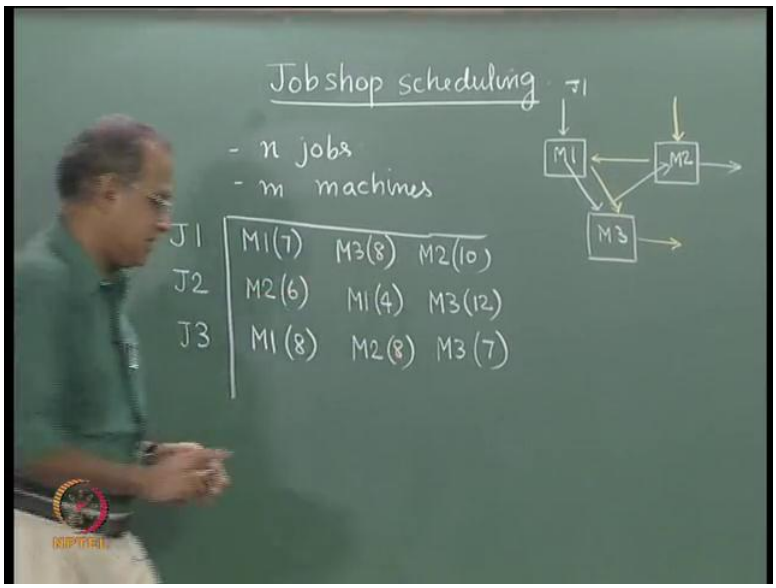
Operation and Supply Chain Management
Prof. G. Srinivisan
Department of Management Studies
Indian Institute of Technology, Madras

Lecture – 28
Job Shop Scheduling - Gantt Chart, Different Dispatching Rules

(Refer Slide Time: 00:25)



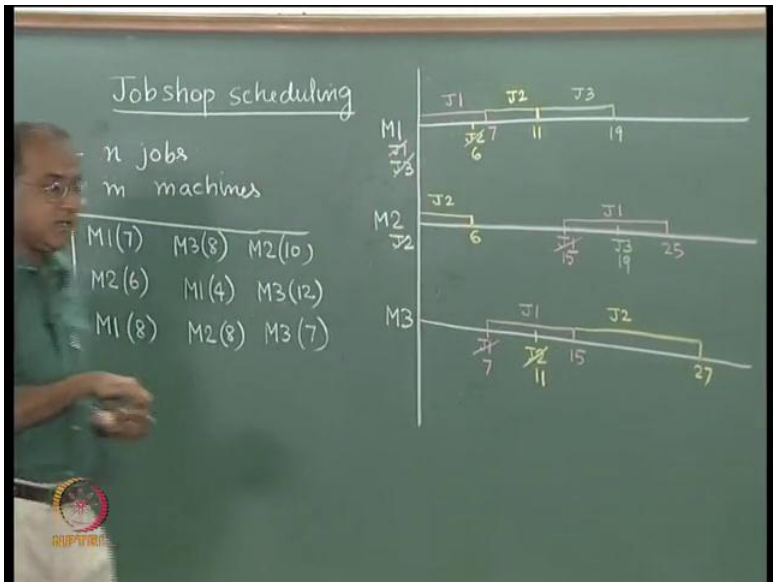
In the last lecture, we saw some heuristics for the floor shop scheduling problem. In this lecture we will address the job shop scheduling problem. So, as we mentioned earlier, in a job shop there are several jobs, let us say there are n jobs, and there are m machines. Unlike a floor shop each job has a unique route or a unique order of visit of the machines. For example, if there are 3 machines one particular job may come here, first then visit this, then visit this and leave, while some other job may start with this, visit this come back to this and then leave.

So, in a job shop each job has a pre specified route or order of visit of the machine, it is also not absolutely necessary that all the jobs will visit all the machines. A job can visit a subset of the existing set of machines. So, let us explain the job shop scheduling problem using an example. So in this example, we are going to look at three jobs and three machines. So, let me call the three jobs as J 1, J 2 and J 3, so let us consider this particular example, where there are three jobs and three machines. Each job has it is own pre specified route, which means, job 1 will first will visit machine 1, then visit machine

3, and then visit machine 2. So, like job one first visits machine 1 then visits machine 3 then visits machine 2 and goes out, another job J 2 may first visit machine 2, and then machine 1, and then machine 3.

For example, J 2 will first visit machine 2, and then it visits machine 1. What I have shown in yellow essentially stands for J 2, first visits machine 2 then visits machine 1, then visits machine 3 and then leaves. Now, the number shown in the brackets are the processing times for the particular job on the particular machine. Now, let us try and get a make span for a particular schedule, and let us draw a schedule to solve this particular problem.

(Refer Slide Time: 04:00)



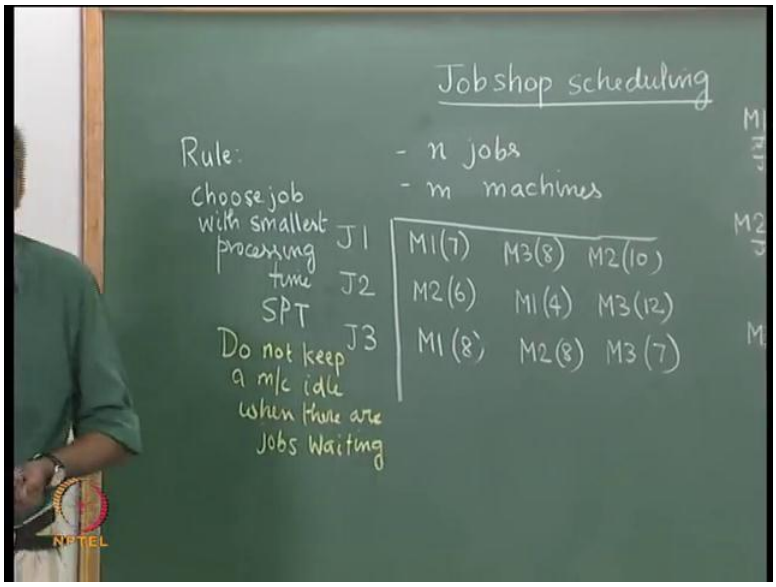
So, what we do first is we make a chart where, we are now going to show all the 3 machines on this chart. So, let us say this is machine M 1, this is machine M 2, and this would be machine M 3 on this chart. Now what we do is we let each job go and stand in front of it is first machine. So, J 1 will go and stand in front of M 1, so J 1 will be here. J 2 will go and stand in front of M 2, so J 2 will be here. J 3 also will go and stand in front of M 1. J 3 will be here.

Now, we are at time equal to 0. All the jobs are available and each job is now waiting or standing in front of it is first machine to be processed. So, if we look at machine M 2 at time equal to 0, job J 2 is standing in front of M 2 there is only one job waiting in front of M 2. So, job J 2 can be loaded on M 2, so it starts processing on M 2 at time equal to

0, finishes at time equal to 6 because it requires 6 time units, and at time equal to 6 it will leave M 2 and come to M 1 for further processing, which we show here as J 2 coming here at time equal to 6.

Now, at time equal to 0, M 3 does not have any job in front of it, so M 3 will remain idle, now we have to look at M 1 there are two jobs waiting in front of M 1 at time equal to 0. We have to choose only one of them for processing, because a machine can do only job at a time, now J 1 requires 7 units of time, J 3 requires 8 units of time. So, we would be inclined ordinarily to choose J 3 ahead of J 1 or J 1 ahead of J 3 because J 1 has lesser processing time compared to J 3.

(Refer Slide Time: 06:43)



So, we now follow a rule and say that rule choose job with Smallest Processing Time or use SPT rule. We have already seen the SPT rule. So, let us assume that if we have to choose between two jobs or among a set of jobs that are waiting, we would pick a job which has the smallest processing time. So, between J 1 and J 3 here J 1 requires 7 units of time, J 3 requires 8 units of time, so we choose J 1, so J 1 will start at 0 finish at time equal to 7.

And at time equal to 7, we will now move to M 3, so it will come here. So J 1 at time equal to 7. Now, we have to look at two issues at this point. Now in this particular example we had two jobs waiting J 1 and J 3, and we decided to choose one amongst the jobs waiting based on shortest processing time. So, when we apply this rule there is no

tie because J 1 requires 7 units, J 2 requires eight units clearly J 1 is chosen based on the rule.

But, if we had a situation where this was also a 7, then two jobs would have been waiting and both of them have the same processing time therefore, if we apply the rule there is a tie. So, if there is tie, we need a tie breaking rule also, and we need to define a tie breaking rule also. Now right now there is no tie, so the tie will happen when you apply your first rule, and after applying the first rule you still have more than one candidate. In which case we need to define a tie breaking rule and use the tie breaking rule to choose, the jobs that are in the tie.

Second issue is this, at here a t equal to 0 we realize that J 1 and J 3 are waiting, J 1 requires 7, J 3 requires 8. So, we picked J 1 ahead of J 3 at the same time we also have an additional information that J 2 is going to come at 6 here, but then one look at the table we realize that J 2 actually requires only four units of time. So, the question is will we leave out J 2 in the choice, and choose between J 1 and J 3 because they are waiting at time equal to 0, J 2 has lesser processing time. But, in order to take J 2 ahead of J 1 or J 3 the machine will have to be idle for some more time.

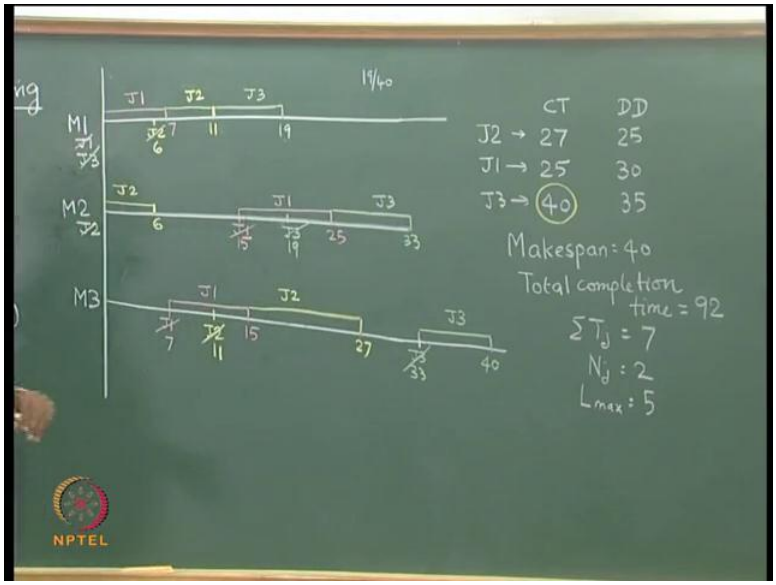
So, we always make a decision, but we will not keep a machine idle if there are jobs waiting in front of it. So, another rule is do not keep a machine idle when there are jobs waiting in front of it therefore, we will not consider that J 2 is going to come after 6 minutes when we make a decision here at time equal to 0. So, at time equal to 0 we will see what is available and then make a decision accordingly, now we go back and see that up to time equal to 6 or up to time equal to 7, machine 1 is busy. Machine 2 is busy up to time equal to 6, machine 3 does not have anything except that time equal to 7 it can start J 1.

So, the clock shifts to time equal to 6 now, so at time equal to 6. M 1 is busy at time equal to 6; M 2 is free, but there is no job waiting, at time equal to 6. M 3 is still idle, and there is no job in front of it. So, the clock shifts to time equal to 7 where on M 1 we have J 3 waiting we also have J 2 waiting, so J 3 requires 8 units of time, J 2 requires 4 units of time. So, based on the shortest processing time rule, we would choose J 2 ahead of J 3 even though J 3 came earlier, the rule that we use consistently is to choose the one with the smallest processing time.

So, we would chose J 2, so we start J 2 at 7 takes another 4 units, so at 11 J 2 is completed. J 2 will go to M 3 at time equal to 11. Now, once again at time equal to 7, M 2 does not have any job in front of it. Now at time equal to 7 M 3 has J 1 in front of it, so it starts J 1. So, J 1 on M 3 is another 8, so 7 plus 8 15, so J 1 finishes on M 3 at 15 and goes to M 2 at 15, so J 1 comes at 15.

Now, the clock moves to time equal to time equal to 11. Only J 3 is waiting in front of M 1. So, we start J 3, J 3 requires 8 units, so 11 plus 8, 19, so at time equal to 19 J 3 completes on M 1 and comes to M 2. So at 19 J 3 comes here, now the clock moves to time equal to 15 at time equal to 15, J 2 is available here. So, we take J 2 for processing J 2 on M 3 is another 12, so 15 plus 12, 27.

(Refer Slide Time: 14:46)



So, J 2 finishes at 27 on M 3, so J 2 completes at 27, so at time equal to 27 J 2 has finished it is last job on M 3. Now, once again we are at time equal to 15, J 1 is available here, and as I mentioned earlier we will not look at the presence of J 3 at 19 because we follow this rule that we will not keep a machine idle waiting for something to happen. So, at this point we will take J 1. So J 1 on M 2 is another 10, so 15 plus 10, 25. So J 1 has completed processing on M 2 at 25, so J 1 finishes at 25.

Now, once again we have J 3 has come here at 19, now we are at 25, J 3 is waiting on M 2 takes another 8, so 25 plus 8, 33. So J 3 finishes at 33 here. And then at 33, J 3 comes to, so J 3 comes at 33, so at 33 J 3 is the only 1 available here. It takes another 7 units of

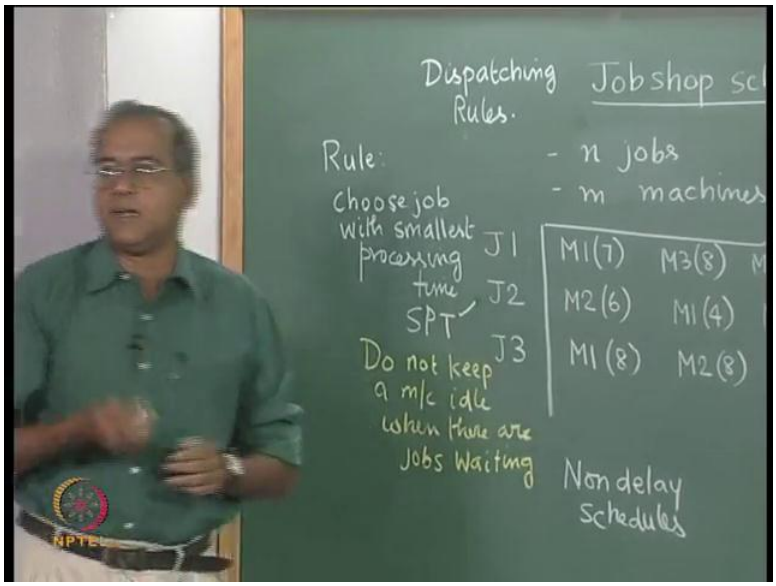
time. So, starts at 33 and finishes at 40. So J 3 finishes at 40, so all the jobs are completed at time equal to 40, so the make span for this fixed schedule is 40. So now, we can write that make span is 40, total completion time is 27 plus 25, 52 plus 40, 92.

Now, if each job has a due date and if the due date for the jobs are say 25, 30 and 35 then this is completion time. So, this is tardy with tardiness equal to 2. This is early this is tardy with tardiness equal to 5, so sigma T j total tardiness will be 7, number of tardy jobs N j will be 2. Because, this is tardy, this is also tardy, maximum tardiness which is usually called L max will be equal to 5 because the due date is 35 completion time is 40.

Now, if we want to see the utilization of these machines, so till time equal to 40 the shop is busy, so this takes 19 out of 40. So, utilization of M 1 can be shown as 19 by 40, utilization here would be the time taken on M 2 which is 10 plus 6, 16 plus 8, 24 by 40, the utilization on M 3 is 8 plus 12, 20 plus 7, 27 by 40. If we want the idle time, this is idle for 21 time units, this is idle for 16 time units, and this is idle for 13 time units.

So the moment we get a chart like this we will be able to evaluate all the objectives. The objectives that we have evaluated are make span, total completion time or mean completion time and for assumed due dates the total tardiness. We can also look at total earliness. Only this job is early, so earliness is 5, total tardiness is 7 number of tardy jobs 2, number of early job 1, maximum tardiness is 5, utilizations, idle times. So, all possible objectives we can evaluate once we draw this chart. Now this chart is called the Gantt chart named after Henry Gantt who first proposed a chart or a pictorial representation of a schedule.

(Refer Slide Time: 20:08)



So, the Gantt chart can be drawn for a given rule like the SPT rule. Now these rules on which we draw the Gantt chart are these rules, using which we will choose one job out of the available jobs for processing are called dispatching rules. It is very customary, and very commonly used practice that we use dispatching rules to try and draw the Gantt chart. Once the dispatching rule is decided, the Gantt chart will be the same if more than one person works on the Gantt chart.

So, we could have we first define a dispatching rule, and then if necessary we have to define a tie braking rule, as mentioned earlier. Right in the beginning if both J 1 and J 3 they had 7 and 8. If both of them had 7 and 7, then the shortest processing time rule would not be able to get a unique job from the jobs waiting, so there will be a tie. Now, those ties are broken by what are called tie breaking rules, a commonly used tie breaking rule could be first in first out or lowest subscript, in this case J 1 would be seen as a job with the lowest subscript and so on.

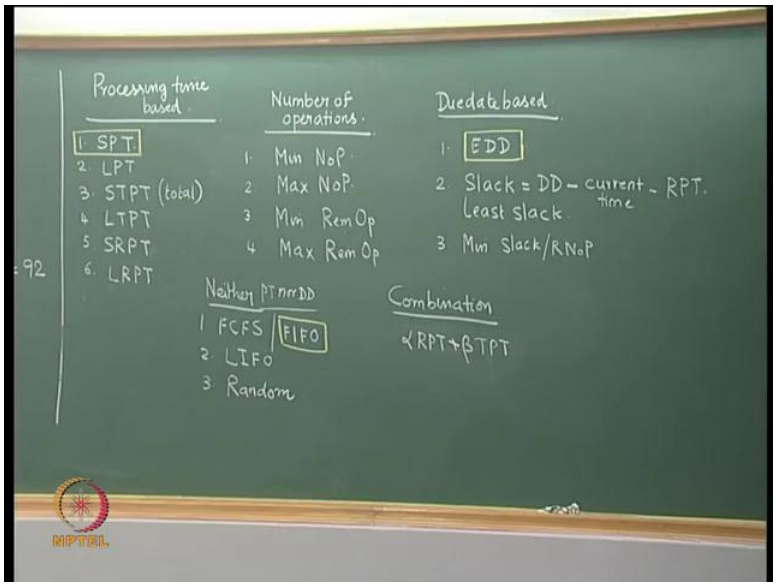
So, depending on the dispatching rule and the tie breaking rule, the Gantt chart is defined, and Gantt chart is an extremely efficient and intuitive way to draw a schedule. But, the Gantt this schedule need not be optimum with respect to make span or with respect to any other performance measure, Gantt chart is essentially an evaluative tool given a dispatching rule, we are now able to show pictorially how the schedules will look like, with absolutely no guarantee for optimality.

If the dispatching rule changes, then this Gantt chart will also change. We also look at when we used two things consistently. The first thing was the rule, which we now defined as a dispatching rule, the second principle that we used consistently was that we would not keep a machine idle, if there are jobs waiting front of it, with the hope that some other job that comes later if that is processed. We may be able to optimize the objectives. We will not follow that principle which we wrote here which we also used here.

For example, we did not consider this J 2 when we made a decision to choose J 1. Here, now schedules where this particular rule is used are called non delay schedules. We will not deliberately create a delay, so that we can optimize on the performance measure for that, and in general non delay schedules perform better than delay schedules where delays are intentionally introduced. From a practical point of view non delay schedules are important and are used.

Now, as mentioned earlier, depending on the dispatching rule the Gantt chart can be created, and there are several dispatching rules that are available which one can use to create Gantt charts. So, let us see some dispatching rules which are commonly practiced, we also have categories of dispatching rules.

(Refer Slide Time: 24:26)



So, the first categories are processing time based, now the rule that we used to create this Gantt chart is called shortest processing time rule. So, the first rule that is based on short

term processing time is the Shortest Processing Time rule also called SPT rule, now if we can draw a chart using this SPT rule, we can also draw a chart using the LPT rule. For example, we chose between J 1 and J 3 which was J 1 first here based on SPT, but based on LPT, we could have chosen J 3 and the schedule would have looked very different.

So, if SPT is a good rule LPT by itself is not a bad rule. If SPT will try and give a good make span, it does not mean that LPT will give very long make span, LPT in it is own way will also try to get a good value of the make span. So, SPT is also a rule that is often used, now we could consider instead of SPT we could have considered for example, we have to choose between J 1 and J 3, now let me look at what is the total processing time on J 1, 7 plus 8 15 plus 10, 25, 6 plus 4 10 plus 12, 22.

So, total processing time I am sorry 8 plus 8, 16 plus 7, 23, 8 plus 7, 15 plus 10, 25, so total processing times are 25 and 23. So, if we use what is called STPT Shortest value of Total Processing Time, we could have picked J 3 ahead of J 1, so STPT is another rule that we can think of shortest total processing time. Similarly, if STPT is a rule then LTPT can be another rule, choose the job which has the largest value of the total processing time.

Many times, we also do another thing for example, let us say we were here, so in this chart we are not able to encounter that situation except in the front. But, let us have a situation where, somewhere in the middle let us assume that we are in a particular machine. Let us say we are here, and a particular job has just finished processing, let us say J 2 has just finished processing here, now let us assume that J 1 has come at this point, and J 3 has come at this point.

Now, which means we have to apply a dispatching rule to choose between J 1 and J 3 to choose one of them. Now, let us assume that in this case J 1 has a total processing time of 25, but when J 1 has come here it has finished 7 and J 1 requires a remaining 18 processing time, including the machine that I have. Whereas, if I look at J 2 when it comes there, let us say this is machine M 2, now in J 1 has come to M 2, J 1 has already finished M 1.

And including M 2, J 1 has 18 units of time to go, when J 3 comes here on M 2. J 3 has a total of 23 it has already finished 16 it has 7 to go including the current processing time.

So, this has 7 to go including the current processing time, this has 18 to go including the current processing time, this has a total of 25, but let us assume it is waiting for M 3 after it has finished M 1. So, it has finished 7. So the remaining processing time for J 1 including the current machine is 18 whereas, when J 3 has come to M 3 it has already finished 16, so the remaining one including the current is 7.

So, you could choose a job based on Shortest Remaining Processing Time, where remaining includes the current processing time. So, you could have SRPT and you could have LRPT Longest Remaining Processing Time where remaining includes the current processing. So, these are some examples of processing time based dispatching rules, now other rules based on number of operations, in this particular example we have chosen three jobs, and all three jobs visited all three machines.

It is not necessary for all the jobs to visit all the machines, now suppose we had a situation where this one was not there. Then J 1 visits three machines, J 3 visits two machines. So, J 1 requires three operations, J 3 requires two operations, so one could choose J 3 based on minimum total operations, so you could have minimum total operations minimum N o P Minimum Number of operations, you could use maximum number of operations as two different dispatching rules.

Similarly, we could use minimum number of remaining operations for example, when we went back to the earlier situation. When we are looking at visit of say J 1 and J 3 on M 3, when J 1 visits M 3 it requires two more operations and whereas, when J 3 visits M 3 it requires one more operation. So, you could look at minimum remaining number of operations, and maximum remaining number of operations these are not processing time based, but these are number of operations based methods.

Then we can look at due date based methods. One of course, is the earliest due date, the second is called slack. Slack equal to due date minus current time minus remaining processing time. For example, write here to choose between J 1 and J 3 current time is 0, due date for J 1 is 30, so due date 30 minus current time 0, 30 minus remaining processing time is 7 plus 8, 15 plus 10, 25. So, slack is 5 for J 1, now for J 3 due date is 35 current time is 0, total time is 23, so slack is 12.

So, based on minimum slack J 1 can be taken, slack represents the extra time that is available, which acts like a slack or a buffer or a cushion to try and meet the due date.

So, that job which has the least slack will have to be taken first, so you could look at least slack rule, another commonly used rule is called minimum slack by remaining number of operations, I have a slack of 5 I may have only one more operation to go. I may have another situation where I have a slack of 8, but I have two more operations to go. So, 8 by 4 will be 2 or 8 by 2 will be 4 versus 5, so the other one can be taken.

So, we compute the slack and then we see how many more operations are to go, so divide the slack by remaining number of operations, and then chose the minimum of slack by remaining number of operations. Then we have a set of rules, which are very common rules which are neither or none of the above one would say, neither processing time based nor due date based general rules. So, first is First Come First Served or First In First Out, they mean the same First In First Out, Last In First Out, the one that came first will be taken the one that is came last will be taken.

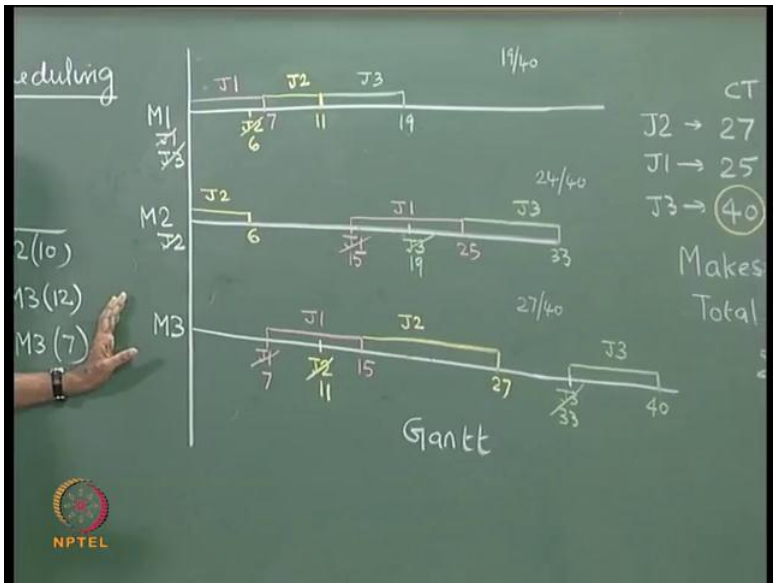
So, first in first out last in first out one could use some kind of a random rule, just pick the job randomly out of the jobs that are available. So, these are examples of popularly used dispatching rules. Sometimes we could have other combination rules where we could think in terms of some alpha into SPT plus some beta into LPT, where we could have weights, we could have weighted combinations and so on. We could have priorities we could have all of them.

So, a simple example would be, we could use alpha into remaining processing time plus beta into remaining number of operations or beta into total processing time that can be a combination rule. So, we could have rules like this. Now out of these, so many rules it is also common practice to essentially look at SPT, first in first out, and EDD to single them out as three extremely popular dispatching rules.

Now, these three rules are very common easily understandable, and easily implementable when we know that certain number of jobs are here, it is very intuitive that we choose the one which has the smallest processing time. Similarly, intuitively we choose the one that came in first, and if we have an idea of the due date we would intuitively choose the job or activity that has the earliest due date. So, when we really want to put this to use, the Gantt chart based schedule generation to be put to use in a floor shop or in a shop floor.

The most convenient thing to do is to use one of these three rules, and then draw the Gantt chart and make a schedule. So, if this schedule goes to an operator who is working on a machine, say let us say we are looking at the operator who is working on say M 3.

(Refer Slide Time: 38:41)



Now, based on this schedule we will tell the operator that at time equal to 7, J 1 will come put J 1 in, at time equal to 11, J 2 will come put J 2 as soon as you finish J 1. And then J 3 will come later and put it here. Now in this particular schedule there is no dispatching rule on this machine. Whereas, if you are looking at the operator on M 1, we have to tell the operator that this is your schedule, even though J 1 and J 3 are coming first you take J 1 based on SPT, then you will use J 2 even though it came later and then you use J 3.

That is one way of explaining it or giving this, the other is to say when you have to make a decision choose based on the one that has the shortest processing time. So, it is extremely easy for the operator to implement, in practice it there could be situations where, there can be slight delays. For example, J 1 that requires 7 units of time may not actually reach M 3 at 7, it may reach M 3 at 8 or little later than 7 and so on, but then the operator also will not be very specifically bound by these numbers.

But, the operator will be very specifically bound by the rule using which he or she can pick a job and put it on the machine. So, when we use very common rules like SPT or EDD or first in first out the, the Gantt chart becomes usable and any one can understand

the Gantt chart and appreciate the consistency with what is actually happening in practice. But when we make these then this conflict comes as to the next question is, we also mentioned that this particular Gantt chart based on the SPT rule, need not give us the optimum solution all the time.

Now, we have listed about 6 plus 4, 10, 13, 16, 17 rules. There are actually hundreds of rules that are there, and today with the amount of computerization and use of information technology devices, it is possible to write a program that will generate the Gantt chart. It is also possible for the user to define a dispatching rule, which is either one of those listed here or another dispatching role. And it is easy to get the corresponding Gantt chart very, very quickly.

Now the question is if SPT Gantt chart give us the solution with 40 and let us assume that we are using slack by remaining number of operations as another rule or we are using an 0.5 times RPT plus 0.5 times TPT as a dispatching rule, and we are able to get a solution with 38 say, make span of 38. Then from an optimization point of view 38 is better than 40, but from implementation point of view 40 would be a little easier to implement on the shop floor. Because, it is easy to understand and use, so the question is do we emphasize on advanced or more complicated or more involved rules.

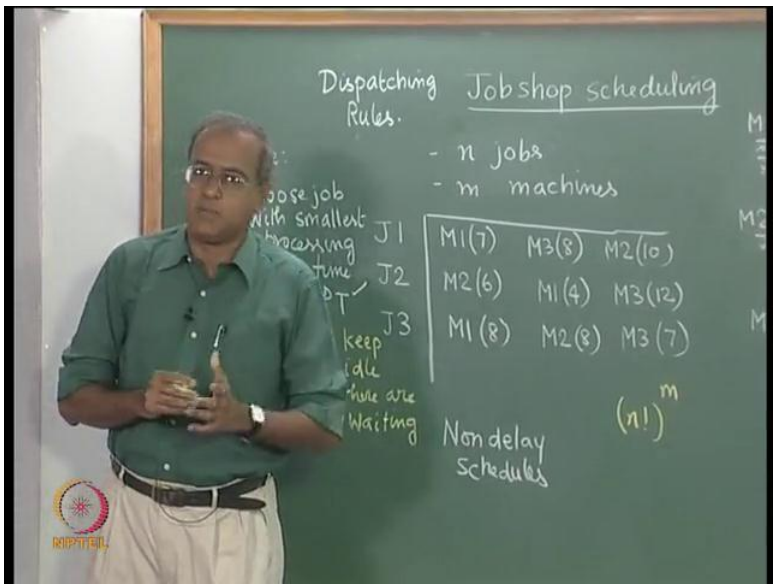
If they can provide better make span or are we content and happy with commonly used rules like SPT FIFO and EDD, because they are easy to implement and the person implementing will not make a mistake in implementation. So, the answer to that comes in the amount of training, and the amount of education that is provided to the people who do the job. The people who do the job can understand and appreciate, the role of these dispatching rules, and the fact that a slightly involved dispatching rule like minimum of slack over remaining number of operations, can give a solution which is lesser than this.

And if people are aware educated and are ready to use, then it will be an advantage to use involved dispatching rules, as long as the performance measure is optimized. But, if on the other hand the organization is comfortable with the use of simpler and well known dispatching rules like SPT or EDD or FIFO, which is more commonsensical than the organization could use, such rules and try and try and schedule the jobs in the job shop that is being looked at.

The other advantage as I have mentioned with the Gantt chart based scheduling, and the Gantt chart based dispatching rule is that these are evaluative in nature, with a single Gantt chart schedule, we can evaluate each and every one of the objectives. The positive thing is the efficient pictorial representation, easy to understand, ability to evaluate a variety of objectives. They are not, so positive thing it is the fact that, the it is simply dependent on a particular rule, and it does not depend on a particular objective function.

Therefore, while it has the ability to evaluate all objective functions, it perhaps also has the lack of ability to optimize a particular objective function. Because, it was based on a particular objective function therefore, the ability to guarantee optimality is also very less, and job shop scheduling is a very hard problem. If we look at now in this particular example, there were 3 machines and each machine has all the three jobs visiting, so every machine if all the n jobs visit every machine.

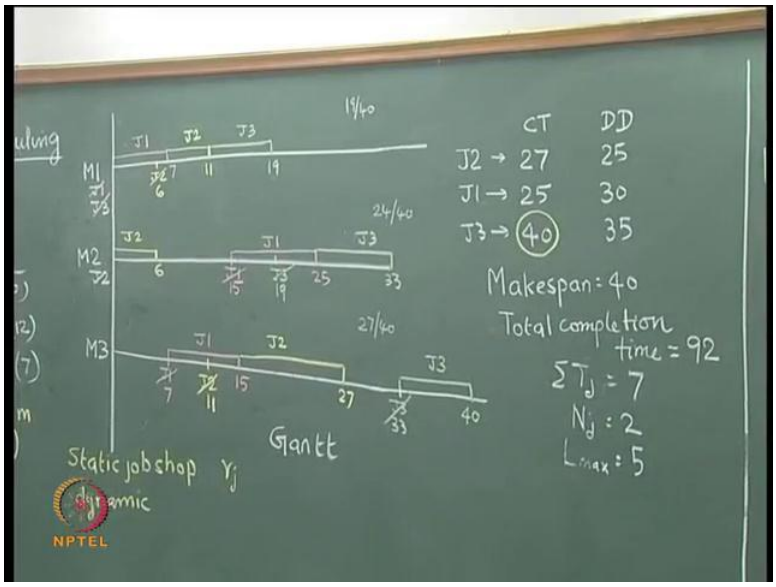
(Refer Slide Time: 45:06)



So, for each machine the n jobs can be arranged in n factorial ways, and since there are m machines we can have a total of n factorial to the power m possible alternatives in schedules. So, n factorial to the power m is a very large number and job shop scheduling problems are known to be very hard problems, so though dispatching rule is a very convenient way or method to solve job shop scheduling. It's inability to give optimal solutions is actually not very good.

But, over a period of time people have been using dispatching rules, to try and get Gantt charts. And with more and more computerization happening, and perhaps more and more dispatching rules available, one could program or use these dispatching rules into a software. And then try out several alternative dispatching rules, to try and optimize the performance measures, like a make span or a tardiness or whatever, and then use it accordingly by training the people.

(Refer Slide Time: 46:26)



Now, we have also looked at one aspect of job shop scheduling which is called a static job shop, a static job shop assumes that the number of jobs that we are going to do is known. So, in this example there are only three jobs, so the planning period or the planning horizon will end, as soon as all the three jobs are completed therefore, from a Gantt chart schedule it is possible to compute a make span, because the point at which all the three jobs are over will be the make span.

In real time job shops are not static in nature, jobs keep coming jobs keep arriving, so when jobs keep arriving then we can make certain variance to it. Now, and we also made an assumption here that all the jobs are available at time equal to 0, which is not a valid assumption in practice. So, in practice jobs keep coming dynamically, so we have to move from what are called static job shops to dyamic job shops, where the jobs can come at any time.

And the time at which the jobs are going to enter the system is not known a priori. When the time at which the jobs enter the system is known a priori, then it becomes static job shop with release times, which is usually called r j, where r j is the release time for job j. So in dynamic job shops, we do not know what time the jobs are going to be released into the system. So, in dynamic job shops there is also no end to the jobs coming in, so there is no make span concept in a dynamic job shop.

Dynamic job shops are essentially concerned with other measures like, flow time or completion time, tardiness, number of tardy jobs and so on. Dynamic job shops do not consider make span, we could also model job shops with the release times or no release times which is called r j. So, this gives us a complete view of certain aspects of what is called static job shop scheduling. Most static job shop scheduling is concerned with the use of dispatching rules, based on which we draw Gantt charts and then we evaluate all the objectives.

The limitation of course, is the lack of optimality or the inability to give the optimum solution to either make span or any objective for a static problem. Static problem implies number of jobs are known, the inability to give the optimum solution. Now, if we want to focus on the optimum solution, then we need to look at other methods such as mathematical programming and so on. And the problem also becomes very hard to try and get to the optimal solution.

So, the next question that comes is while dispatching rules are very good to try to give solutions, which are implementable, which are easy to understand. Are there other heuristic methods or other methods, which may not be based on dispatching rules, but can they also be used to try and get solutions, to the job shop scheduling problem. For example, can we have other heuristic methods other than dispatching rule based method, which may give us a value less than 40. One such method is called the shifting bottleneck heuristic, which follows an entirely different approach to solving the job shop scheduling problem. So, we will look at the shifting bottleneck heuristic in the next lecture.