationship, which is also captured in this network.

For example, activity number 2 or say component number 2 cannot be assembled unless one is assembled, so 1 precedes 2. Similarly, 1 precedes 4, 4 precedes 5, 3 and 5 precede 6 and so on, which means we cannot do activity 6 unless both 3 and 5 are completed. As we have mentioned in the previous lecture, if there is only one person who is actually assembling everything, then this person would take the sum of all the processing times which is, 12, 15, 16, 22, 30, 32, 47 units, to assemble to each final product.

(Refer Slide Time: 01:43)



So, if one person is sitting and doing everything it would take 47 time units, if two people do it, then at steady state it will be 47 divided by 2, every 23 and a half time units 1 could get an output and so on. But, in general it does not happen that way, because some of these could involve equipment and machinery and testing devices, which we may not duplicate so much. So, you would rather have different workstations or benches to which these activities are assigned.

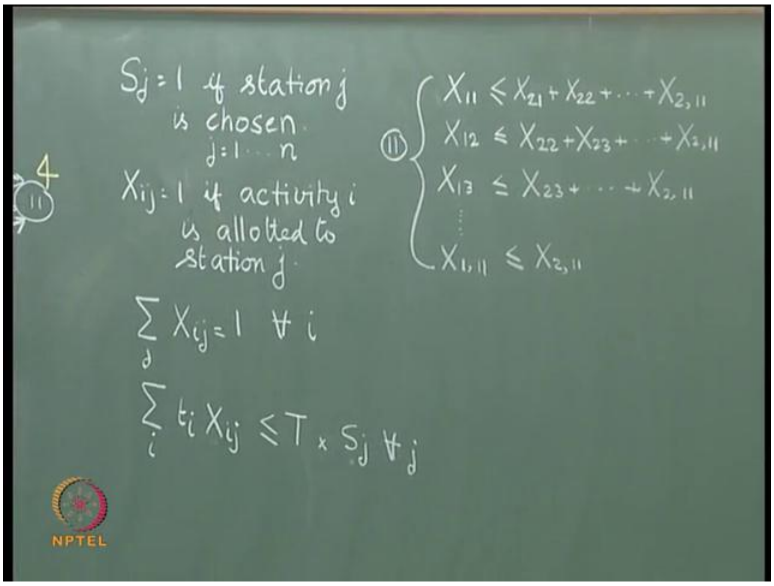
So, if we imagine a situation where there are the 11 activities, and there are 11 workstations or benches say 1, 2, 3, 4 up to 11. We also assume in this example as an every line balancing problem, there the order 1, 2, 3, 4 up to 11 is feasible, so if we have 11 benches or workstations, which say 11 operators, and each person carries out 1 activity. Then the first unit will come out at time equal to 47, but at steady state the output of the system will be the bottleneck, which will be the maximum of the individual assembly times, which would be 8 units.

At the same time we also define something called T called cycle time, which is the expected output from the system. For example, in this, if we say that, if we say T equal to 12, then the demand is such that we would expect an assembly to come out every 12 time units. So, once we define this T, the problem is one of trying to find out, how many workstations we require - minimum workstations such that, each task or activity is assigned to only 1 workstation. The time taken in a workstation does not exceed the cycle time 12, and the precedence relationships among the activities are satisfied, so this is called the line balancing problem.

So, given a set of activities that have to be performed, given their activity times and the precedence relationships, for a given cycle time T what is a minimum number of workstations it is required? How these tasks are assigned to these workstations such that, the total time taken in a workstation is less than or equal to T. Each tasks is assigned to only one workstation and the precedence relationships are satisfied; now this is called the assembly line balancing problem. So, let us first show a formulation of the line balancing problem and then we would look at a couple of solutions to the line balancing problem.

Now, there are 11 tasks, so we could say that there can be a maximum of 11 workstations or 11 benches that we could have 1 for each. But, then that would give us a feasible solution with 11 workstations or 11 benches, where each activity goes to 1 workstation. And the cycle time would still be less than or equal to 12, in this case because the cycle time will be 8. Since, the cycle time is less than or equal to 12, it is possible to add activities to workstations and thereby try and minimize the number of workstations.

(Refer Slide Time: 05:56)



So, we now start defining the first variable as S j equal to 1, if workstation j is chosen, we initially assume that a maximum of 11 stations can be used, there are n activities we would assume that a maximum of n stations can actually provide us a feasible solution. And therefore, we would like to minimize the number of stations, so here we would define j equal to 1 to 11 for this problem or j equal to 1 to n for a general problem. So, we would say that a maximum of 11 stations are there, how can we bring down the number of stations.

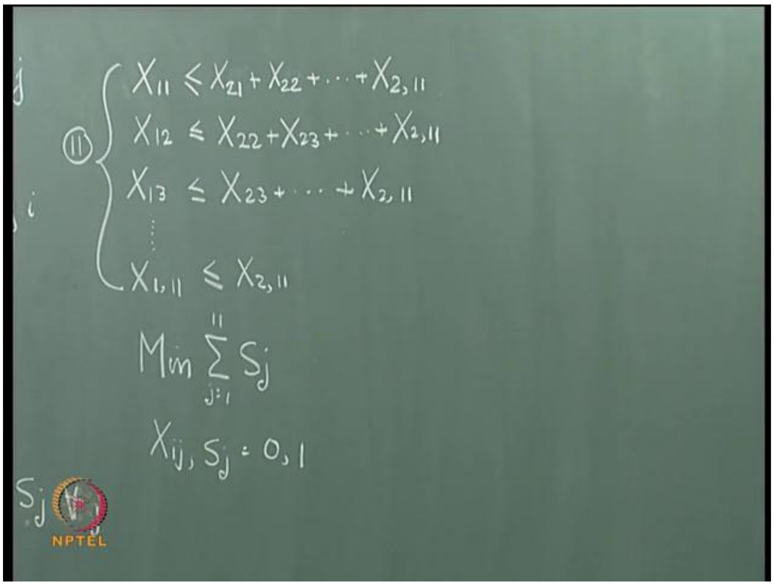
Now, if we can solve it with 5 workstations, then the solution would take 5 out of the 11 values will take 1 and the remaining 6 out of the 11 values will take 0, saying that these stations are not created. So, S j equal to 1, if station j is chosen or created, and let X i j equal to 1, if activity i is allotted to station j; now we start writing the constraints, each activity should go to only 1 station. So, we would have the constraint sigma X i j summed over j equal to 1 for every activity i, each station the activities allotted to a station, a sum of the activity time of the activities allotted to a station, should not exceed T.

So, this would be sigma T i is the activity time for activity i, so T i into X i j summed over i should be less than or equal to T for every j, j represents the workstation. So, the activities that are allotted to a workstation, the sum of the activity times of those activities should not exceed the cycle time T. More importantly we can allot an activity to a workstation, only when the workstation is chosen. So, we redefine it by saying that, if there are activities allotted to a chosen workstation, then the sum of times of those activities should not exceed T.

So, the left hand side tells us, the sum of the activity times of those activities that are allotted to a station, now this is the cycle time, but then we could also add an S j, where S j equal to 1 when the station is chosen. This also means this constraint would also ensure this is T into S j, so we will write is as T into S j for every j. So, only when S j equal to 1 that workstation is chosen and to which activities can be assigned. So, you cannot assign activities to a workstation that is not chosen, and for every chosen workstation the sum of the activity times should be less than or equal to T.

This constraint also helps us in linking X i j variables with S j variable, we still have not come to the objective function, we will do that, right now this links the X i j variable with the S j variable. Then we have this precedence constraint, so let us take 1 precedence which is say 1 and 2, so activity 2 can be assigned only after activity 1 is assigned, it means 1 precedes 2.

(Refer Slide Time: 10:47)



Now, this would mean that, for example we have X 1 1, which means activity 1 is assigned to station 1, activity 2 can be assigned to station 1, 2, 3, 4 up to 11. Now, this implies that, if 1 and 2 are assigned to the same station 1, the operator will first assemble component 1 and then assemble component 2. So, we this is modeled as if 1 goes to 1, then X 1 1 should be less than or equal to X 2 1 plus X 2 2 plus X 2 11, this would mean X 1 1 plus X 2 1, X 1 2 plus X 2 2 component 1 goes activity 1 is in workstation 1. Then activity 2 can go to workstation 1 or workstation 2 or 3 or 4 and so on.

But, then if for example, activity 1 is allotted to workstation 2, then activity 2 should not be allotted to workstation 1, it can be allotted to 2, 3, 4 etcetera till 11. So, this will be modeled as X 1 2 will be X 2 2 plus X 2 3 plus etcetera plus X 2 11, similarly if activity 1 goes to workstation 3, then we have X 2 3 plus etcetera plus X 2 11. So, like this the last set for this pair would be X 1 11 is less than or equal to X 2 11, so these 11 constraints capture the precedence relationship between 1 and 2.

Similarly, to capture the precedence relationship between 1 and 4, we will have 11 other constraints, so we have to first find out what are all the, how many precedence relationships we have, so here we have the number of arcs essentially represent, the number of precedence relationships. So, we have 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 and 13 precedence relationships, each precedence relationship is between a pair of activities and then we have to have 11 constraints for each of the precedence relationships.

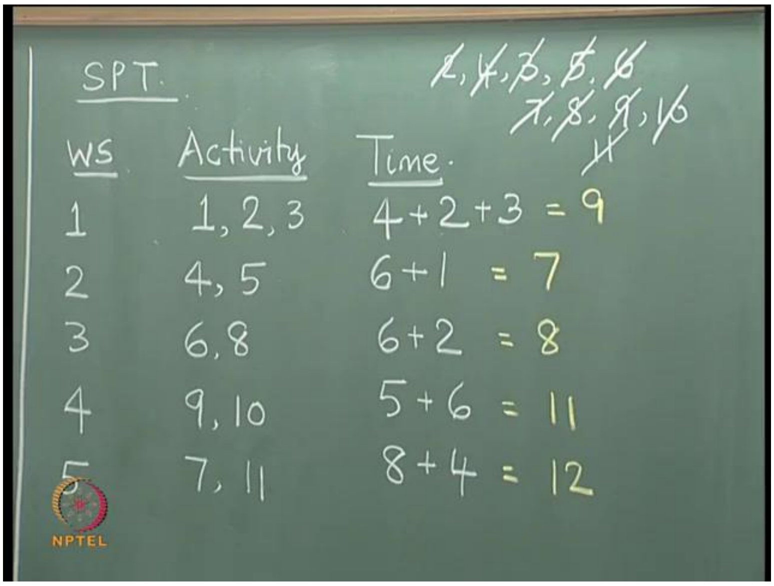
Giving us 13 into 11, 143 constraints to handle this, we are just shown the 11 constraints for one particular precedence relationship. Now, there are actually better ways of writing the precedence relationships, but they are not going to emphasize on that, the purpose here is only to show how the formulation looks like. There are slightly more efficient ways of reducing this 143 constraints into a lesser number of constraints, which capture the same thing.

But, what we have shown here is, the reason or the rationale behind the precedence constraints, where if there is a relationship between 1 and 2, the workstation to which activity 2 is assigned should be higher numbered workstation or equal to than the workstation to which 1 is assigned. So, if 1 is assigned to workstation 1, then 2 can be assigned to 1, 2 up to 11, but if 1 is assigned to workstation 2, then 2 can be assigned only up to 2 to 11, so we cannot have X 2 1 assignment there.

So, we cannot this would prevent activity 1 from being assigned to workstation 2 and it will it will have activity 2 assigned to workstation 1, it will prevent that, because if activity 2 is assigned to workstation 1, through this all of these will be 0. Therefore, we will have 1 less than equal to 0 and this will prevent activity 2 from being assigned to workstation 1. So, we have 11 such constraints which model this precedence relationship and then we have 13 such precedence relationships, then the objective function is to minimize the number of workstations.

So, minimize sigma S j, j equal to 1 to 11 will be the objective function associated with this of course, all X i j is and S j are binary variables, so they are either 0 or 1. So, optimally solving this 0 1 problem, can give us the optimum solution to the line balancing problem. But, then we will look at two very simple heuristics to solve this line balancing problem. One is a heuristic solution, which is based on shortest processing time based heuristic.

(Refer Slide Time: 16:28)



So, we would start by saying workstation activity time, so we would say to begin with, we can do activity 1 only, because all the rest of them have precedence relationships which are related to activity 1, so we start with activity 1, which goes to workstation number 1. We assign activity 1 and activity 1 has processing time equal to 4. Now, that we have assign activity 1 to workstation 1, 2 and 4 are available now, because activity 1 has been allotted. So, we have let us say 2 and 4 with us, now 2 requires the time of 2 units, 4 requires the time of 6 units.

So, we would assign based on SPT rule we would assign activity 2 and then it takes 2 more units, so activity 2 has been assigned and since, activity 2 has been assigned we have activity 3 that is available now. So, 3 and 4 are available, when the 3 and 4 are available, we can try and assign one of than, 3 requires 3 units, 4 requires 6 units, so once again based on SPT, so 3 will go here, so plus another 3 units. So, since 3 is assigned we cannot add 6 into the list, because 4 is yet to be assigned.

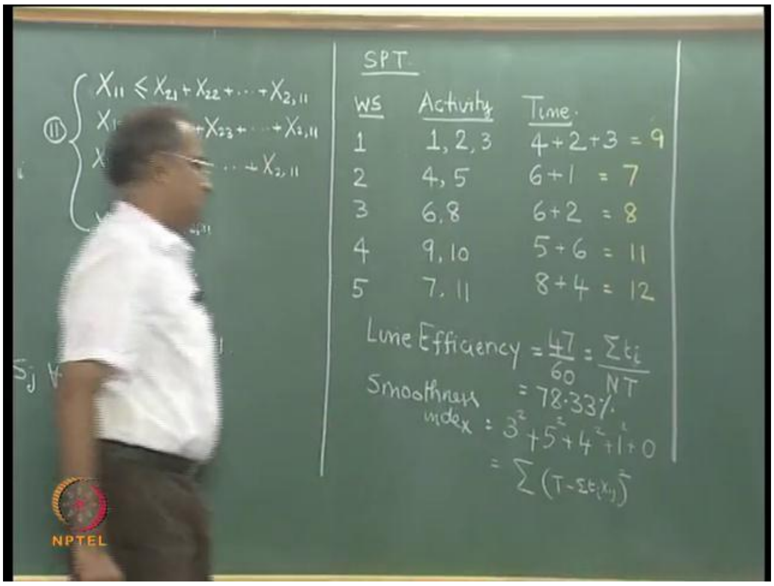
So, we have only 4 that is available, so we try and put 4 here, but then we realize that the cycle time exceeds 12 as 4 plus 2, 6 plus 3, 9 plus another 6 is 15, so cycle time exceeds 12. So, we create a second workstation and try and put activity 4 here, this goes at time equal to 6. Now, when activity 4 is allotted, 5 is the only one that is remaining, so you put 5 comes here, 5 is the only one that remaining, so 5 goes here ((Refer Time: 19:09)) plus 1 and once 5 is allotted 6 can be brought in, so 6 comes here.

Now, 6 requires 6 time units, so we try and put 6 here and we realize that it exceeds the value of 12, so create a third workstation with activity 6 with time 6, 6 goes now; the moment 6 is allotted we could bring in 8, 7, 8 and 9. So, 7, 8 and 9 with 7 has time equal to 8, 8 has time equal to 2, 9 has time equal to 5, so we choose 8 based on SPT rule. So 6 and 8 plus 2, so 8 goes now when 8 goes, 10 can come in. So, we have 7, 9 and 10 available, 7 requires 8 time units, 10 requires 6 time units, 9 requires 5 time units. Now the smallest is 9 with 5 time units.

We try and put it here, but we are not able to, because it exceeds the cycle time. So, create a 4th workstation with 9 here and time equal to 5. So 9 goes, then we have 7 and 10, ((Refer Time: 20:52)) this is 8 and this is 6, so the smallest number is 6. So, we take 10 plus 6, so 10 goes, we can still we cannot bring 11 into it, because 7 has not yet been done. So, now, 7 is the only one that is available. So 7 requires 8 time units adding an 8 here will exceed the cycle time. Create one more workstation with 7 and time equal to 8, so 7 goes.

Now, 11 is the only one that is remaining 11 has 4 time units required 4, so add 11, 7 and 11, 8 plus 4, so now all the activities are assigned, and we get a solution like this. Now, the cycle times in each of these are 4 plus 2, 6 plus 3 9, 6 plus 1 7, 6 plus 2 8, 5 plus 6, 11 and 8 plus 4 12. So, based on the shortest processing time rule, we now have a solution with 5 workstations and these are the allotments to it. Now, the goodness of a solution is measured, using two, a primary measure and a secondary measure. Now the primary measure is called line efficiency.

(Refer Slide Time: 22:44)



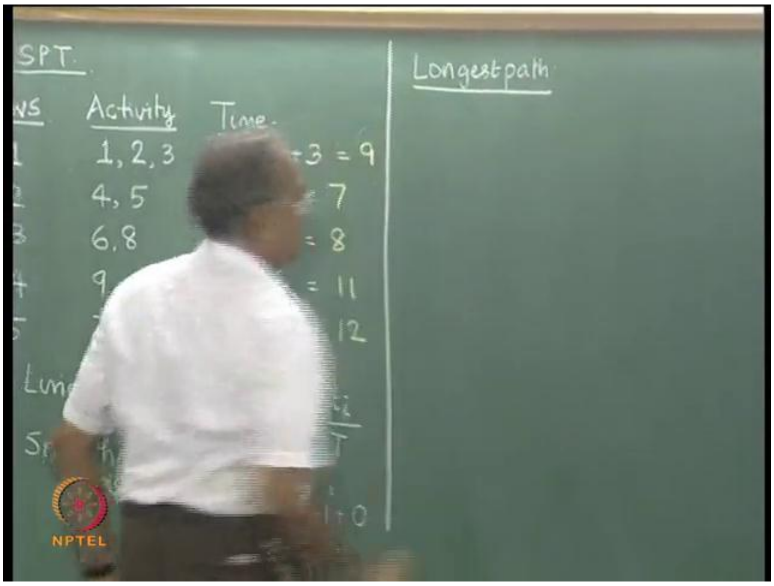
Line efficiency is we now have 5 lines with the cycle time of 12 and the maximum is also 12, so 5 lines with 12 cycle time is about 60 time units, and the total of these 47 time units are now made in 5 stations each with 12 in 60. So, line efficiency is 47 by 60 or in general it is sigma T j or T i sum of the activity times divided by N into T, where N is the number of workstations chosen and capital T is the cycle time. So, this is 47 by 6, so dividing 6 7's are 42, 50, 6, 8 are 48, so line efficiency is 78.33 percent. We also realize that in the line efficiency equation the numerator is a constant and actual variable comes only in the denominator with this N, because this is the constant and the N and T, depending on capital T, the N gets defined.

So, the actual measure is dependent only on the denominator and not on the numerator, numerator is a constant. Now there is a secondary measure, which also tells us how balanced the line is within these 5 workstations. For example, this is going to take 9 units 7 units 8 units 11 units 12 units, now when even though the allowed cycle time is 12, if some stations take less time and some stations take 12, there is some kind of an imbalance. Now, the operator who is working on the workstation, which takes 12 units is going to be occupied all the time, whereas the operator who works on this is going to use only 7 time units, for every 12 time units.

Or will be producing excess and building inventory within the line, which cannot be consumed, because the slowest person is going to take 12, either this person has to stop working for every 5 time units, or this person will end up building some inventory. There are many things that matter, because the input to this is the 9 that comes from this workstation. So, we all need to ensure that the workstations are sufficiently balanced, so there is very little difference between these 5 numbers, the closer these 5 numbers are either inventory will be less or the idle time will be less.

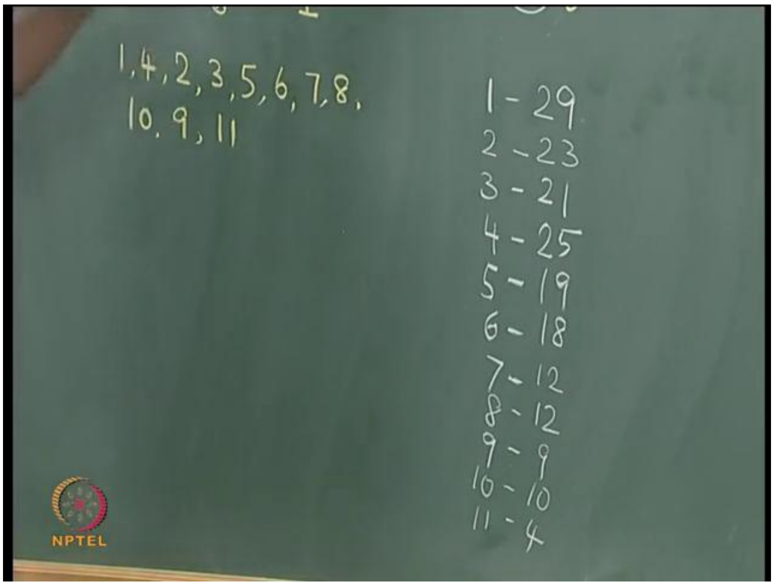
If these numbers vary as in this example, you will find that either the person has to be idle or the person will end up creating inventory which cannot be consumed. So, there is a measure called smoothness index, which tries to in this example the smoothness index will be the actual cycle time is 12, so 12 minus 9, which is 3 square 12 minus 7, 5 square plus 4 square plus 1 square plus 0. So, in general the smoothness index formula will be sigma T minus sigma T i X i j whole square, so this represents the time in a workstation, this represents a cycle time; so square and sum as a smoothness index. Larger the line efficiency more efficient is the line, smaller the smoothness index better it is for the line, so the solution to a line balancing problem is measured on these two factors, which are line efficiency and smoothness index.

(Refer Slide Time: 27:58)



Now, let us look at another solution to this problem, which is this heuristic based on longest path in the network.

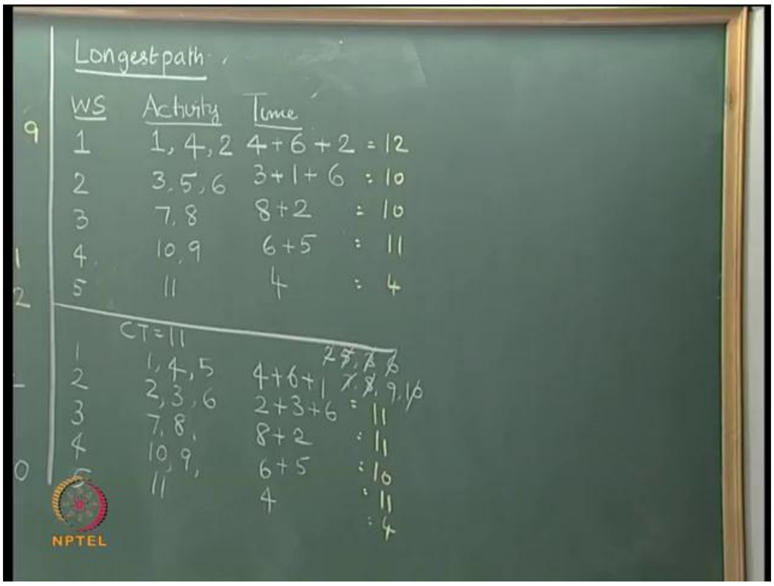
(Refer Slide Time: 28:11)



Now, if we take activity 1, the longest path is going to be 4 plus 6, 10 plus 1, 11 plus 6, 17 plus 8, 25 plus 4, 29 will be the longest path for 1, once again 4, so you could reach 6 using this which would give us 5 or using this which would give us 7. So, 4 plus 7, 11 plus 6, 17 plus 8, 8 and 5, 25 plus 4, 29. For activity 2 the longest path will be 5 plus 6, 11 plus 8, 19 plus 4, 23. For activity 3 the longest path will be 3 plus 6, 9 plus 8, 17 plus 4, 21. For activity 4 the longest path will be 7 plus 6, 13 plus 8, 21 plus 4, 25.

For 5 the longest path will be 1 plus 6, 7 plus 8, 15 plus 4, 19. For 6 the longest path will be 6 plus 8, 14 plus 4, 18. For 7 it will be 12. For 8 it will be 12; for 9 it will be 9; for 10 it will be 10 and for 11 it will be 4. So, these are the longest paths for the 11 activities that we have, now we will sort the activities in the decreasing order of the longest path. And the arranged order will become 1 first with 29, 4 with 25, 2 with 23, 3 with 21, 5 with 19 6 with 18, 7 with 12, 8 with 12, 10, 9 and 11. So, this is the order of arranging the activities based on the longest path, which is there in this network. Now, we use this information and develop another heuristic. Now such a heuristic would start like this.

(Refer Slide Time: 30:52)



So, once again we have workstation activity and time, so we start with workstation 1 and activity 1 is assigned to workstation 1 with time equal to 4, now the moment activity 1 is assigned we have 2 and 4 that are available. Now, earlier we chose that activity based on shortest processing time, now we will choose the activity based on which one appears early in this order. So, between 2 and 4 you realize that 4 actually appears earlier than 2, so we choose, so activity 1 followed by activity 4, so 4 goes 4 requires time equal to 6, so 4 plus 6.

Now, once 4 is over, then you can take 5 into it now go back and see between 2 and 5 which one goes first, so 2 goes first, but then we realize that if we add 2 we have to create another workstation, because this requires or we cannot take 2 here. So, between 2 and 5, 2 comes first, so 2 requires time unit of 2. So, we add time unit of 2 here, so this goes activity 2 plus time unit equal to 2.

Now, once we have activity 2 competed, then we have 3; let us here that comes in. Now between 3 and 5, 3 comes first in this. So we take 3 requires 3 units, we cannot add 3 into it, because it exceeds the cycle time of 12. Create a 2nd workstation with 3, activity 3 and with time equal to 3 units. So, once activity 3 is allotted we cannot allot 6, because 5 is still to be allotted. So we allot 5, so 5 is the only one. So, we do not look at this order, we look at this order only when we have more than 1 activity available for allocation.

So, 5 is the only one, so it takes plus 1 time unit, so now 6 is available. 6 is the only one available and 6 can be added here, so plus 6. 6 also takes 6 time units once activity 6 is allotted, we have 7, 8 and 9 that are available. So, we have 7, 8 and 9, we have 3 activities that are available, now we look at this order first it is 7, and then it is 8, so we take 7, 7 requires 8 time units 7 goes. We cannot add 11, because we still need to do 8 and 9, now between 8 and 9 8 comes first 8 requires 2 units, so allocate 8 here, 8 requires 2 units; 8 goes.

Now 10 comes first, so now 10 comes here next, now between 9 and 10 we look at this order and realize that 10 comes ahead of 9, now we cannot add 10 there, so create a 4th workstation and then put 10 here with time equal to 6. So 10 goes. So, that brings us 11 into the picture, so between 9 and 11 9 comes ahead, so 9 plus 5, so 11 is the only one that remains, so 11 requires 4 time units. So, 11 cannot be fit into this, so create a 5th workstation states activity 11 and has cycle time equal to 4.

So, this has cycle time equal to 12, 10, 10, 11 and 4, now this will have the same line efficiency, because there are 5 stations with maximum time equal to 12 - same line efficiency. But, this will have a larger smoothness index, because this would contribute 12 minus 4 the whole square. So, this from a smoothness index point of view, this is not very desirable, because it is going to create a lot of variation, so either a lot of inventory or a lot of idleness into the system.

Now, we can actually do another thing here which is this, now the next effort can be in both these solutions, we have a total of 47 time units for allocation and by choosing 5 workstations and having a maximum time of 12, we have 60 time units available, so the line efficiency was 47 by 60. Now, if we have 5 workstations and instead of cycle time of 12, if we can have the cycle time of 11, then not only will the line efficiency go up, but the smoothness index will also come down.

So, the next issue will be given the number of workstations can be reduce the cycle times slightly further, so one could think in terms of solving this problem again, and using some heuristic to see whether we can actually get a cycle time of 11 instead of a cycle time of 12. Now, if we try to do this here, can we get a cycle time of 11 in this, let us try and do this once again with a cycle time equal to 11. And let us do it little quickly, so workstation 1 would get activities 1 and 4 with cycle time 4 plus 6.

So, 1 and 4 are allotted, then we have 2 and 5 remaining, now in this order 2 comes first

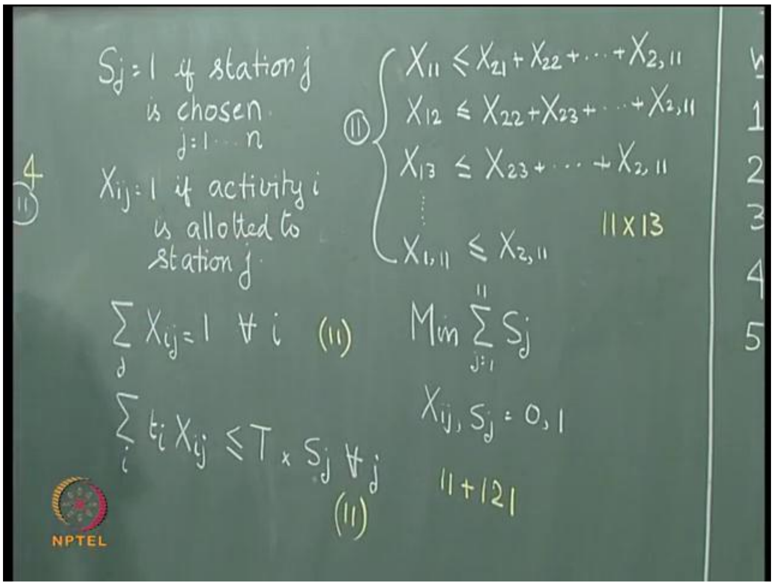
2 requires 2 time units, so create a second workstation with activity 2 and time 2. So 2 goes 5 remains, 3 comes in. Now, between 3 and 5, 3 requires, 3 comes first, so 3 can be added here plus 3, 3 goes. Now, 5 is the only thing that is remaining. If 5 is the only activity remaining ordinarily we would try and put 5 here, because this is the current workstation, but we also realize that the 5 can actually go here, because both 1 and 4 are satisfied.

So, you could add 5 here ((Refer Time: 39:24)) to give a plus 1, so 5 also goes which brings us 6 into the picture. 6 requires 6 units of time, so 6 can go here plus 6, so 6 is over. Then we have 7, 8 and 9; out of these 7 comes first with time equal to 8, so 3rd workstation 7 comes first with time equal to 8; 7 goes. Then we have 8 and 9, 8 comes next, so 8 comes next plus 2, so 8 goes then 10 comes in, now between 10 and 9, 10 is first.

So, create a forth workstation with 10, 10 has time equal to 6 and goes. Then comes 9, 9 has time equal to 5 and then we need to create a 5th workstation with 11 and time equals to 4. Now, both the line efficiency and the smoothness index will be better for this solution, because line efficiency will be higher, because it will become 47 by 55 instead of 47 by 60. Smoothness index would also be slightly better, because the loads on the workstation are 11, 11, 10, 11, 4, so smoothness index will also be better here compared to this one.

So, we could if we end up from the solution if we realize, that the line efficiency is slightly lower, then we could think in terms of for the same number of workstations that I have in this solution, can I bring down this T by 1 and get a better line efficiency. So, we can work on that till we cannot improve the line efficiency any more. Now there are a few other advantages of using these heuristic solutions, now if we go back and try to solve this optimally in the way that has been given here.

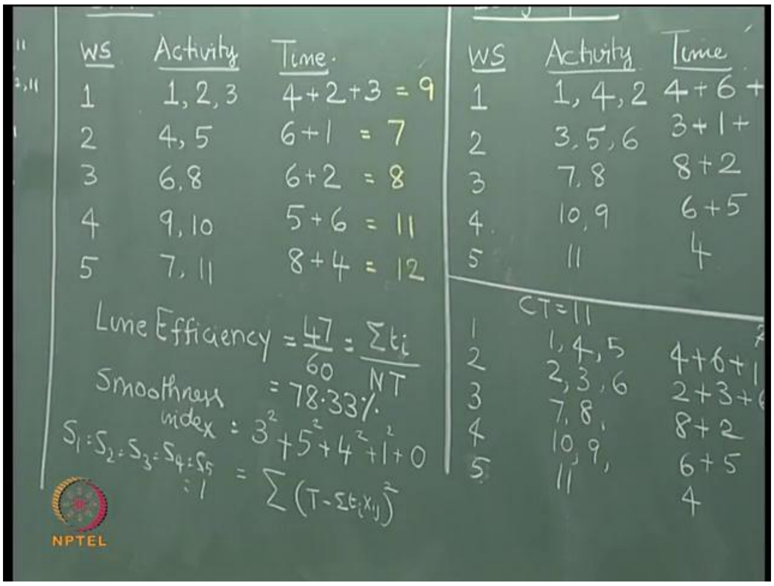
(Refer Slide Time: 41:58)



For this particular problem, we will have 11 variables for S j, because we maximum of 11 S are available, S 1 to S 11 and then we have another 121 variables for the X i j. And then we have many constraints, this will be for every i, so this will be 11 constraints, this would also be for 11 constraints and this set will be 11 for each precedence and there are 13 precedence, so 143 constraints. So, if we solve it this way even though as I mentioned there are better ways of representing this, if we solve it as it is we are talking of 132 variables.

And 143 plus 22, which is 165 constraints. Now we solve a slightly large binary problem with 132 variables and 165 constraints, now let us try and map, let us say this solution into this formulation and see what happens. Now, this feasible solution has 5 workstations.

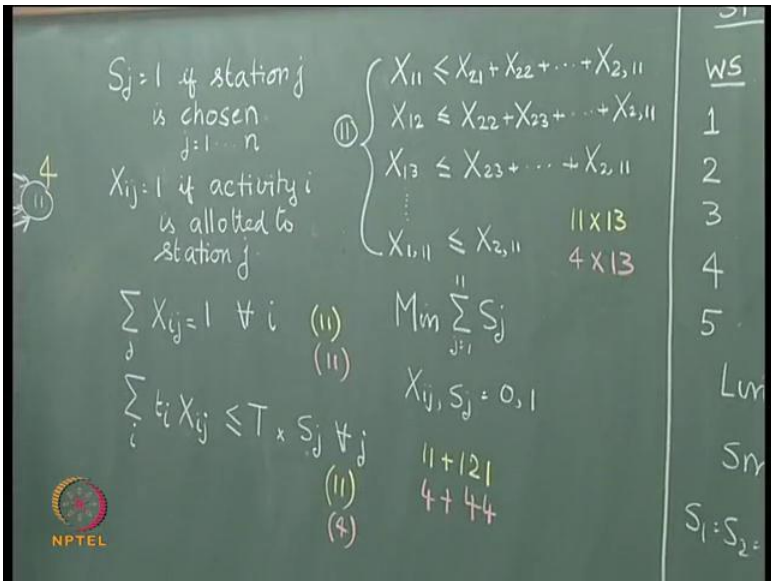
(Refer Slide Time: 43:19)



So, if I try to map this here, for this I will have a solution which this is like S 1 equal to S 2 equal to S 3 equal to S 4 equal to S 5 equal to 1, so 5 workstations. And I will have X 1 1, X 2 1, X 3 1, X 4 2, X 5 2, X 6 3, X 8 3, X 9 4, X 10 4, X 7 5, X 11 5 equal to 1 that will be the solution, and we map it to this. Now, what does this mean, this has given me a feasible solution with 5 workstations, so a feasible solution is an upper bound to a minimization problem.

And therefore, since I have a feasible solution with 5 workstations, I am not interested in feasible solutions with optimum solution, cannot be 6 or 7 or 8 or 9 or 10 or 11, so the optimum solution can have 5 workstations or less. So, what I will do is, I will now try and solve to find out, if there is an optimum solution with 4 workstations, because already I have a feasible solution with 5 workstations. So, I will try and see, if I can have an optimum solution with 4 workstations. So, if I try and solve it with 4 workstations, then you can see the number of variables and constraints that I will have.

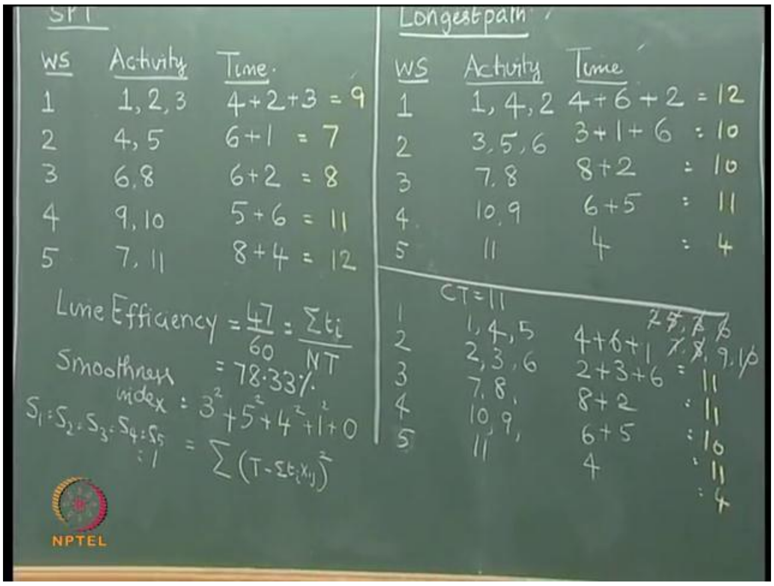
(Refer Slide Time: 44:57)



Now, if I solve for 4 workstations there will be only 4 S j values, and there will be 44 X i j values, so X i j will now take only 44 variables, because each of the 11 activities can be assigned to each of these 4 workstations. Now, the constraints X i j, this will be 11 constraints, this will be only 4 constraints and here, for every precedence relationship there will be only 4 constraints into 13. So, I end of having 48 variables and 52 plus 11, 63 plus 467 constraints and if the optimum solution is 4, it will give me the optimum solution.

So, getting a heuristic solution first, actually helps us in getting the optimum solution in a slightly better way, otherwise we would be solving an optimization problem with 132 variables and 165 constraints. And because of the heuristic solution that we have, it is enough now that the binary programming to get the optimum solution will have 48 variables and 67, 52 plus 11 63 plus 4, 67 constraints, so it is enough to formulate and solve that problem. Now, in case the optimum solution does not have 4 benches, but has 5 benches then the optimum would give us infeasible, because we have worked with 4 benches.

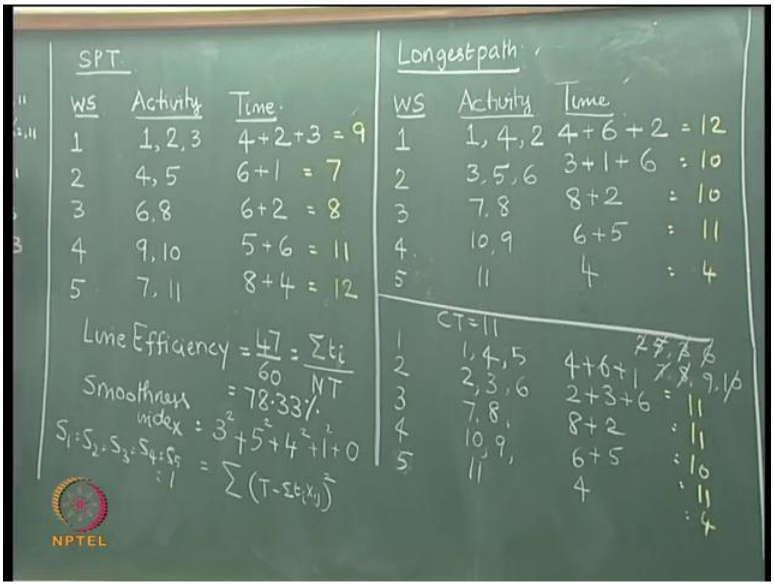
(Refer Slide Time: 46:44)



Then we already know that we have a solution with 5 benches, and therefore this is 5 benches are 5 workstations, and therefore this is optimum. If the optimum had a solution with 4 workstations, then it will show that solution, then we will have to try and see whether a solution with 3 is possible. Now, in order to approach that we will look at it from a lower bound point of view, because total time required to assemble is 47 and if we have 12 as a cycle time, then the minimum number of workstations which is a lower bound is 47 by 12 upper integer value, which would give us 4.

So, there cannot be an optimum solution with 3 for this case, so the only case that we have to look at is can we have a solution with 4 workstations, which means it is enough to solve this problem with 48 variables and 67 constraints. And if we actually do that, we get an optimum solution to this problem which will look like this, so the optimum solution is like this.

(Refer Slide Time: 48:01)



So, the optimum solution has 4 workstations, so workstation 1, activity and time, so 1, 2 and 4 with 4 plus 6 plus 2, station 2 is 3, 5, 6, 8 with 3 plus 1 plus 6 plus 2. Station 3 is 10 and 9 or 9 and 10 with 6 plus 5 and station 4 is 7 and 11 with 8 plus 4, so this is the optimum solution to this particular problem. Where we have 1, 2 and 4, once again precedence are satisfied, 3, 5, 6 and 8 precedence are satisfied, 9 and 10 precedence are satisfied, 7 and 11 precedence are satisfied.

So, the cycle times here will be 4 plus 6, 10 plus 2, 12, 12, 11, 12 the total equal to 47, line efficiency is equal to 47 by 4 into 12, which is 47 by 48, which is the best that is achievable. And smoothness index, this is about 98 percent the smoothness index will be 12 minus 12, the whole square plus 12 minus 12 the whole square plus 12 minus 11 whole square plus 12 minus 12 whole square, which will be 1, which is the smallest that is possible. So, this is how we solve the assembly line balancing problem, so we first try and find out given a cycle time, what is a minimum number of workstations that are possible.

Now, if we get to the optimum solution, we first in order to make the optimum solution a little simpler, in terms of number of variables, we first try and get a heuristic solution like what we have shown here. So, in for 11 activities we got a heuristic solution with 5 workstations, and therefore we try and solve the optimum solution case with 4 workstations. So, that if there is a solution it will show, if the optimum is 5 then it will show in feasibility or alternatively we could actually do it with 5. So, the number of variables and number of constraints will increase slightly and then if the optimum is 5 it will show up as 5, if the optimum is 4 it will show up at 4 and so on.

In fact, if the optimum were 3, which is not possible in this case, then we solved it for 4 stations, if we could get a solution with 3 it would still show the 3, but in this case it is not possible, because the total is 47 and the cycle time is 12, so minimum of 4 workstations are required. So, the main problem is to minimize the number of workstations for a given cycle time, the next problem if we were to look at only heuristic solutions is to try and get invariably the heuristic solution will over estimate, the number of benches slightly.

And in such a case the problem will then be for a given number of workstations, can I reduce the cycle time further, so that the line efficiency can go up, the smoothness index can come down. But, the basic problem is always 1 that minimizes the number of workstations for a given cycle time. So, with this we come to the end of certain production control related decisions, we have seen some aspects like scheduling, as well as line balancing.

Now, after this we will start looking at some aspects of location decisions, how do we locate plants from a manufacturing and supply chain point of view. And layout decisions as to how, what are the algorithms and methods by which we can have relative location of facilities inside a given location. So, these aspects related to location decisions, we will see in the next lecture.