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

Lecture 39
FMS Loading, multiple batches and changeover times

We continue the discussion on flexible manufacturing systems.

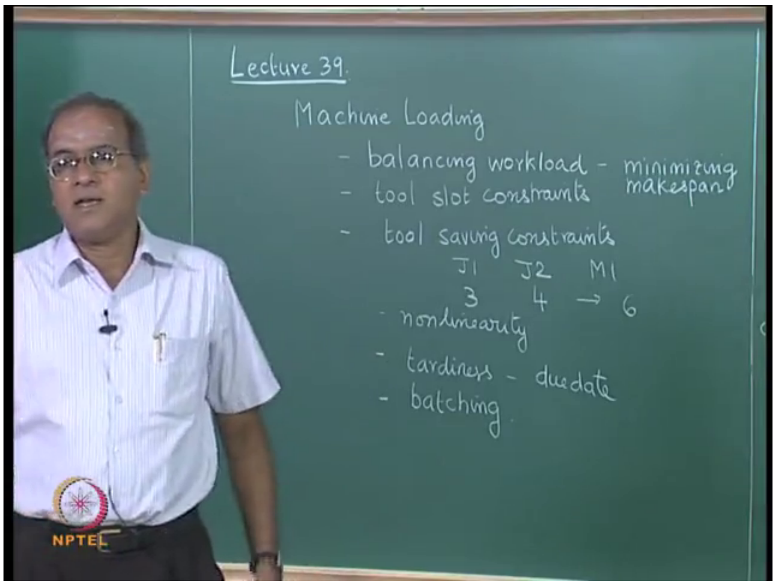
(Refer Slide Time: 00:16)

Lecture 39

Flexible Manufacturing Systems

In the previous lecture, we have seen machine loading problems, where jobs are assigned to similar machines with the objective of balancing workload, subject to tools not constraints.

(Refer Slide Time: 00:19)


Lecture 39.

Machine Loading

- balancing workload - minimizing makespan
- tool slot constraints
- tool saving constraints

J1 J2 M1

3 4 → 6

- nonlinearity
- tardiness - duedate
- batching.

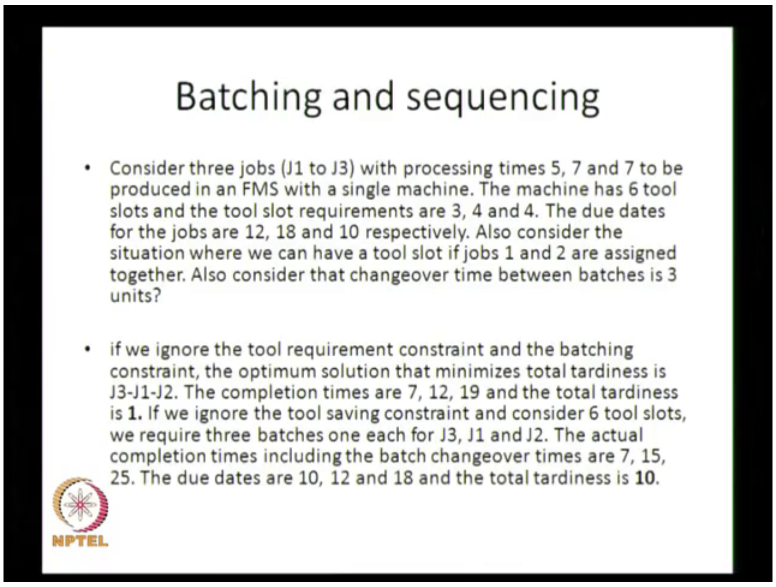
Now, the objective of balancing the workload effectively results in minimizing makespan or minimizing the total time at which all the jobs are completed. Each machine has a tool drum with a certain capacity and each job requires certain number of tools, and the job should be assigned to machines such that the total tools required is less than or equal to the tool handling capacity of the machine. In addition we also saw the effect of tool saving constraints, where if 2 jobs assigned to the same machine require common tools, then the total number of tools required is the union set of both the tool requirement and can be less than the sum of the tool requirement in case of common tools. For example, if J 1 and J 2 go to M 1 machine 1, if this requires 3 tools this requires 4 tools and if 1 tool is common then it is enough to give 6 slots for these 2 jobs put together.

We also saw that tool saving constraint can further reduce the makespan, at the same time from a modelling perspective tool saving constraints brings in nonlinearity to the problem and even the problem of reassigning jobs to machines becomes involved and complicated, if tool saving constraints are included. There is enough literature where researchers and authors have considered tool saving constraints in their solution to machine loading problems. In addition to minimizing the makespan, we also said that actually we need to minimize tardiness, which means that each job has a due date and these jobs have to be scheduled such that they meet the due dates, and in the event of them not being able to meet the due date we minimize the deviation from the due date or minimize the delay which is called tardiness.

We also mentioned that in case of large number of jobs, where the total tool slot requirement is very large and tool slot availability is small, then we perform the operations in multiple batches, where jobs are assigned to machines and then jobs are grouped to form batches and such that all the jobs within a batch the tool slot requirement is less than or equal to the tool availability on the machine. So, we also get into multiple batching problems. So, on now the decision problem is given a set of jobs which job will go to which machine, which is the loading problem in which batch of which machine which is the batching problem and in what order which is the sequencing problem. So, the makespan as well as the tardiness will depend on all 3 aspects to it.

So, let us look at this with a very simple example, to understand the effect of tools slots saving and the effect of tool slot saving on objectives such as makespan and tardiness.

(Refer Slide Time: 05:04)

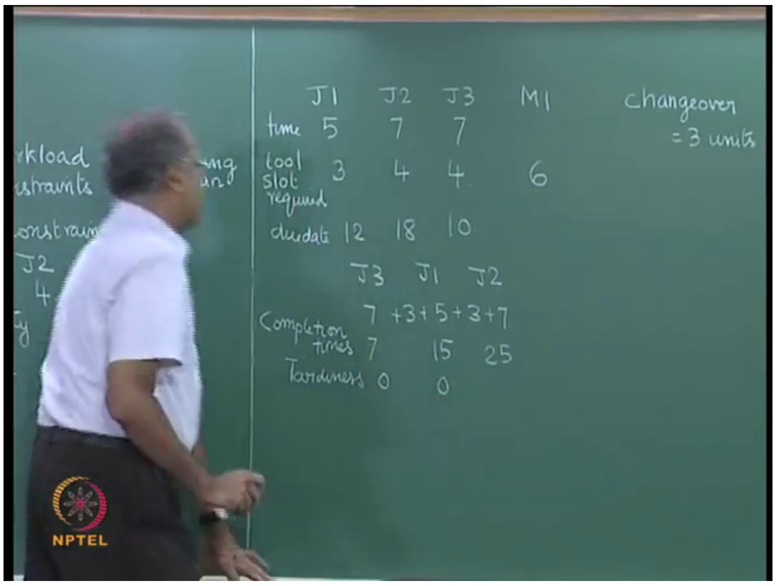


Batching and sequencing

- Consider three jobs (J1 to J3) with processing times 5, 7 and 7 to be produced in an FMS with a single machine. The machine has 6 tool slots and the tool slot requirements are 3, 4 and 4. The due dates for the jobs are 12, 18 and 10 respectively. Also consider the situation where we can have a tool slot if jobs 1 and 2 are assigned together. Also consider that changeover time between batches is 3 units?
- if we ignore the tool requirement constraint and the batching constraint, the optimum solution that minimizes total tardiness is J3-J1-J2. The completion times are 7, 12, 19 and the total tardiness is **1.** If we ignore the tool saving constraint and consider 6 tool slots, we require three batches one each for J3, J1 and J2. The actual completion times including the batch changeover times are 7, 15, 25. The due dates are 10, 12 and 18 and the total tardiness is **10**.

So, we consider 3 jobs.

(Refer Slide Time: 05:14)



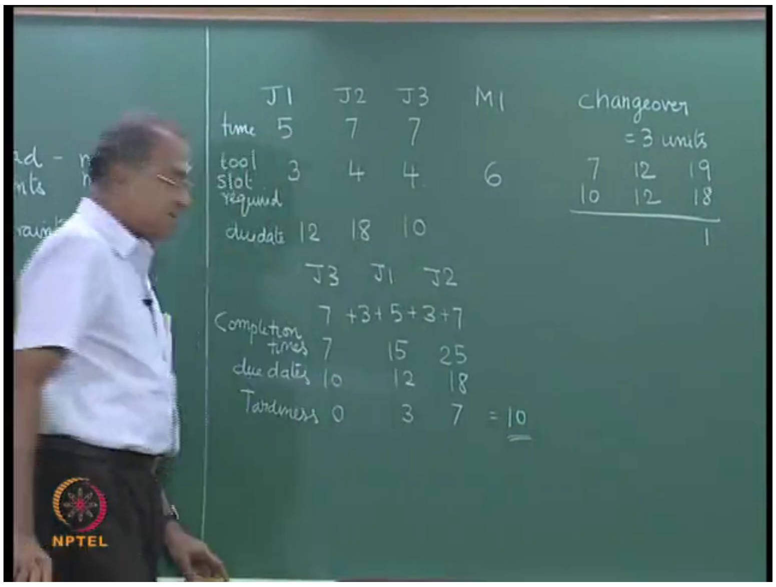
J 1, J 2 and J 3 and we consider a single machine which is M 1. So, there is no machine loading problem because all the jobs will go to machine M1. So, the processing times are 5, 7 and 7 and tool slot requirements are. So, tool slot requirement 3, 4 and 4 and tool slot availability is 6; the due dates are 12, 18 and 10.

Now, let us look at this issue, now there are 3 jobs there is a single machine the tool slot requirements are 3 4 and 4. So, we can assign only one job at a time, because if we assign 1 and 2 the requirement is 7 against capacity of 6, similarly if we assign 2 and 3 the requirement is 8 against capacity of 6 and 1 and 3 also the requirement is 7. So, at present we are not considering tool slot saving constraints. So, we will be able to do only one job, at a time and we need 3 batches each batch having one particular job.

Now, we have also said that changeover is 3 time units, now the due dates are 12 10 and 18. So, if we arrange them based on earliest due date, then we will do J 3 first then we will do J 1 and then we will do J 2. Now J 3 will be over at time equal to 7, because it takes 7 units of time 4 tools are inserted into the tool drum having 6 tool slots. Now once J 3 is over they have to change the tools to do J 1. So, that is going to take plus 3 units of change over and then we process J 1. So, 3 slots 3 tools are put in. So, we have another 5 units here. So, J 3 finishes at times 7, J 1 finishes at time 15, and then between J 1 and J 2 there is a changeover. So, plus another 3 18 and J 2 requires another 7 time units, 18 plus 7 25. So, these are the completion times.

Now, the due dates are 12 18 and 25; 7, 1, 18 and 25. So, now, this is tardy. So, tardiness will be 0 0 sorry tardiness will be the due dates are.

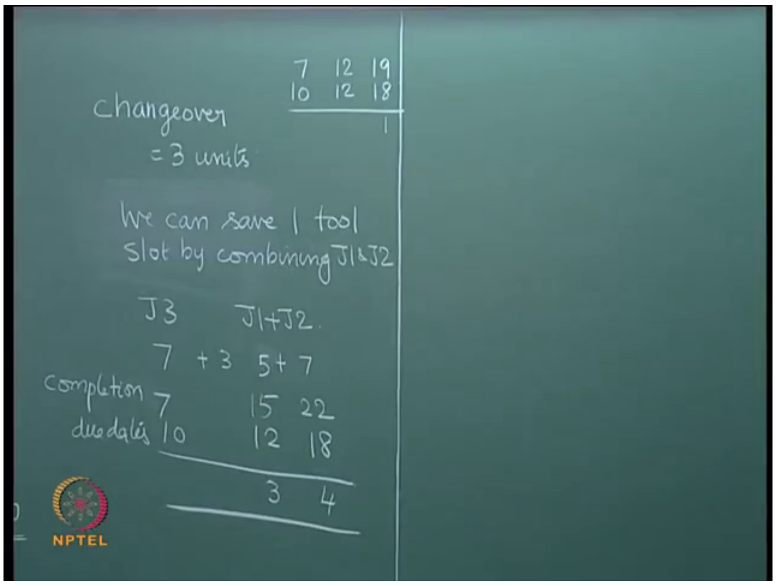
(Refer Slide Time: 09:23)



Due date for J 3 is 10, due date for J 1 is 12 and due date for J 2 is 18. So, we will have. So, now, tardiness will be this is ahead of the due date. So, tardiness is 0 tardiness, this is after the due date. So, tardiness is 3 and tardiness is 7. So, total tardiness is 10. If we ignore this change over time, see if we do not have any change over time then the completion times will be 7, 7 plus 5 12, 12 plus 7 19 will be the completion times. The due dates are 10 12 and 18. So, this is the only one with tardiness total tardiness is equal to 1, but if we include the change over time the total tardiness is 10.

Now, if we consider tool slot saving and if we say that jobs 1 and 2 are combined, we can save one tool slot.

(Refer Slide Time: 11:09)



7 12 19
10 12 18
1

Changeover = 3 units

We can save 1 tool slot by combining J1 & J2

J3 J1+J2.

7 + 3 5+ 7

completion 7 15 22

due dates 10 12 18

3 4

So, we can save one tool slot by combining J 1 and J 2. Now what we can do is since we can save 1 tool slot by combining J 1 and J 2, we now combine J 1 and J 2 and together when we combine we require 3 plus 4 minus 1 6 tool slots. So, it is enough to run 2 batches. So, first batch will be J 3, and the next batch will be J 1 plus J 2.

Now, we start J 3 at time equal to 7 J 3 finishes at time equal to 7 now there is a single changeover plus 3. So, this is 7 there is a single changeover plus 3. Now J 1 and J 2 together are going to be done in the second batch, which means the tools for J 1 and J 2 are going to be put into the tool drum together and between J 1 and J 2 the order of due date is J 1 is done first and then J 2. So, 7 plus 3 10, plus 7 17 plus sorry plus 5 plus 7. Now the tool drum has the tools for J 1 and J 2, there is no change over between J 1 and J 2. So, between the 2 jobs J 1 and J 2 we again arrange them based on early as due date, do J 1 first and then J 2. So, J 1 comes first finishes the processing it leaves at time 10 plus 5 15 and then J 2 is immediately taken up. So, it takes another 7 units at 22 units J 2 is completed. So, these are the completion times, and these the due dates are for J 3 due date is 10 12 and 18. So, this completes ahead of the due date, here the tardiness is 3 here the tardiness is 4. So, total tardiness is 7.

Now, we are able to save those 3 time units which happens to be this changeover of 3. By reducing the number of batches we have reduced 1 change over which is 3 units and we are able to save this, because this changeover which we had earlier is removed. So, plus 3 7 would have become 10 here; we save 3 units here. So, total tardiness is this; if there is no change over time then the completion times again will become 7 plus 5 12 plus 7 19 there is no effect of this the due dates are 10 12 and 18 tardiness is 1. Only when there is a change over the batching problem becomes significant or the problem of assigning jobs to batches becomes significant, only if there is a positive change over time.

So, we can now see the effect of tool slot saving particularly the effect of tool slot saving we can see the effect of batching and we can see the effect of tardiness through a simple example as to how tool slot saving helps in reducing or minimizing the tardiness, particularly in the context of multiple batches. It is not uncommon to have multiple batches because these machines are quite expensive and we will not have too many of these machines, there could be 1 or 2 or 3 machines in a group while the jobs that require them will be very large, because they are special purpose machines capable of doing a variety of operations precisely and very well, demand on these machines will be very high. So, the number of jobs that we will be doing will be much more and they will require a large number of tools, and then the batching problem will also come into play. In fact, the number of individual tools can also become a constraint at some point.

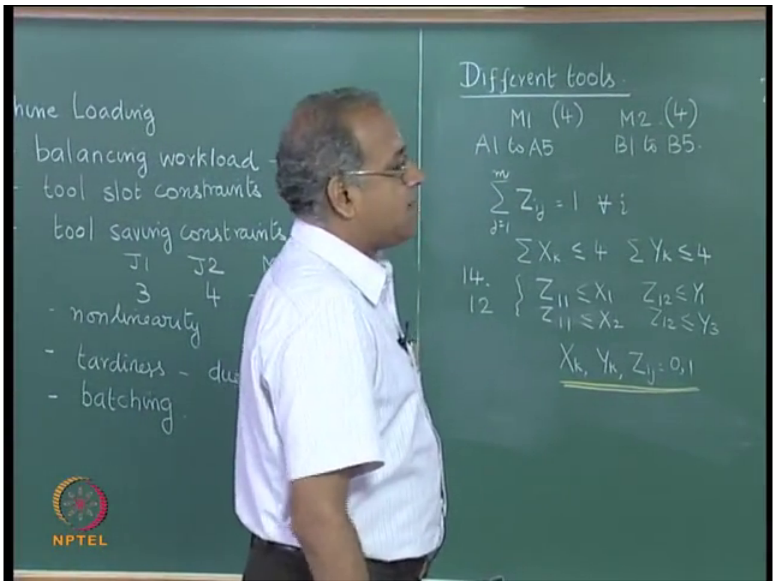
So, we do have its not only enough to look at a machine loading problem to balance work load, we should also look at the batching problem to assign jobs to different batches on the machine and then we should incorporate the change over time, and then we should exploit the tool changeover and then build a sequence such that the final objective of minimizing tardiness is achieved. Now all these are shown through a very simple numerical illustration here, but then the optimization problem becomes extremely complex because of the nature of decision variables.

Now, we have decision variables of the type x i j k where job I goes to machine j in the kth batch. And then within that there will be a t ij k or c i j k which will be the completion time or s i j k which will be the start time. So, the optimization problem becomes complex and as mentioned the moment we bring in the tool slot saving constraint the nonlinearity creeps in and it becomes very difficult to solve as a integer programming problem or a binary I p because it is non-linear. So, it becomes a binary non-linear I p. Linearizing this non-linear is possible, but not easy. So, it we have to add more constraints to solve it. So, the only way we could look at solving this problem optimally is by trying a kind of a branch and bound algorithm, but if the objective is to minimize tardiness, it also becomes difficult because branch and bound algorithms though they exist for the purpose of minimizing tardiness, they or they have been approached with very minimal success in terms of number of jobs that they are capable of handling, because lower bounds for tardiness are not very tight.

So, this problem is indeed a very complex problem, if we increase the number of jobs if we increase the number of machines and also if we include the tool slot constraints. There are other simple observations such that, if the tool slot constraint is very loose in the sense that the tool slot capacity is large then the problem the branch and bound algorithms take lesser time and if the tool slot constraints are tight, then the branch and bound algorithm takes more time. Which means the moment the effect of tool slot saving has an impact on the solution it takes a lot more time to solve. So, this is how we understand the effect of all of these particularly on the batching loading scheduling problem to minimize the makespan as well as to minimize tardiness

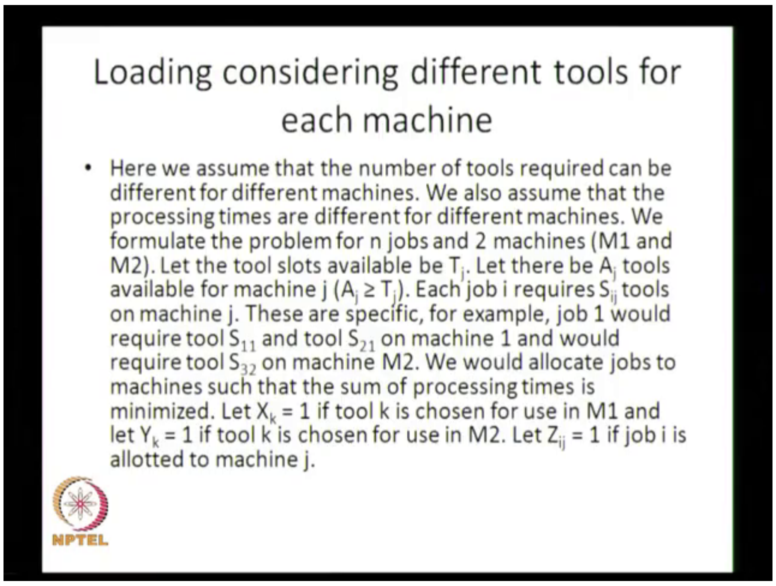
Now, let us look at one more problem, which is considering different tools for each machine now let us look at this through another example considering different tools for the machines.

(Refer Slide Time: 19:58)



So, here we assume that the number of tools are tools required are different for different machines.

(Refer Slide Time: 20:20)



Loading considering different tools for each machine

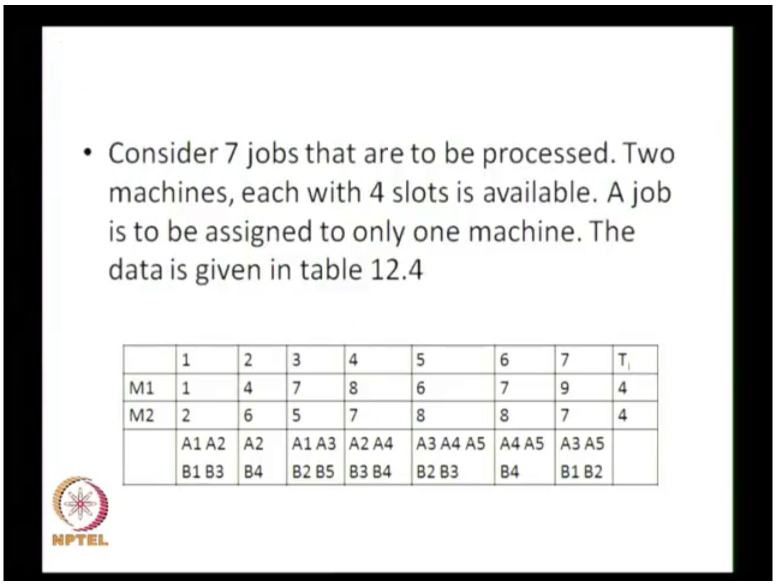
- Here we assume that the number of tools required can be different for different machines. We also assume that the processing times are different for different machines. We formulate the problem for n jobs and 2 machines (M1 and M2). Let the tool slots available be T_j. Let there be A_j tools available for machine j ($A_j \geq T_j$). Each job i requires S_{ij} tools on machine j. These are specific, for example, job 1 would require tool S_{11} and tool S_{21} on machine 1 and would require tool S_{32} on machine M2. We would allocate jobs to machines such that the sum of processing times is minimized. Let $X_k = 1$ if tool k is chosen for use in M1 and let $Y_k = 1$ if tool k is chosen for use in M2. Let $Z_{ij} = 1$ if job i is allotted to machine j.

For example if there are 2 machines M 1 and M 2 we assume for a moment that the tools are not interchangeable, which means M 1 has its own set of tools and M 2 has its own set of tools.

Now, this is also a practical issue as well as a practical constraint, because sometimes when organizations build flexible manufacturing systems, they buy machines from the same of the same make. Now when they buy similar machines of the same make, then the tools are interchangeable, but if there are 2 machines each is of a different make from a different manufacturer, then the tools are not interchangeable. In such instances it becomes a little more difficult because the programs also have to be compatible with each other. Each machine will work on its own programs developed. So, it could be a slight advantage to have machines from the same manufacturer, because of substitutability of tools because of good control over the programs. Sometimes cost considerations force organizations to buy machines from different manufacturers; therefore, each machine will have its own set of tools also the processing time will be different on each machine. This assumption was there even for the earlier problem because the loading problem without tool is like a makepan minimization on non-identical parallel processors.

Now, we formulate this problem for 2 machines, now let us look at this problem as such.

(Refer Slide Time: 22:14)



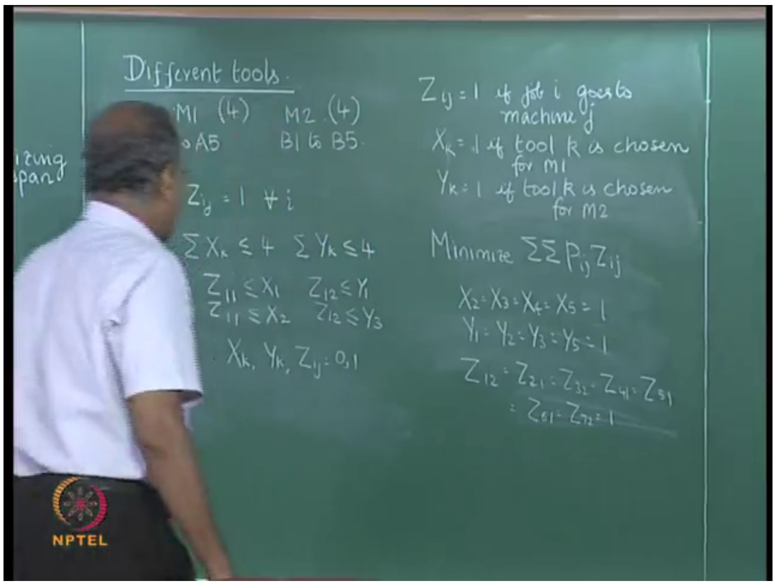
- Consider 7 jobs that are to be processed. Two machines, each with 4 slots is available. A job is to be assigned to only one machine. The data is given in table 12.4

	1	2	3	4	5	6	7	T_i
M1	1	4	7	8	6	7	9	4
M2	2	6	5	7	8	8	7	4
	A1 A2 B1 B3	A2 B4	A1 A3 B2 B5	A2 A4 B3 B4	A3 A4 A5 B2 B3	A4 A5 B4	A3 A5 B1 B2	

Now, this is the problem that we are going to solve, we will do the formulation for this particular example. So, let us assume that 1 has tools A 1 to A 5 a 5 and machine 2 has tools B 1 to b 5 now there are 7 jobs the processing times are given in the table, on 2 machines. Now M 1 has 4 tool slots M 2 has 4 tool slots. Now let us explain this A 1, A 2, B 1, B 3 etcetera. So, if J 1 job J 1 or job 1 is assigned to machine 1, then we have to assign 2 tools a o and A 2 to machine 1 if job 1 goes to machine to then 2 tools B 1 and B 3 will have to go to machine 2. So, that is the way we describe if job 5 is assigned to machine 1, then machine 1 would require tools A 3, A 4 and a 5 it requires 3 tools; obviously, there is a tool saving slot constraint because if J 1 and J 2 are both assigned to M 1, then it is enough to have only 2 tools A 1 and A 2. The difference here between this example and the earlier example is, here we are specifying the tools we the number the tools each tool has a unique identity. In the earlier example we said it requires a certain number of tools and some number of tools can be shared. So, we bring in one more level of detail into this problem

Now, we want to balance the workload.

(Refer Slide Time: 24:28)



Different tools.

M1 (4) M2 (4)

A5 B1 to B5.

$Z_{ij} = 1 \ \forall i$

$\sum X_k \leq 4 \quad \sum Y_k \leq 4$

$Z_{11} \leq X_1 \quad Z_{12} \leq Y_1$

$Z_{11} \leq X_2 \quad Z_{12} \leq Y_3$

$X_k, Y_k, Z_{ij} = 0, 1$

$Z_{ij} = 1$ if job i goes to machine j

$X_k = 1$ if tool k is chosen for M1

$Y_k = 1$ if tool k is chosen for M2

Minimize $\sum\sum P_{ij} Z_{ij}$

$X_2 = X_3 = X_4 = X_5 = 1$

$Y_1 = Y_2 = Y_3 = Y_5 = 1$

$Z_{12} = Z_{21} = Z_{32} = Z_{41} = Z_{51} = Z_{61} = Z_{72} = 1$

So, we will now say that let Z i j equal to 1 if job i goes to machine j. X i equal to 1 if to lie is chosen or X k equal to 1 you could use any notation. So, here I am using a notation X k equal to 1 if tool k is chosen for M 1 and Y k equal to 1 if tool k is chosen for M 2. So, we are the first constraint that each job is assigned to only one machine. So, sigma Z i j, j equal to 1 to m machines is equal to 1 or every I for every job each job is assigned to only one machine.

Now, sigma X k is less than or equal to t in this case 4, where a maximum of 4 tools can go to machine 1. Similarly sigma Y k is less than or equal to 4, because machine to also has a capacity of 4. Now Z 1 1 if job 1 is assigned to machine 1, then from this table we understand that x 1 and x 2 should both be there. So, Z 11 should be less than or equal to X 1 and 0 1 1 should be less than or equal to x 2 for example, I cannot have a situation where X 2 is 0 which means tool A 2 is not assigned to machine 1, if that happens then Z 1 1 I cannot assign I cannot assign job 1 to machine 1.

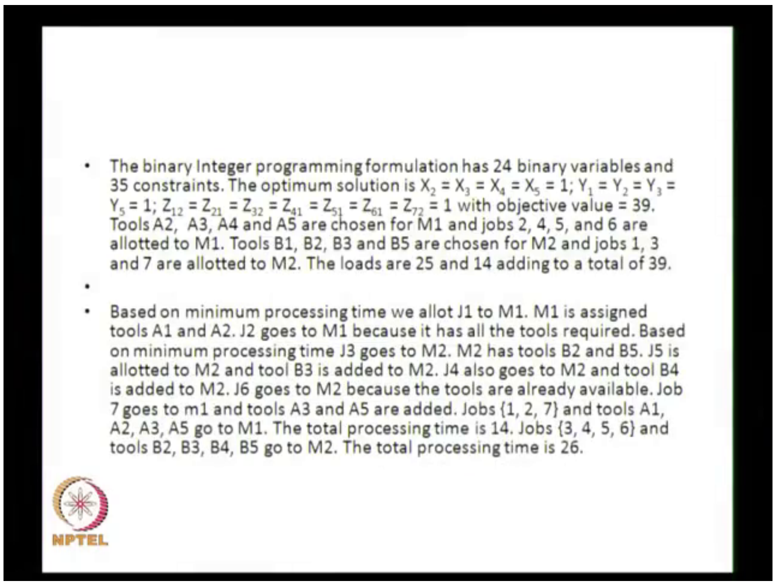
Now, once we write these 2 the rest of them are obvious. So, this will give us Z 1 2 is less than or equal to Y 1, Z 1 2 is less than or equal to Y 3 because tools B 1 and B 3 are required if we have to do job 1 on machine 2. So, we have one constraint for each of these tools that we have there. So, we are going to have 2 plus 1 3 5 7 10, 12 14 such constraints 14 such constraints considering all the allocations for example, the next one will be Z 2 1 is less than or equal to x 2 and Z 2 2 is less than or equal to Y 4 like this we have 14 constraints for the first machine and another 14 constraints for the second machine as you have 2 plus 1 3 5 7 9 10 12 2 3 5 7 9 10 12 constraints for machine 2 and 14 constraints for machine 1. 2 3 5 7 10 14 for machine 1 and 12 for machine 2 of course, X k, Y k, Z i j are all binary.

Now, we could either balance the workload or we could minimize the total sum of processing. So, we would go back and do minimize P i j Z i j. Now this would minimize the total time of processing, some of the times of processing on both the machines. We can also balance are workload by minimizing the maximum of the loads, by defining the maximum of the load as u and u greater than or equal to P i j X i j for all j. So, we can do that.

So, this is the formulation the nice thing about this formulation is that, the tool saving tool slot saving constraint becomes implicit it is not non-linear, but it takes care of the tool slot saving constraint. If we see carefully it is not non-linear, but as I said if we assign job jobs 1 and 2 to machine 1, we would actually be utilizing only 2 slots and not 3. Because job 1 on machine 1 would require tools 1 and 2 A 1 and A 2, job 2 on machine 1 requires only A 2. So, you could assign both of them to machine 1 by simply fixing tools A 1 and A 2.

So, tool slot saving constraint is taken care of, the moment we name the tools and say these are the tools that are required. Only when we do not name the tool we get into non nonlinearity coming in when we define the saving as so much is saved if these 2 are put together. So, the problem continues to be linear and we solve this problem to get.

(Refer Slide Time: 31:04)

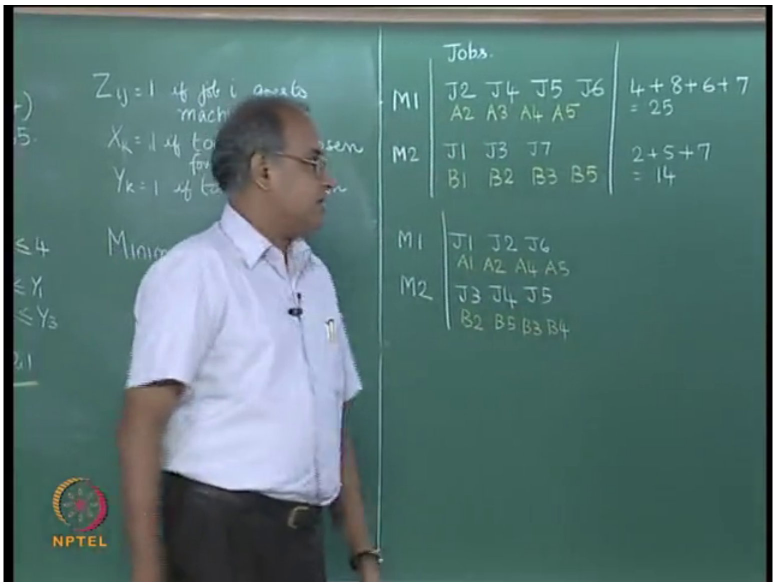


- The binary Integer programming formulation has 24 binary variables and 35 constraints. The optimum solution is $X_2 = X_3 = X_4 = X_5 = 1$; $Y_1 = Y_2 = Y_3 = Y_5 = 1$; $Z_{12} = Z_{21} = Z_{32} = Z_{41} = Z_{51} = Z_{61} = Z_{72} = 1$ with objective value = 39. Tools A2, A3, A4 and A5 are chosen for M1 and jobs 2, 4, 5, and 6 are allotted to M1. Tools B1, B2, B3 and B5 are chosen for M2 and jobs 1, 3 and 7 are allotted to M2. The loads are 25 and 14 adding to a total of 39.
-
- Based on minimum processing time we allot J1 to M1. M1 is assigned tools A1 and A2. J2 goes to M1 because it has all the tools required. Based on minimum processing time J3 goes to M2. M2 has tools B2 and B5. J5 is allotted to M2 and tool B3 is added to M2. J4 also goes to M2 and tool B4 is added to M2. J6 goes to M2 because the tools are already available. Job 7 goes to m1 and tools A3 and A5 are added. Jobs {1, 2, 7} and tools A1, A2, A3, A5 go to M1. The total processing time is 14. Jobs {3, 4, 5, 6} and tools B2, B3, B4, B5 go to M2. The total processing time is 26.

The binary integer has X 2, X 3, X 4 X 5 equal to 1. So, it has x 2 x 3 equal to 1 Y 1, Y 2 Y 3 Y 5 equal to 1 and it has Z 1 2, Z 2 1, Z 3 2, Z 1 2, Z 2 1, Z 3 2, 5 1 5 1 6 1 7 2 4 1 5 1 6 1 7 2 equal to 1.

So, now let us look at this solution now from this solution we assign.

(Refer Slide Time: 32:06)



This is M 1 this is M 2 this is jobs. So, let us go back to this job 1 goes to machine 2. So, job 1 goes to machine 2, job 2 goes to machine 1, job 3 goes to machine 2 job for goes to machine 1, job 5 goes to machine 1, job 6 goes to machine 1 and job 7 goes to machine 2 now from this X 2 X 3 X 4 X 5 means tools A 2, A 3, A 4 and A 5 go here Y1, Y 2,Y 3, Y 5 B 1, B 2, B 3 and B 5.

So, if you look at J 1 goes to M 2 it requires B 1 and B 3 B 1 and B 3 are here. J 2 goes to M 1 it requires A 2 A 2 is here. J 3 goes to M 2 it requires 2 and B 5 they are here, J 4 A 2 and A 4 are here J 5 A 3, A 4, A 5 are here for J 6 its A 4, A 5 they are here and for J 7 B 1, B 2 are here. So, all the 4 tool slots are utilized, but we have enough tools to carry out all these jobs now the times are for M 1 it is J 2 which is 4 J 4 which is another 8, J 5 is 6 and J 6 is 7. So, this is equal to 12 18 plus 7 25, 1 3 and 7, 2 plus 5 7 is 7 14.

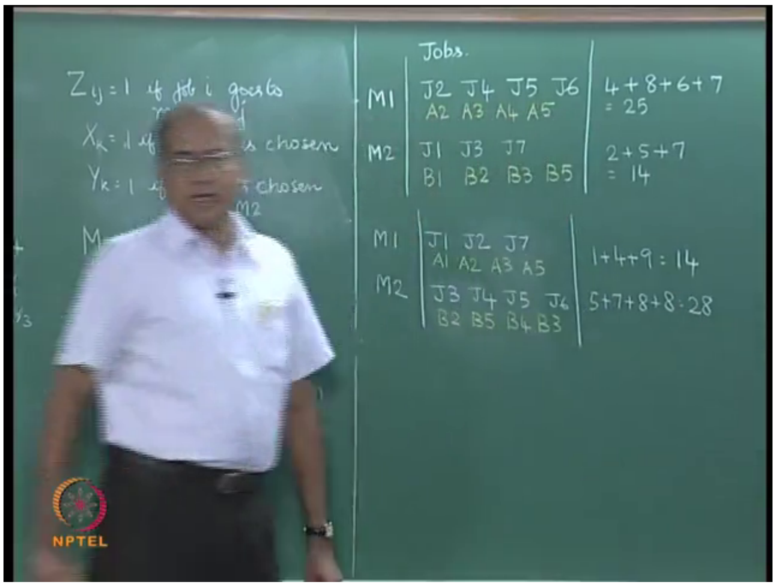
So, the loads are 25 and f14 with the total equal 39. We could solve another problem to balance our clods, we may get a slightly different solution also, but the total would be 39 or more in such a case. Now if you wish to balance then we would minimize u, u greater than or equal to P ij X i j for both the machines and then we will be able to balance the workload on we could use any of these objectives, but here we are not looking at tardiness we are not right now we are not defined due dates for these jobs and then we have not defined audience. If we define tardiness then we get into difficulty, and as I mentioned the decision variables will be based on time of completion. Right now it is only an allocation problem. So, the decision variables only represent the allocation of jobs and tools to various machines.

Now, it we bring in due dates and tardiness the problem becomes a little more complicated, it continues to be linear because we have the tool slot saving constrained as I mentioned is not non-linear in this case. Never there is the problem becomes involved and is hard because of the binary nature of the variables, now all these variables are 0 1 variables; because of the binary nature are the variables the problem becomes hard. So, we could think in terms of some heuristics.

Now, let us look at this based on minimum process in time we could think in terms of a heuristic like this. So, let us start assigning them. So, M 1, M 2 based on minimum processing time let us start assigning. So, M 1 goes to J, J 1 goes to M 1. So, along with J 1 tools A 1 A 2 go here J 2 also goes to M 1 tools are already available J 3 goes to M 2. So, tools are B 2 and B 5, J 4 goes 2 M 2 B 3 and B 4. So, all the 4 tools are exhausted tool slots are exhausted, J 5 on M 1 A 3 A 4 A 5 cannot include, there for J 5 B 2 B 3 are already here. So, J 5 goes here, J 6 A 4, A 5. So, we could look at J 6 and A 4, A 5, J 7 goes to B 2 B 1 B 2 we do not have, but A 3 a 5 also we do not have, because J 7 requires A 3 and A 5. So, we are not able to allocate J 7. So, let us see what happens here.

So, we change this now in this way we are we are unable to allocate. So, let us follow or slightly modify this heuristic to see what we do. So, we allocate J 1 to M 1. So, we allocate J 1 to M 1, tools A 1 and A 2 go to M 1. J 2 also goes to M 1 because the tool is already available and minimum processing time. So J 1 and J 2 go here.

(Refer Slide Time: 39:09)



Now, we look at J 3, J 3 goes to M 2 based on minimum processing time. So, J 3 comes here and tools B 2 and B 5 come here. Now J 4 goes to M 2 and B 4 is added. So, J 4 goes to M 2 tool B 4 is added because it requires no the heuristic says J 5 goes to J 5 is alerted to M 2 and B 3 is also alerted to M 2. So, J 5 is allotted to M 2. So, J 4 goes this now J 5 also J 5 cannot be put on M 1 because J 5 requires tools A 3 A 4 and A 5. So, we will exceed this. So, J 5 has to come here. So, J 5 requires tools B 2 and B 3. So, B 3 is added.

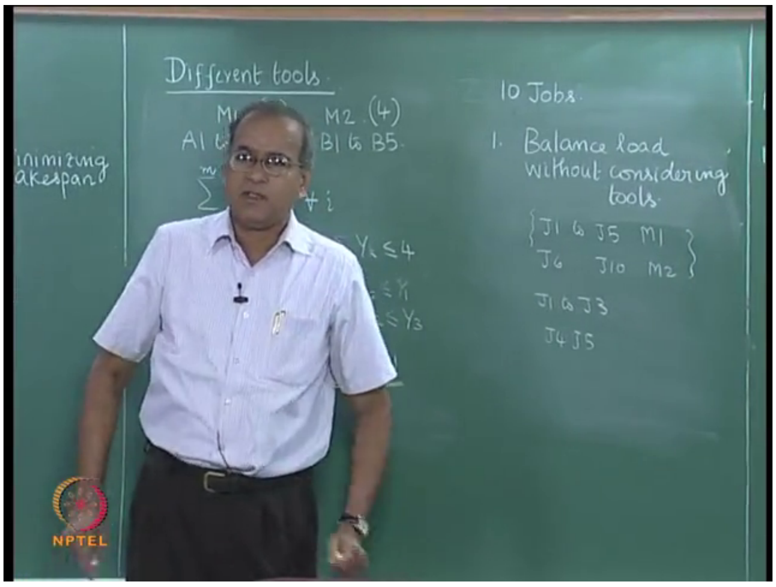
Now, J 6 requires B 4, but based on minimum processing time, J 6 has to go to M 1. So, now, we have a conflict because J 6 requires A 4 and A 5. So, we have enough slot to give you A 4 and A 5 here and put J 6 on the minimum load machine, but we know that if we do that we cannot accommodate J 7, so we have to accommodate J 7. So, we follow the other rule by saying that if the full set of tools are already there on one machine do not go by a minimum rule assign it there and then if it is not if you have to add something then go to the minimum rule. So, by that condition J 6 will come here, because be 4 which is the machine that it requires has B 4 is available here so.

Now, for J 7 we need A 3 and A 5. So, we put J 7 here, we need A 3 and A 5 now J 7 should ideally go to M 2 and requires B 1 and B 2 B 1 and B 2 are not here and we cannot add. So, we look at the other one. So, it requires A 3 and A 5. So, I put it here I add A 3 and A 5. Now what is the workload on this? The workload is 1 2 and 7 form 1 1 plus 4 5 plus 9 14, 2 3 4 5 and 6, 5 plus 7 12 plus 8 20 plus 8 28. 5 plus 7 plus 8 plus 8 28.

So, this heuristic algorithm gives us a solution like this, but this heuristic algorithm is based on some thumb rules, that if we pick a job and if all the tools required to do that job are already assigned to a particular machine, then we do not go by the minimum processing time rule, we assign the job to that machine. But if all the tools are not there on a particular machine and some tools have to be added, then we go by the least processing time and see if we can add the tools and put it there if there enough space otherwise we go to the next least processing time, and we keep proceeding using this algorithm till we are able to allocate all the jobs.

So, this is how we carry out the loading of jobs on machines, with specific tools required and tools large capabilities on each of these machines. We have not addressed the batching problem here, now have we addressed the tardiness and scheduling problem here. We took a problem where with 4 tools we are able to allocate all of them. If we had more jobs say we had 15 jobs or 10 jobs, then we will not be able to do it with tools large capacity of 4. So, it would require multiple batches. In such case what we will do is we will temporarily leave out the tool constraint we will simply solve for example, if instead of these 7 jobs.

(Refer Slide Time: 45:14)



If we had 10 jobs that have to be allocated to these 2 machines each of these 10 jobs has a specific set of tool requirement like A1, A 2 etcetera.

So, now we will we know for sure that we will require multiple batches to operate on this. So, phase one problem will be to simply balance workload without considering 2 slots. So, this problem will be similar 2 non identical parallel processor makespan minimization on a single machine, because each job will require different processing in times on the machines. So, parallel processors are non-identical. Now tool slots are not considered then for example, if we have a solution which is jobs J 1 to J 5 will go to M 1 and let us say jobs J 6 to J 10 will go to M 2 suppose this is the solution, then we have to solve this problem only considering tools such that we now define them in 2 batches, and then we say that out of these 1 to 5 we may consider 2 batches or 3 batches and then say this job will go in this batch.

And then if the set of jobs go into this batch, then the union set of tools should be we should be able to accommodate the union set of tools into the tool slot. So, the processing time does not come into the picture, we simply maximize the number of jobs that can go to the first batch. So, maximize number of jobs that go to the first batch such that the tool constraint is satisfied. Similarly maximize the number of jobs that go to the second batch, such that the tool constraint is satisfied. Then remove those jobs and then from the remaining jobs solve this problem again, to maximize the number of jobs that go to the second batch and so on or they can combine the batches and solve such that we maximize the total number of jobs that go to the first batch, second batch, third batch and so on.

We can actually combine all of them and bring them into one large optimization problem where we can combine the batching problem, maximize the number of jobs that can be done on each batch. Then we get into the issue of scheduling if each of these jobs has a due date. Sometimes we put a due date constraint the due date will force a particular job to go into the first batch because of due date considerations. In such cases we can either define completion time for each job and do it or if that is a little difficult we can define completion time at an aggregate level for example, if in this 3 we divide them into 2 batches and if let us say J 1 to J 3 go to the first batch, J 4 J 5 go to the second batch now the sum of the time when this batch is completed, the sum is the processing time of J 1 plus J 2 plus J3.

Now, we may make an assumption that all the jobs actually come out at the same time and not immediately after they are processed. If so, then the sum of these should be within the unit. The only difficulty there is in practice if all 3 go in one batch, depending on the sequence they can actually come out sooner than the batch is complete. The jobs within a batch they do not come out at the same time, the jobs can they come out as and when the processing is over. Only the tools are available for a change after the entire batch is completed.

So, if we really want to minimize the tardiness, then we have to define the variables in terms of completion time of the job on that machine. So, it becomes a fairly complicated problem, that involves loading and involves batching it also involves scheduling and sequencing. The nice thing about this formulation is that it does not explicitly have the nonlinearity component which the earlier formulation had and this is also practical in the sense, that the machines could be of different makes and therefore, they could have different types of tools that go in and the tools are now numbered and named saying if this job is to be done, then these specific tools are required. Further models on flexible manufacturing we will see in the next lecture.