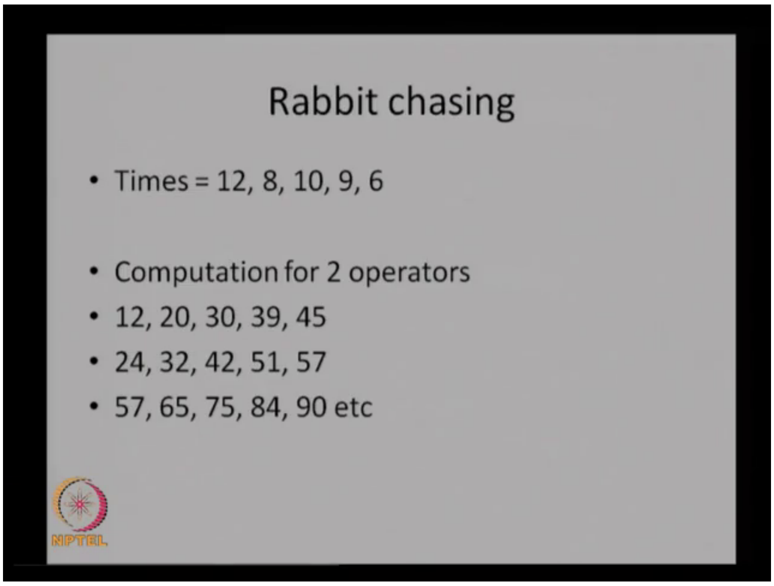
Manufacturing Systems Management
Prof. G. Srinivasan
Department of Management Studies
Indian Institute of Technology, Madras

Lecture - 20
Operator and task assignment

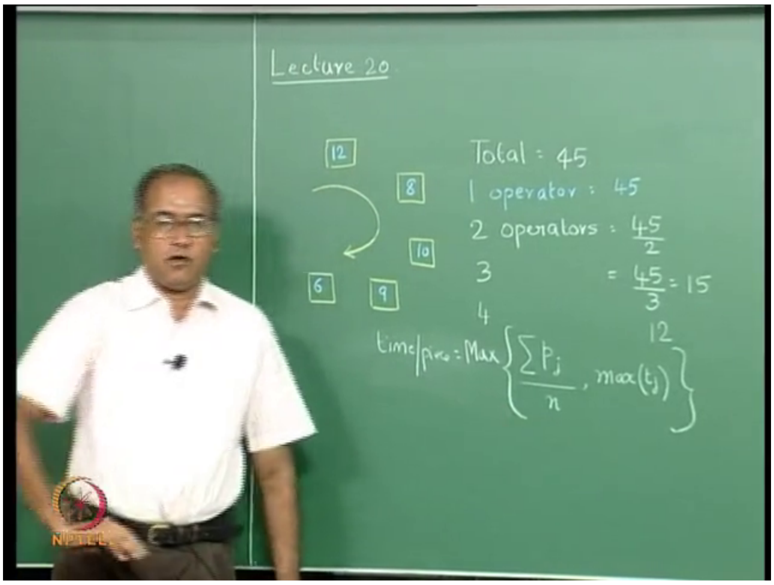
In this lecture we continue the discussion on Operator and Task Assignment.

(Refer Slide Time: 00:17)



In the previous lecture we saw some aspects of what is called rabbit chasing. So, we took an example, where there are 5 cells.

(Refer Slide Time: 00:28)



So, we took an example, where there were 5 machines in a cell, and we assumed that the cell would make only one product, and the processing time were 12 8 10 9 and 6; 12 8 10 9 and 6. Rabbit chasing assumes that if there is more than one operator. The first operator would come, take a piece, do some machining here, go to the next one, finish it, come to the third and then the fourth, and then the fifth, and then he or she leaves it at the end.

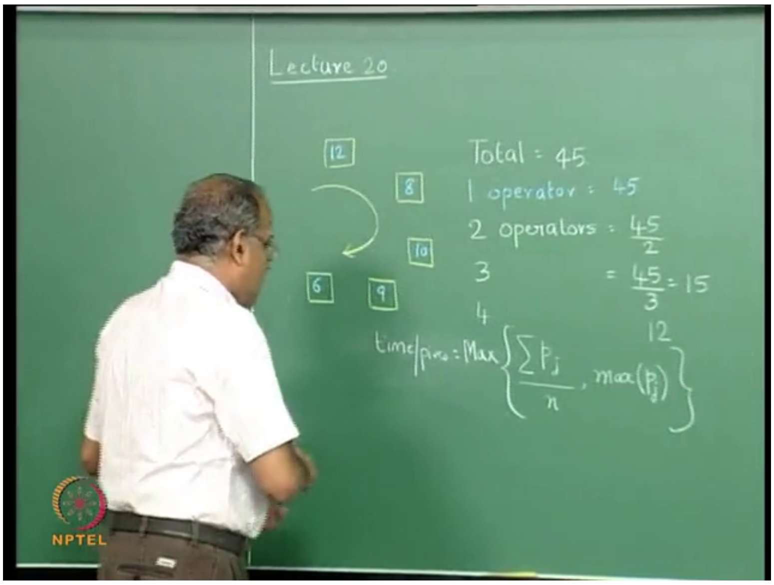
So, if there is only one operator, then assuming that these are, say seconds. Then if there is only one operator then 12 plus 8 20 30 39 45 seconds, or 45 time units, one piece will come out. The operator will go back and continue to do that. Now if there are 2 operators then, by the time the first operator finishes this and comes to this machine, the second operator would start here. So, this process continues and notionally they keep chasing each other, till all the production is complete. So, such a thing is called a rabbit chasing

In the previous lecture we saw that, in the total time in this case is 45. We saw that, if we use one operator in a rabbit chasing context, the time taken will be 45 units or time units per piece. If we use 2 operators, the steady state output in terms of time will be a piece every 45 by 2 or 20 2.5 time units. With 3 operators it will become 45 by 3 equal to 15 time units, but with 4 operators, it will not be 45 by 4, 45 by 4 gives us 11.25, which is smaller than the maximum of the individual processing times.

So, the moment the total by the number of operators becomes smaller than the maximum of the individual operations. This time will be the steady state output. So, from now onward for 4 operators it will be 12. Even if we have 5 operators, the output will only be 12 time units per piece. So, every piece will come out after 12 time units. So, the general formula here is, the output in terms of time, time per piece will be equal to sigma p j; sum of processing times divided by n; the number of operators. It is either this or maximum of t j, maximum among the processing times, which happens to be 12, whichever is bigger.

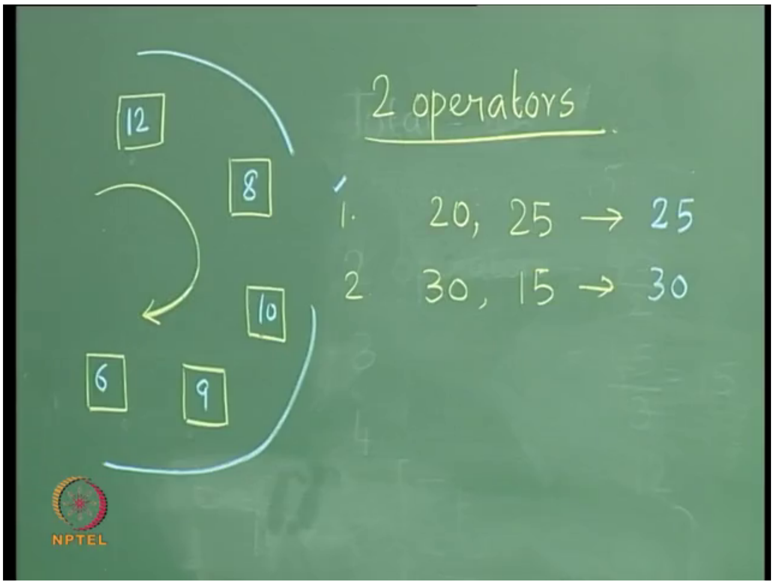
So, this will be, time per piece will be, this is 45. So, if we have 3, then this is 45 by 3 15. This is 12, maximum of this is 15; if this is 4 then this is 45 by 4, which is 11.2 5, maximum of the p j, the processing times, maximum of the processing times as 12.

(Refer Slide Time: 05:08)



So, this is 45 by 4 which is 11 point 2 5, this is 12. So, the total time will be 12. Now the next thing that we should do is, to try and see what happens, if we dedicate machines to operators. Now, if we have 2 operators, and if we look at if we have 2 operators.

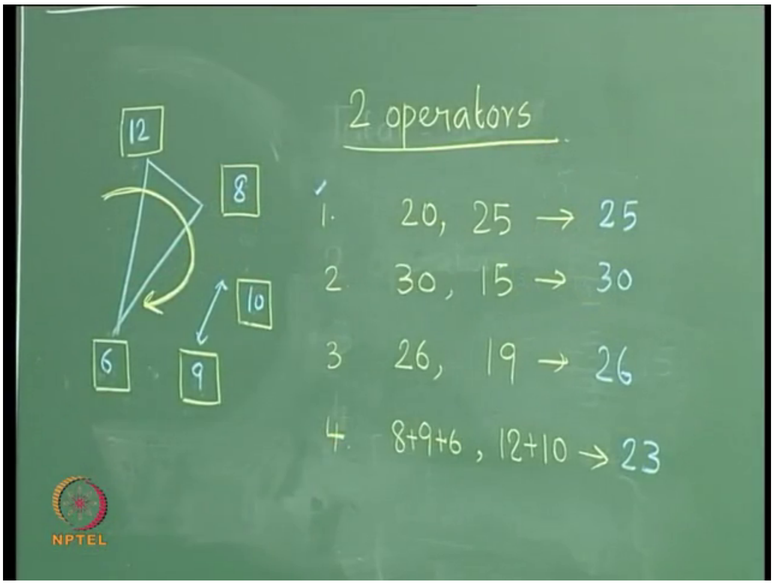
(Refer Slide Time: 05:43)



Now, this operator can take care of these 2 machines. The operator moves back and forth, while the next operator can take care of these 3 machines. And in such case, the time taken by this person will be 20. The time taken by this person will be 25, and the steady state output will be 25 time units per piece. Every piece will come out, or a piece will come out every 25 time units. If we give these 3 to the first person, and these 2 to the other, we would get 30 plus and 15, and the maximum will be 30.

So, we would prefer this, which means we would prefer assigning these 2 to one person and these 3 to another person. We can also look at a situation, where the first operator can do this, while the second operator will do this.

(Refer Slide Time: 07:19)



For example, the first operator could handle this machine, this machine and then this, and then come back while the second person will switch between these 2. He will do these 2 machines.

Now, we have to understand that while the operators moved like this, the flow of the product is still unidirectional, and is given by this direction, which means that at steady state we can assume, that there is a buffer for one piece in front of these. And while the first operator would take a piece, finish these 12, go back to the other one, finish this, and would leave it here, after spending these 20 time units, would leave the piece here, and then what up to this place. By which time this person would have finished an earlier piece and kept it here for this operator to come and take, and then this operator will process it for 6 more time units and leave it.

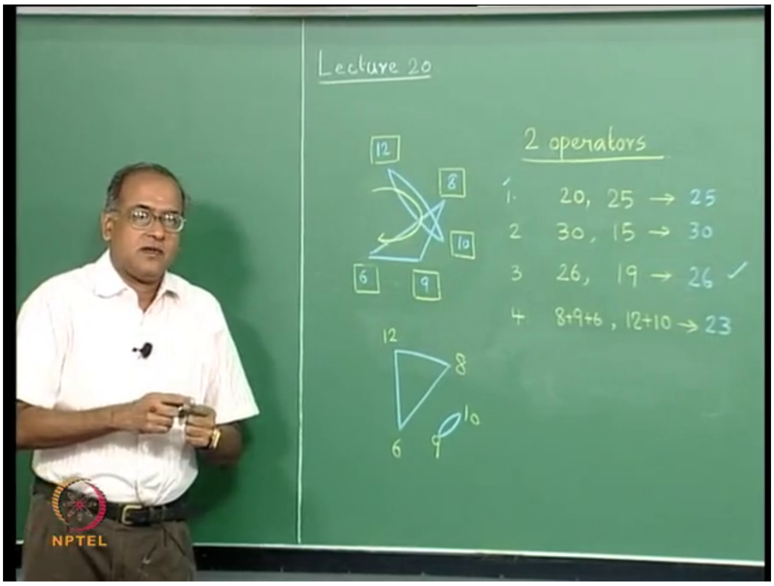
Now, when this operator is coming here and processing this 6 more time units. This person will go back, pick up what has been left by this person, complete this 10 plus 9, leave it here. By the time this person will go take another piece, do these 2 and keep it here and so on. So, the flow of the piece is unidirectional, but the consecutive operations are not, and need not be performed by the same operator.

So, we could have a situation where, the first operator handles this, this and this. While the second operator handles this and this, and a third such solution would give us 12 plus 8 20 plus 6 26 and 19, and a steady state output will be every 26 time units while this is preferred to this. This is still better than this failure. Similarly one could think in terms of 12 plus 9 plus 6 plus, 8 plus 10, which would give 27 plus 18, which is also not really preferred.

Now, we ask ourselves another question, which is the allocation of machines to operators that balances it the best, because the output is actually the maximum of these 2 numbers. So, we would like to minimize the maximum of these 2 numbers. Ideally if they are equal; that is the best scenario, but that does not happen. So, which one is the allocation that would give us the best value here or the minimum value here. Can we get something less than 25? Now if we could do that, if we look at 8 plus 9 17 plus 6 23, 12 plus 10 22.

So, let us look at another solution where, the first operator would do 8 plus 9 plus 6, and the second operator would do 12 plus 10. So, this is 23 this is 22. So, the steady state output is 23. So, we could think in terms of an allocation where, the first operator 8 plus 9 plus 6 for the first operator and for the second operator.

(Refer Slide Time: 11:21)



So, to explain this, now we can call this person as the first operator, this person is the second operator. So, it would be 22 and 23. So, this person will take the first piece, work for 12 time units and keep it here. In between we will assume that the second person has worked on a piece, which has already been kept here. There is a buffer piece, which has worked and kept it here. So, this person will come, take this piece, finish this operation, keep it here, go back take a new piece. Once this is kept this person will come back, take this piece, finish this work, finish this work, leave this and then walk back to this. Meanwhile this person would have taken a new piece, finished this operation and kept it here, and the process will repeat.

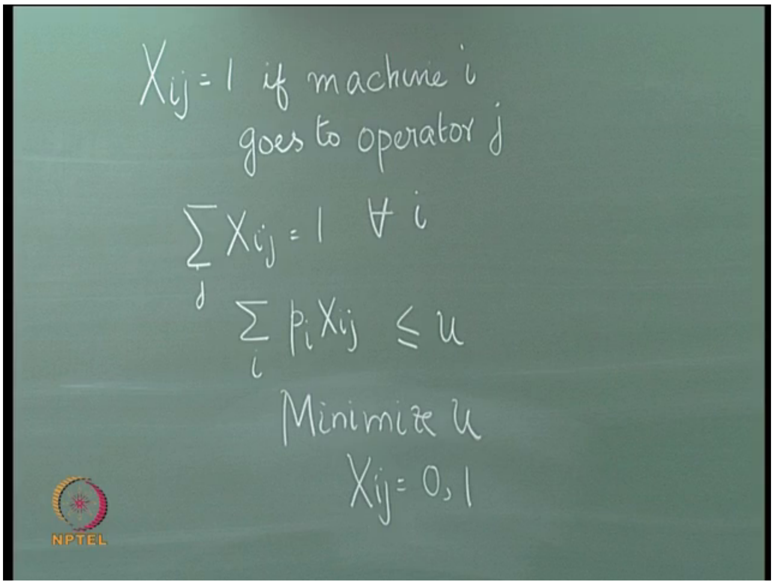
But the flow is again unidirectional. The flow of the piece is unidirectional, but the way the operators work, is not unidirectional. So, now, the best one that we can think of is 23, because the sum is 45, and all these are integer numbers. So, we cannot get 22 and a half, but we could get 23 in this case.

So, the first comparison between this one and the rabbit chasing method is that. The rabbit chasing method is able to give a steady state output of 22 and a half for 2 operators, whereas this method can give only 23 for 2 operators. So, if possible, the rabbit chasing implementation is always a better alternative than dedicating machines to operators. There are other related issues that we will come to after a while, when we again make a comparison between the rabbit chasing method and the dedicating method.

Now, let us go back and see how we actually solve this problem. Right now for a very small example that we are looking at, with 5 operations, or 5 processing times, we are dedicating them to 2 operators and then we should do this. Usually the cell will not contain more than 8 or 9 machines. And therefore, this problem is not a very complicated optimization problem to solve. Nevertheless, this problem is very close to some interesting problems in the operation research literature. And therefore we will look at this.

So, if we formulate this problem. The formulation of this problem will look like this; let x j equal to 1.

(Refer Slide Time: 14:14)



If machine or let X ij equal to 1, if machine I goes to operator j. Now in this case there are 5 machines, and there are 2 operators. So, there are 10 variables; X ij equal to 1, if machine i goes to operator j.

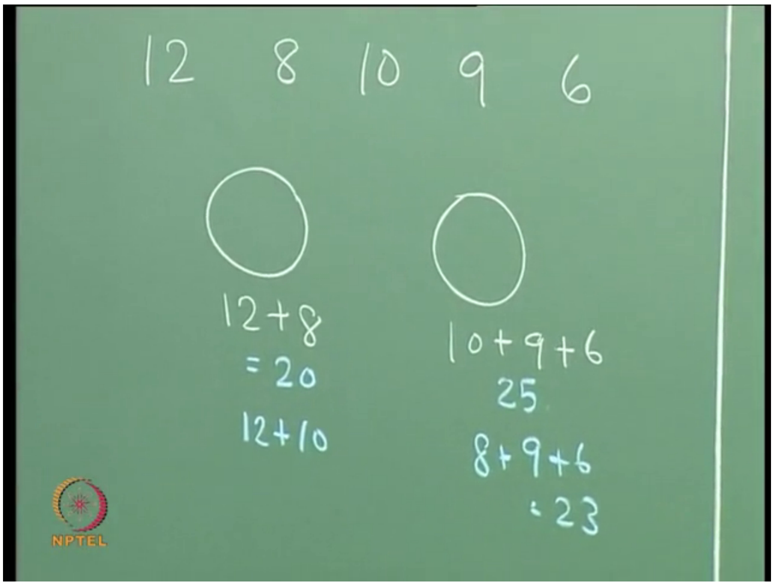
So, each machine should be assigned to only one operator. So, sigma X ij equal to 1 summed over j for every i, each machine will go to only one operator. Now the time that is taken by an operator will be p j p p i X ij summation over i. This is the time taken by the jth operator.

Now, I want to minimize the maximum of this. So, this will be less than or equal to some u, and then we minimize u X ij equal to 0 1. So, this will be the formulation, based on which we can solve this optimization problem, to finally, get the solution that one operator gets this machine 1 and machine 3, the other operators get these.

Now, this problem is actually the same as what is called a parallel machine scheduling problem. It is like assuming that. Now let us for a moment forget this application, where this is an application where there are 5 machines, which now have to be assigned to 2 operators. Each machine has a processing time 12 8 10 9 6 respectively. This is a different setting, this setting is the setting in cellular manufacturing within a cell.

Now, let us look at another problem, where let us say there are 5 jobs. These jobs have processing times of 12 8 10 9 and 6. So, there are 5 jobs with processing times; 12 8 10 9 and 6, and there are 2 machines.

(Refer Slide Time: 17:01)



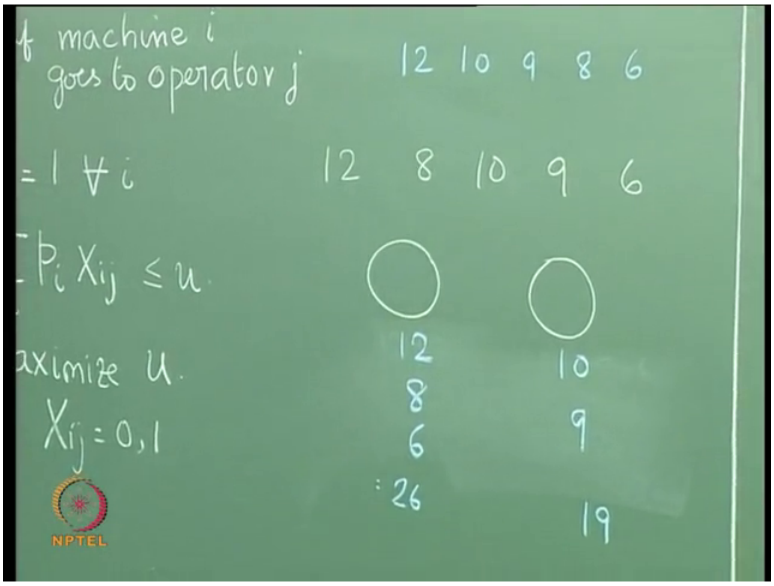
In single machine scheduling problem with parallel processors; so there are 2 identical machines.

Now, which job is assigned to which machine in this problem; such that time at which all the jobs are over is minimized. So, it is like saying if I assign 12 and 8, these 2 jobs to this machine, and I assign these 3 jobs to this machine. This will be over at time equal to 20. This will be over at time equal to 25. So, the time at which both are over, is the maximum of these 2 which is called the makespan. So, it is a make span minimization problem on identical parallel processors, given n jobs.

So, the makespan minimization problem would finally, ask you to do 12 plus 10 22, 8 plus 9 plus 6 equal to 23 will be the optimum solution to this problem. So, the problem of, in a cellular manufacturing context, given certain number of machines in a cell, single product which visits all these machines, and given number of operators 2 in this case. What is the operator assignment that will maximize the output, is the same as another problem where, given these 5 jobs and 2 machines, what is the allocation such that the makespan is minimized.

So, these 2 are related problems. So, there are some heuristics, and there are some optimization procedures for this problem, which can now be applied to that problem, because they are one and the same. Except for the fact that what are machines now become jobs and what are people now become machines. Now one of the standard heuristic to solve this problem is called the s-shaped heuristic, where we arrange the processing times in decreasing order, and start assigning.

(Refer Slide Time: 19:23)

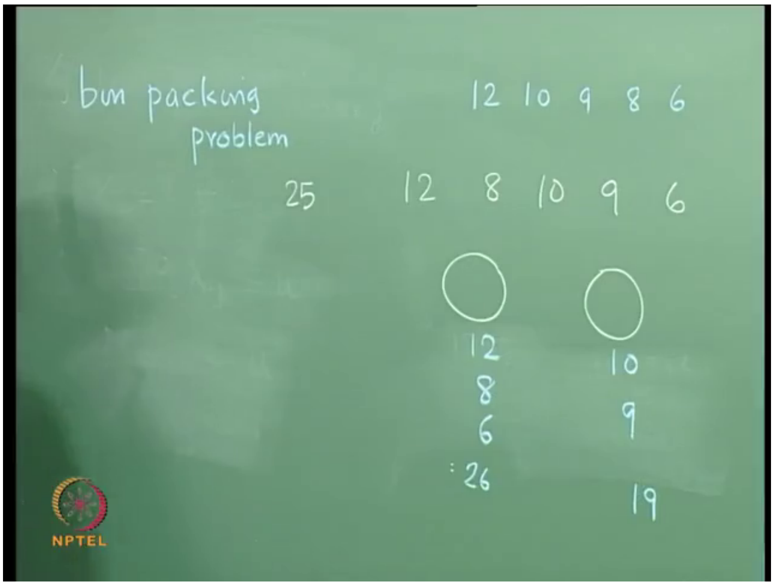


The first one will go here, the second one will go here, the third one will again come here the fourth one will go here the fifth one will come here. So, this s shaped heuristic or alternate s shaped heuristic would finally give us 26 here and 19 here, which is not the optimum solution, but nevertheless it is a heuristic solution.

For a small size that we are looking at with just 5 operations, we can solve it optimally to get the solution, but if in this context of machine scheduling, if there are a large number of jobs, then we have to use heuristics and solve it. Now let us take a closer look at this problem, and then relate it to one other problem, which is famous in the operations research literature. Now this problem is like; there are 5 jobs with non processing times, and there are 2 identical machines, which job is assigned to which machine such that the makespan is minimized. So, this is called makespan minimization in identical parallel processor.

The other equivalent of this problem with a slight difference is called a bin packing problem. Its called a bin packing problem, where we can assume that these lengths.

(Refer Slide Time: 21:06)



These are lengths we can assume that, there are a large number of sticks whose length is say 25, given length say 25. There are a large number of sticks with that length. Now we can take some sticks, and start cutting these lengths. Suppose we take one stick of length 25, and cut a 12 and a 10. The remaining 3 will be a waste. Then we could take another stick and say that we could cut 9 8 and 6, then there is some waste, but within 2 sticks, we will be able to cut all of them.

If the length of the stick is given as 25 if for some reason the length of the stick is given as 20, then we know that we cannot do it with 2 sticks, because the total length is 45, and individual piece maximum length is 20, so we cannot do it in 2 sticks. Same problem is like this, if we give bins of length or height 25, and there are many bins that are available. These items with given lengths have to be put in the bins; such that they are put entirely in the bin.

So, if we take a bin of length 25, we can put a 12 and a 10 in it. We cannot put 12 10 and 6, because that will exceed the length of the bin. So, what is a minimum number of bins that are required, so that I can put all these elements or items into the bin. The size of the bin is known; that is called a bin packing problem. So, this problem can be said as, given bins of size 25. What is the minimum of bin's I require, to put 12 8 10 9 and 6 into the bins; that is called a bin packing problem.

Now, this problem is, given jobs with processing times 12 8 10 9 6, and given 2 machines, what is the makespan, which is equivalent to saying. Now find out the length of the bin; such that in 2 bins, I will be able to accommodate 12 8 10 9 and 6; that is the difference between these 2 problems. Let me repeat it, this problem says, given bins of a certain length. What is the minimum number of bins that I require; such that I can accommodate 12 8 10 9 6 into the bins. This problem is, what should be the length of the bin such that in 2 bins, I will be able to accommodate 12 8 10 9 6.

So, they are related problems. There is a lot of literature on this problem, there is a lot of literature on this problem as well, but the problem that we are trying to look at here, is the same as this, which is very similar to this. So, sometimes the bin packing heuristics are also used to solve the machine scheduling problem. So, let us come back to the cellular manufacturing example, where we have found out that the best one would be 22 and 23.

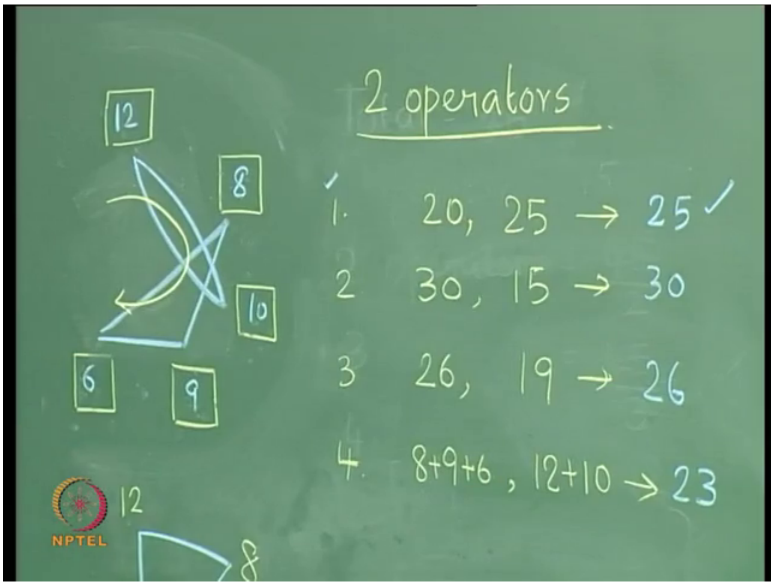
Then we get into the next question of can we implement this 22 and 23. What are their issues in implementing this 22 and 23? Now let me compare this with another instance. For example, let me compare this with, say with this instance. Now this instance we had 12 8 10 9 and 6. So, this instance was 10 and 9, and 12 8 and 6. Now this gives us 26, this gives us 23, 23 is better than 26, therefore, we would like to go for this, but are there implementation issues with this, the answer is yes.

Now, here the operator does this, this, this one and goes back. The second operator does these two. The operator walking paths do not cross each other in this. Even though the operators are not doing exactly considered of operations, this person does 12 8 comes and does a 6 here, this person does 2 consecutive operations, but the operator paths will not cross in this, whereas in this the operator paths are crossing.

So, purely from a safety point of view, we should not be following this, because there is a chance or probability. However, small it is, that the operators may while crossing each other, bump into each other, and if they are carrying some heavy metal or anything, that can fall and create harm to the people who are working. So, purely for safety reasons, we do not encourage solutions like this, where the operator walking paths will cross.

So, one has to understand that when you give a solution to this problem in particular, one has to have the constraint, that the paths do not cross. While 12 8 6 10 9 is acceptable, while 12 9 6 10 8 is acceptable, 12 10 6 9 8 is not acceptable. So, this problem actually at the end of it, we will choose this, which is better than this 23, because there is no operator crossing.

(Refer Slide Time: 27:48)



This is straightforward 20 and 25. So, you get 45 with a maximum output of 25.

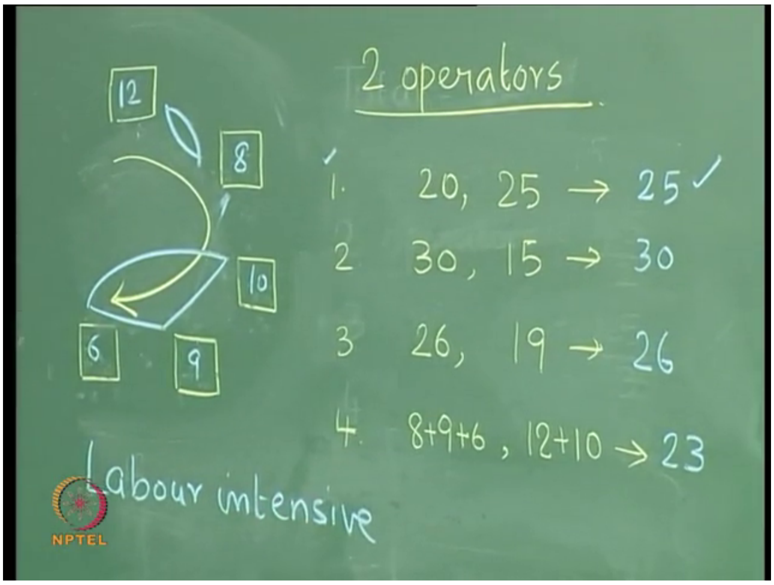
So, this problem is not exactly the machine scheduling problem, but actually this is a further constrained version of the machine scheduling problem. Having said that, I also mentioned that compared to this machine dedication versus rabbit chasing. I did mention that rabbit chasing would always give you a solution; that is better than the one obtained through dedicating. For 2 operators rabbit chasing would give us 22.5, dedicating would give us in this case 25. For 3 rabbit chasing will give you 15, this will not be able to give you 15. This can give you 18, 15, 18 and 12, or 18 17 and 10. This person can do these 2, another one can do these 2, the third one will do this. There is no crossing. One can do 12 18 15. Like that one can provide solutions, but it would still go only up to 18.

For 4 operators there we will get 12. Here we will not get 12, we can get only 15 or 15, because 14 will give you paths that cross. So, 15 is what we will get for 4 operators. 12 this will go to 1, this will go to 1, this will go to 1, these 2 will go to 1 operator. So, 15 will be the time, whereas in rabbit chasing we could get 12. So, wherever possible rabbit chasing is preferred, but rabbit chasing has some concerns. First and foremost, the operator carries the item right through. Secondly, the amount of walking is very high, because every operator walks this entire thing back and forth.

So, if you have the workforce that is a little older, then rabbit chasing would not be preferred. Say if you want rabbit chasing to be executed, then the workforce should be younger. There are some standards and limits set under labor in factories, where one has to ensure that by work definition, the person does not walk beyond a certain distance. So, walking on the one hand is an activity that consumes a lot of energy, and makes it difficult, can affect productivity as you keep walking. There are also regulations which have to be complied with, when one executes a rabbit chasing exercise, one has to do computations to check that, the distance traveled by a person in a shift will be less than or equal to, or less than what is stipulated in the regulation.

So, one has to look at that also. The other aspect is, if I am following a solution where, if I am following the first solution, this is the unidirectional flow of the product.

(Refer Slide Time: 31:23)

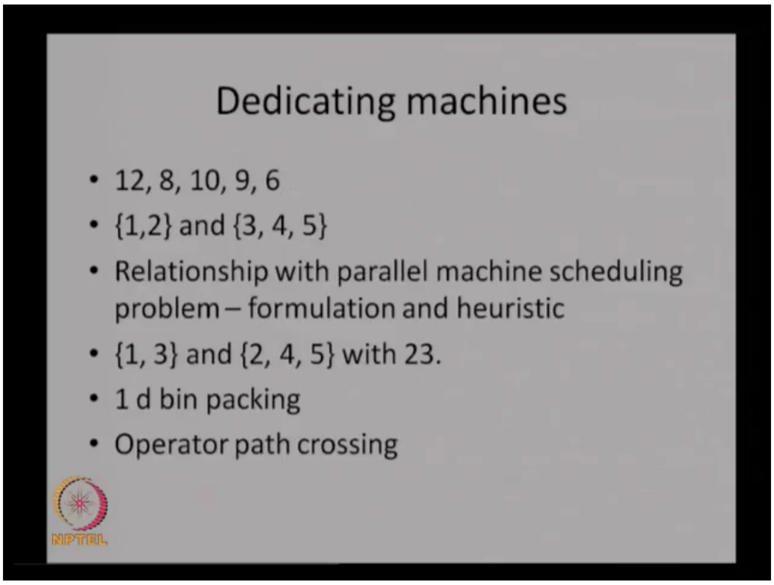


This is the first solution, where its 20 plus 25. Assuming that these are what are called labor intensive cells or labor intensive operations. the operator should have the skill set to work on these 3 machines. And remember always in a manufacturing cell, the machines are functionally dissimilar. They are similar in the context that they require or process similar parts. This machine could be a different machine from this functionally.

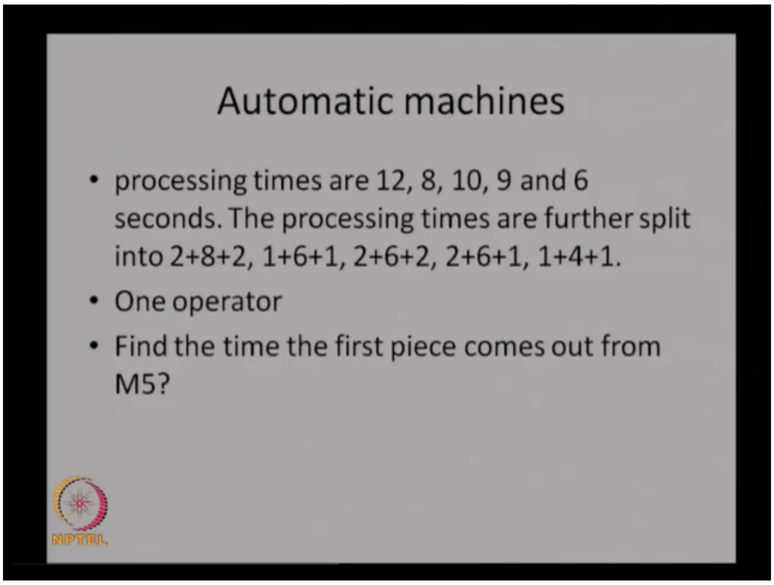
So, this operator who handles these 3 should have the skill set to handle these 3. Another operator should have the skill set to handle these 2. Whereas, if I do rabbit chasing, every one of the operators should have the skill set to handle every one of the machines. So, the skill set requirement is also higher in terms of, when we use rabbit chasing. So, rabbit chasing versus dedicating; one has to be a little careful and one has to understand all these, when we choose which of the methods to follow, and most importantly be able to make sure that when we dedicate, the operator paths, walking paths do not cross.

The output from this system is always the maximum of the operator time versus cycle time, or individual processing time. Now we move to the next one.

(Refer Slide Time: 33:20)



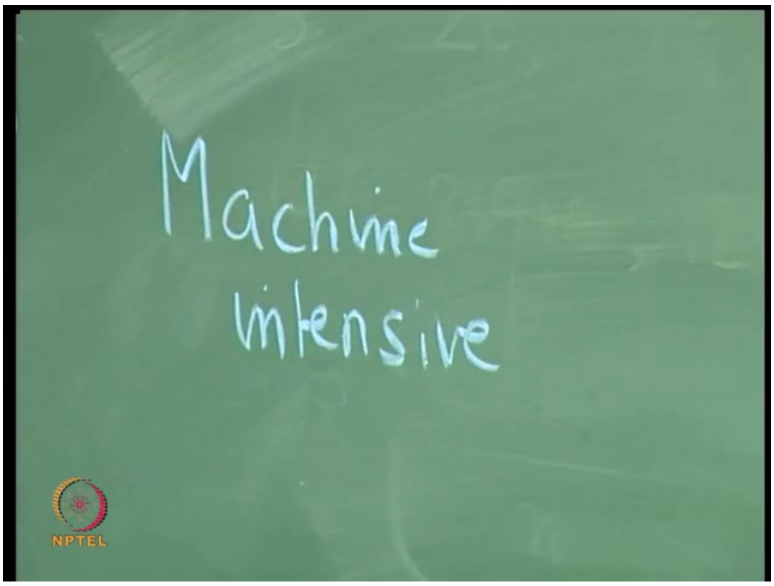
(Refer Slide Time: 33:23)



Now, let us look at the next one where, these machines are all automatic machines, which means that the operator will not or need not be holding a piece or feeding something, when the operation is actually happening. So, the operator has only the task of loading and unloading, while the operation manufacturing is actually happening. The operator need not be next to the machine.

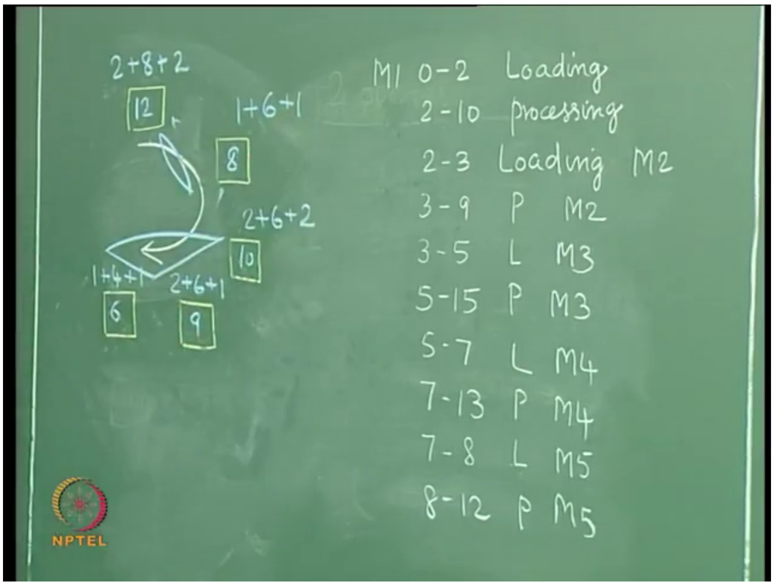
So, we would assume that machines are of this type, again there are 5. Now such cells are called machine intensive cells.

(Refer Slide Time: 34:04)



Labor intensive cells and machine intensive cells, in the sense that the operators job would be only to load and unload the piece on the machine, and need not be present. So, in such a case again, the operator can handle multiple, can handle multiple machines.

(Refer Slide Time: 34:33)



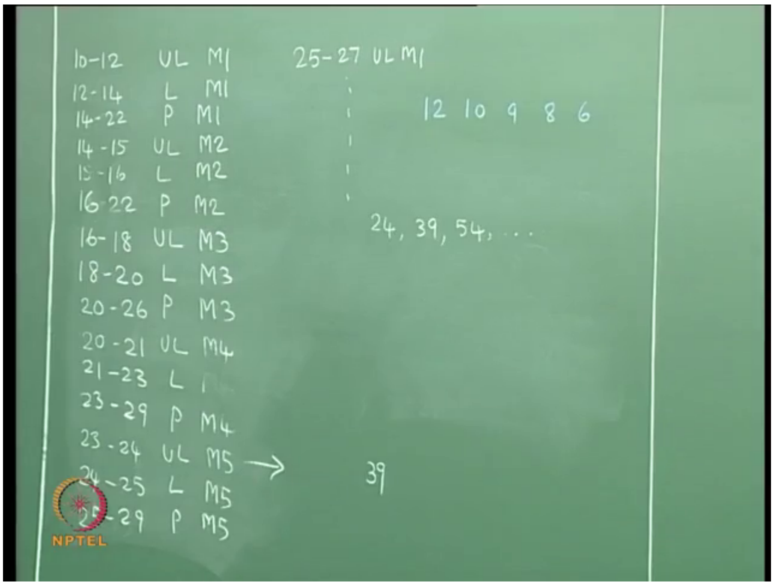
Now, let us look at how this kind of a system works out. So, let us also assume that this processing time or time given, which is 12, is actually 2 plus 8 plus 2, which means 2 out of the 12, if they are in seconds; 2 seconds to load 8 seconds to process, 2 seconds to unload. Similarly this we would take it as 1 plus 6 plus 1. This would take it as 2 plus 6 plus 2. This we would take it as 2 plus 6 plus 1, and we take this as 1 plus 4 plus 1. Now let us do some calculations, as to what happens when, a single operator is actually doing all the 5 then we can extend this when 2 operators are doing.

So, we will also assume that at steady state; one piece is always available here for the operator to work; so at time equal to 0. So, operator on first machine M 1 0 to 2 loading, 2 to 10, this is M 1, 2 to 10 processing, 2 to 10 processing, but the operator is free at 2. So, we assume that there is a piece here. So, 2 to 3 loading M 2.

So, at 3 this begins. So, 3 to 9 processing M 2. P stands for processing, M stands for the machine, L stands for loading, U stands for unloading. So, 3 to 9 is processing on M 2. The operator is free at 3. So, there is a piece here. So, 3 to 5 loading M 3; that is here, 5 to 15 processing M 3. Again the operator comes here at 5. So, 5 to 7 loading M 4, 7 to 13 processing M 4, person is free again at 7.

So, 7 to 8 loading M 5. Then the person will go back. So, at 8, the person has loaded on M 5, because that is 2 plus 1 3 plus 2 5 plus 2 7 plus 1 8. This loading has been completed. So, the person will go back to this. Now what? Now we have to check this, at 8 the person goes back, but the processing is over only at 10 here. So, 10 to 12 unloading on M 1.

(Refer Slide Time: 38:23)



So, the person unloads on M 1. Now he has to go back and load another piece on M 1. So, 12 to 14 loads M 1 again. Now 14 to 22 processing on M 1, comes here at time equal to 14. So, at 14 processing on M 2 is over. So, 14 to 15 unloading on M 2, 15 to 16 loading on M 2, 16 to 22 processing on M 2.

So, at 16 the person can come to M 3 and unloads. So, from 16 to 18, it is unload on M 3, unload on M 3. Loads another piece on M 3, which is 18 to 20, load on M 3, and the processing happens between 20 and 26 processing on M 3.

So, at 20 the operator comes to M 4. So, unloads 20 21 unload on M 4, 21 to 23 loads on M 4, 21 to 23 loads on M 4, and 23 to 29 is processing on M 4. So, at 23 the person comes to M 5. So, 23 to 24 will be unload on M 5, 24 to 25 will be loading on M 5, and processing will happen between 25 to 29 processing on M 5.

In fact, when the person comes here to M 5 at time 23, we have to check whether the piece is over on M 5 at time 23. So, we go back and check that and we realize that 7 to 8 the person are actually loaded on M 5. So, 8 to 12 will be processing on M 5. So, by 12 the processing would be over, person comes again at 23. So, the person can unload this and. So, the first piece will come out at time equal to 24 on employee. So, now the person goes back again at time equal to 25 to M 1. So, now, processing on M 1 gets over at time equal to 22. So, at 25 the person can unload on M 1 25 to 27 unload on M 1, and the cycle continues.

So, the first piece comes out at time equal to 24 when we unload on M 5. And from here it is 10 to 24. So, 14 time units. So, 25 plus 14 at 39 the next piece will come out. So, the piece, first piece will come out at time equal to 24. The next piece will come out at 39, 24 plus 15, 39, 39 plus 15 54 and so on. So, at steady state one would say that a piece can come out every 15 time units. And the steady state implies that initially when we begin, we have one piece that is actually available, there in front of each machine, and then the process actually continues.

So, when we follow this kind of an approach, where a single operator comes and does all the 5, at steady state we are able to get an output every 14, 10 to 24, so 14, 25 to 39. So, actually every 15 minutes we will get every piece that comes out, because this one will come out at 24. There is another one unit that is hidden here. So, the person starts again at 25; so 10 plus 14 24, 25 plus 14 39 and so on. So, every 15 minutes we could expect a piece to come out.

Now, that happens, because there are already pieces available here. So, the steady state output, even for one operator, is not the sum of 8 plus 6 plus 6 plus 6 plus 4, because certain things can happen in parallel, and the operator is not there to. It need not be present when the actual machining is happening, but for every piece what the operator must do, is the loading and unloading. So, we can do that. So, this is 4 plus 2 6 plus 4 10 plus 3 13 plus 2 15, which is actually the steady state output that we have here. While the operator is moving, the work actually gets done.

So, the 15 that the operator spends in load unload in one cycle, is actually more than any of those individual manufacturing times. The individual processing times are 8 6 6 6 and 4. So, the maximum of these is still less than the time the operator actually spends, which is the load unload time. Therefore, at steady state, the operator becomes the bottleneck, and the 15 becomes the steady state output of this system, and because we have enough inventories the whole thing is taken care of. Only the maximum comes into the picture, because there is an inventory of one piece in front of every machine.

So, the maximum of these processing times is 8, which happens to be less than that, and therefore, it will come at every 15 time units. Now we can do a similar computation. If we dedicate, and if we say that we use this particular solution, where one operator does these 2, the other operator does these 3. Now the learning from this exercise is, when one operator does these 2, per cycle, the load unload time that the operator spends will be 2 plus 2 4 plus 1 5 plus 1 6. Now 6 is smaller than the processing times between 8 and 6, the maximum is 8.

So, this at steady state will give a piece every 8 time units. Here the load unload times will be 4 plus 3 7 plus 2 9, 9 is bigger than the maximum of these 9. So, this will give out every 9 time units, and the steady state output will be the bigger of this 8 and 9, which will be 9. So, when one operator desks we can get an output in every 15 time units at steady state, while 2 operators do it, and if we dedicate in this order, we will get it in 9 time units if we dedicate the operators this way 12 9 and 6 8 and 10, if we dedicate them.

Then the load unload times will be 4 plus 3 7 plus 2 9. Again 9 is greater than 8. So, this output will be 9. Here it will be 6. So, this will be. This is 6 1 plus 1 2 plus 4 plus 2 6, which is equal to the maximum of the processing times 6. So, this operator will do 6 this operator will do 9.

But the steady state output will be 9. So, like this we can do computations, and then show; what is the correct configuration, and which one becomes the bottleneck, whether the operator becomes the bottleneck or the machine becomes the bottleneck. Now when we have a single operator calculation here, the operator is the bottleneck, because some of the load unload times is 15, which is bigger than the individual, the maximum of the individual processing times so operator is the bottleneck, but if we dedicate here one operator here, one operator here.

Here the machine will become the bottleneck, because the operator time is only 6, while the maximum machine time is 8. So, machine becomes the bottleneck. So, in all these calculations, we have assumed that, the operator walking time is negligible. Since the operator walking time is negligible, we have not taken into account the time taken for these walking. When that time also is taken, we just have to add that time into the operator time. So, this table or this calculation can simply be circumvented by, finding out the time actually spent by the operator in loading unloading and walking.

So, if we say that there is one time unit to walk, then there will be 3 time units added to the operator. So, this way we can do these dedicating operators to machines, in the context of what are called machine intensive cells. So, we have seen one model for labour intensive cell, and we have seen one model for machine intensive cell. Machine intensive cells assume that all these are automated machines, and these machines the operator does only the loading and unloading, and is not a, need not be present while the actual machining takes place. With some more sophistication, and more advanced machines, sometimes at the end of the machine, machining process. The machine simply ejects the piece out of the machine.

So, in such a case the operator does not even unload it fully. The piece is already ejected. The only thing that the operator has to do, is to take the ejected piece and put it here, which means this unloading time can also be reduced. Now when the unloading time gets reduced, more and more the machine will start becoming the bottleneck. So, depending on whether it is a labor intensive cell, or whether it is a machine intensive cell, we can use these models of rabbit chasing, dedicating operators to machines; such that operator walking paths do not cross. And then separating the time into loading, machining and unloading time, and assuming automatic machines. Do these calculations to find out what is the best operator assignment to get the best output from this system.

Other models related to operator assignment, we will see in the next lecture.