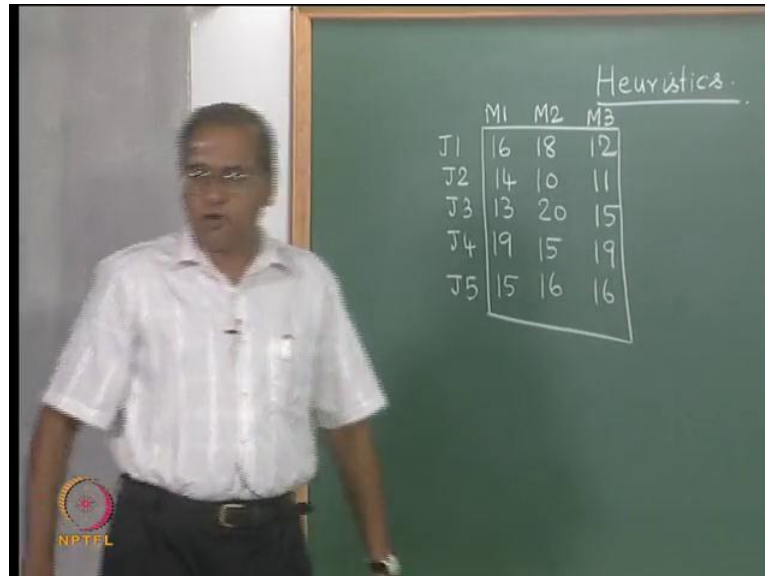


Operations and Supply Chain Management
Prof. G. Srinivasan
Department of Management Studies
Indian Institute of Technology, Madras

Lecture - 27

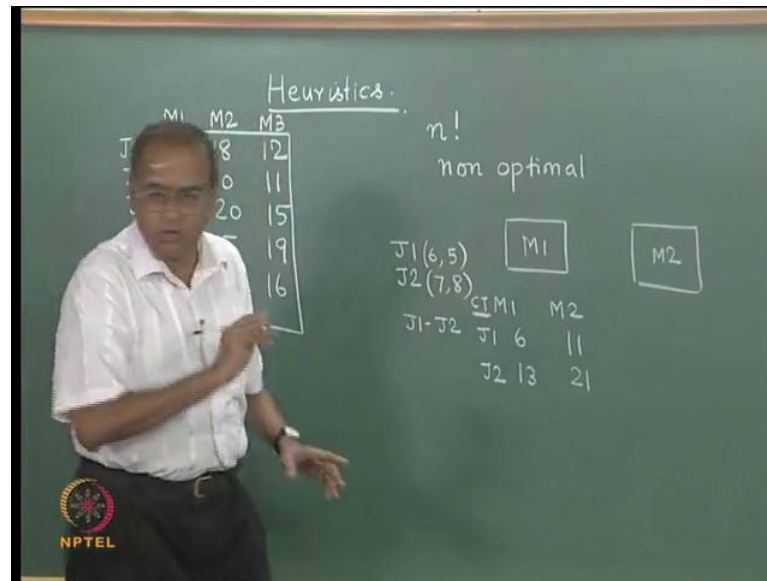
Flow Shop Scheduling - Heuristics - Palmer, Campbell Dudek Smith Algorithm

(Refer Slide Time: 00:14)



In the last two lectures we have been addressing the n job, m machine, flow shop scheduling problem to minimize make span. We looked at the Johnson's algorithm, which would solve the two machine problem optimally, we also looked at the extension of the Johnson's algorithm, where the 3 machine problem the make span can be solved optimally, provided the data on the processing times satisfy certain conditions. Now, we also looked at a branch and bound algorithm, which we could use to solve the general m machine problem optimally in particular we showed an example with three machines.

(Refer Slide Time: 01:01)



We also mentioned that, the branch and bound in the worst case if there are n jobs, then the branch and bound could evaluate all the n factorial possible permutations or n factorial possible sequences, before it comes or before it arrives at the optimum solution. So, in the worst case the branch and bound algorithm can do n factorial, which means that it will or can do an exponential number of computations. So, when it does an exponential number of computations, the computing time taken to solve the problem optimally becomes very large, as the size of the problem increases.

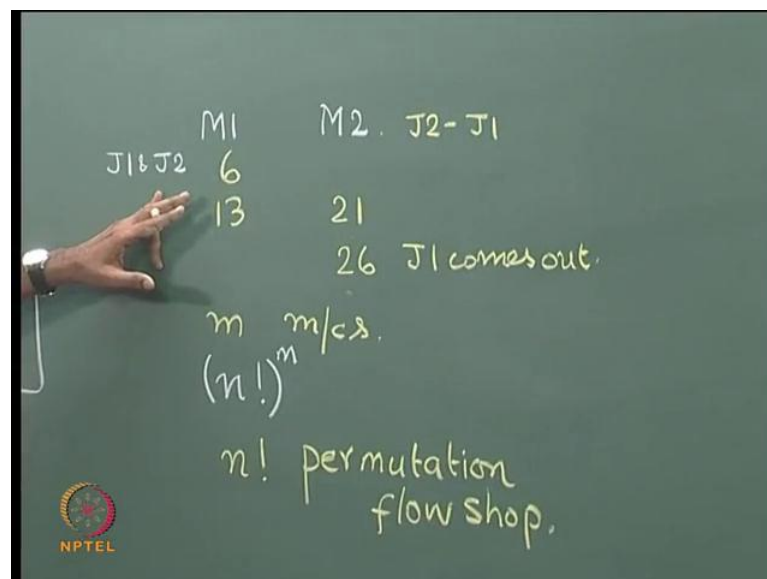
So, in such cases it is customary to resort to what are called heuristics or heuristic solutions, which essentially do not guarantee optimum. So, in some sense they do not guaranty optimum, so they can be termed as non optimal methods, but for certain instances the heuristic solutions can give the optimum solutions themselves. The heuristics are capable of giving the optimum in certain instances, but they do not guarantee optimal solutions on all occasions.

So, we will now look at some heuristics to solve the general flow shop scheduling problem to minimize make span; but before that let us also look at one aspect which is regarding this n factorial. Now, if we consider a two machine flow shop, which means we have M_1 here, and we have M_2 here. And let us say we have only two jobs J_1 and J_2 which are here, so since there are two jobs, there are two sequences possible, which is J_1 followed by J_2 and J_2 followed by J_1 .

Now, let us assume that the processing times for J 1 are 6 and 5 which means 6 on M 1 and 5 on M 2, and processing time for J 2 are 7 and 8, 7 on M 1, 8 on M 2. We have already seen the computation of the completion times, and the make span, but let us revisit that aspect once more. So, if we consider the sequence J 1, J 2 which means job J 1 goes first, and then followed by J 2 we automatically compute the sequence like this.

We look at completion times on M 1 and M 2, let me say completion times here for J 1 for J 2. So, we would say that J 1 first goes to M 1 at time equal to 0 finishes at 6, then J 1 goes to M 2 finishes at 11, J 2 enters M 1 at time equal to 6 finishes at 13, and then this is available at 11. So, J 2 can take M 2 at 13, 13 plus 8 21 this we are very familiar with. We have made an important assumption here that, the moment we decide that the sequences J 1 followed by J 2 that sequences maintained on all the machines, both M 1 and M 2 we have maintained that sequence in our computation. For example, let us look at a scenario where we decide that the sequence on M 1 is J 1 followed by J 2, and the sequence on M 2 is J 2 followed by J 1. So, let me show those calculations.

(Refer Slide Time: 05:08)



So, we will first look at M 1 and then we look at M 2 let us assume on M 1 the sequence is J 1 followed by J 2. So, the sequence is J 1 followed by J 2 on M 1, which means that in M 1 we first start with J 1 takes 6 time units, so this comes out at time equal to 6, now J 2 enters M 1 at time equal to 6 therefore, will come out at time equal to 13. Now, let us here on M 2 the sequence is J 2 followed by J 1, so let me write that on M 2 it is J 2

followed by J 1.

So, this would mean that J 1 will not immediately go to M 2 even though M 2 is free because the sequence on M 2 is J 2 followed by J 1. So, M 2 will wait for J 2 to be over, so J 2 can access M 2 at time equal to 13, and then J 2 requires another 8 units at 21 J 2 will come out. Since the sequence on M 2 is J 2 followed by J 1, now J 1 can enter M 2 after J 2 is completed, so J 1 even though it has completed here at 6 will wait till 21 and then it would take another 5 units and then at 26 J 1 comes out.

The next span therefore, for this order is 26 compared to the make span here, which is 21, now if we see very carefully the flow shop assumption is not violated the flow shop assumption means that every job will visit the machines in the same order. So, even if we look at this schedule or time table J 1 first visits M 1, and then J 1 visits M 2, J 2 visits M 1, J 2 visits M 2. So, the flow shop assumption is not violated, so in principle, in the flow shop, if the flow shop has m machines, in each machine the n jobs can be sequenced in n factorial ways.

So, over each machine it can be done in n factorial ways and since there are m machines it can be done the total sequence is possible actually n factorial to the power m . But, it is not common and customary to use this; it is fairly obvious making a comparison between this and this. If we retain the same sequence for all the machines, on many occasions we would get better value of the performance measure, such as make span or completion time and so on.

For example, the make span is 21, the make span is 26 sum of completion times is 33, sum of completion times is 47. So, if there were a due date related measure, once again this would perhaps show a higher tardiness than this; so for what are called regular performance measures, where the performance measure or the objective function will increase when completion time is increased. It is better not to follow this, and to follow this where the sequence of jobs is the same on all the machines.

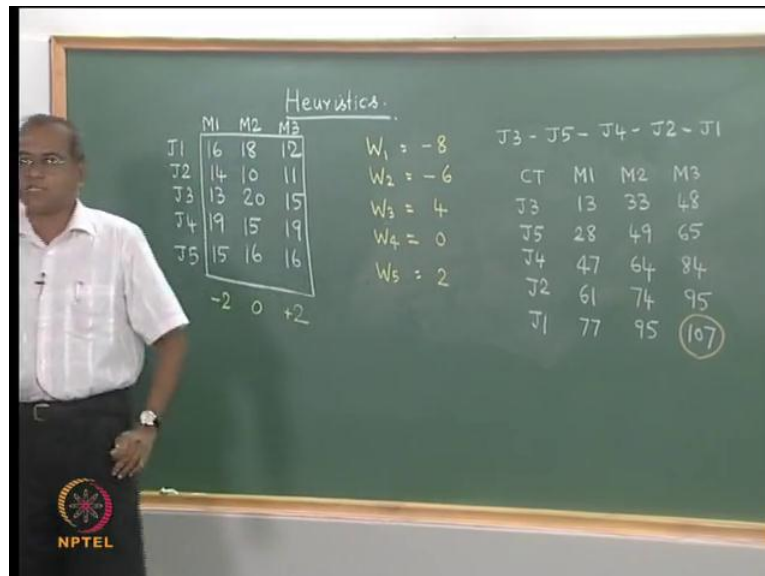
This is a situation where the order or sequence of job is different on different machines, even in a flow shop. So, it is advantageous to have a situation or to model it in such a manner that the sequence of visit is the same on all the machines therefore, we do not consider all the n factorial to the power m ; instead we consider only n factorial sequences. So, 1 out of the n factorial will be optimal for the first machine, and the same

1 out of the n factorial now is chosen, such that it is optimal when we consider all the machines.

So, we reduced the feasible solution space from n factorial to the power m to n factorial, so those sequences where or situation where, we consider the same sequence for all the machines, such as this is called a permutation flow shop. So, unless otherwise stated, flow shop would mean a permutation flow shop and we will not look at non permutation flow shops, where the order or visit on the machines are different. So, far we have seen only permutation flow shops in this lecture series and we will continue to look only at permutation flow shops in this lecture series.

Now, let us go back to the heuristics to solve the problem. So now, let us consider this 5 job, 3 machine flow shop sequencing problem to minimize make span, and as I have just mentioned, we will consider only permutation sequences and we will not look at non permutation sequences. So, 1 out of the n factorial will be optimal in this case, now we observe first of all we have to check whether it satisfies the Johnson condition, if it satisfies the Johnson condition then we could get the optimal solution straightaway using a simple algorithm.

(Refer Slide Time: 11:51)



Now, minimum on the first machine is 13, minimum on the third machine is 11, maximum on the second machine is 20. So, maximum on the second machine is greater than the minimum on the first, as well as minimum on the third. Therefore, the Johnson

condition is not satisfied. So, we cannot apply the Johnson condition to try and get the optimal solution to this problem, we may not get the optimum solution if we continue using the Johnson condition and solve it. So, let us look at some other heuristic solutions to this problem.

Now, the first heuristic that we will look at will try and find out a weighted sum for each of these jobs. So, we will try and give some weights to each of these machines, and then we try and find a weighted sum for every job, so let us begin with giving a weight of 0 to the machine in the middle, giving a weight of plus 2 here and giving weight of minus 2 for this machine. And then let us try and find out a weighted sum of processing times for each of the jobs.

So, let us call that as say let us call W_1 as the weight for job 1, so for job 1 the weight will be $16 \times (-2)$, which is -32 plus 18×0 plus $12 \times 2 = 24$. So, $24 - 32$ is -8 , now W_2 will be $14 \times (-2)$ which is -28 minus 28 plus 22 is -6 . W_3 will be $13 \times (-2)$ minus 26 plus $15 \times 2 = 30$ which will be 4 . W_4 will be $19 \times (-2) - 38$ plus 19×2 plus 38 , 0 and W_5 will be $15 \times (-2)$ is -30 plus $16 \times 2 = 32$, so it will be 2 .

So, we have now found a weight associated with each job, based on the weights that we have arbitrarily assigned to the machines. So, it is very clear that if these weights are changed these numbers will also change, but right now let us assume a standard weight of 0 for the middle plus 2 here and minus 2 here. And then we have computed the weights, now sort the jobs in decreasing order of the weight, non increasing order or descending order of the weights.

So, the maximum is W_3 , so J 3 comes first the next 1 is W_5 , so J 5 comes second the third 1 is 0. So, J 4 comes third the next one is -6 , so J 2 comes here and the last is -8 J 1 comes here, so this is the sequence that we get, if we compute a weight for each job, based on a weight given to each machine. Now, for this sequence let us try and find out the make span, so we write the completion times for this sequence, we have M 1, M 2, M 3, J 3, J 5, J 4, J 2, J 1.

So, J 3 starts at 0 finishes at 13 now goes to M 2 at 13 takes another 20, so finishes at 33 now goes to M 3 at 33 takes another 15 and finishes at 48. Now, J 5 can begin at 10 equal to 13 on M 1 takes another 15, so finishes at 28 has to wait till 33, so that M 2 is free

takes another 16. So, finishes at 49, M 3 is already available at 48, so starts on M 3 at 49 takes another 16, so finishes at 65, J 4 can start on 20 on M 1 at 28. So, 28 plus 19 it finishes at 47.

M 2 is available only at 49, so it waits till 49 visits M 2 takes another 15 units finishes at 64, M 3 is available only at 65. So, goes to M 3 at 65 takes another 19 and finishes at 84. J 2 starts at 47 on M 1 takes another 14, so finishes at 61 waits till 64 for M 2 starts at 64 on M 2 takes another 10 for 74 waits still 84 for M 3 starts at 84 takes another 11, so finishes at 95.

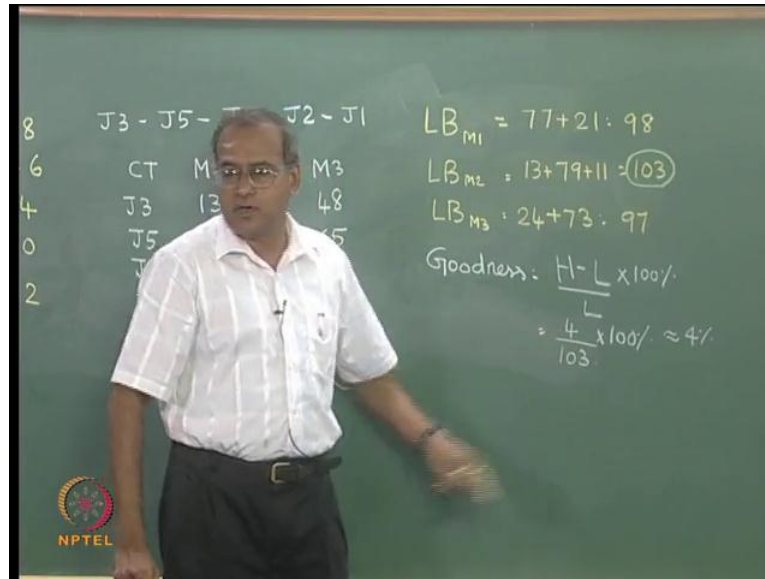
Now, J 1 starts at 61 on M 1 takes another 16, so finishes at 77, M 2 is already available at 74, so starts at 77, 77 plus 18 is 95 it comes out at 95 M 3 is available. So, starts at 95 takes another 12 and finishes at 107, so make span associated with this sequence is 107, this is the make span associated with this sequence.

So, this is how this algorithm works, so very simple algorithm, so one pass algorithm where you define weights for the machines. Right now the weights are arbitrary, but there is a logic in this which we will see. Right now the weights are arbitrary, but for any given weight you can calculate an index or a weight for every job, compute that index and then sort the jobs in decreasing or non increasing value of the indices to get the sequence like this and evaluate the make span of this sequence and that gives a heuristic solution.

So, this heuristic algorithm will evaluate only one sequence it could evaluate 2 if for example, the some of these indices are equal. So, you could have 2 or more if some of these indices are equal, but in general it evaluates only one sequence if all the values of these each of these indices is unique. So, it gives a solution with make span equal to 107, now 1 thing that we know is that this 107 need not be optimal, with the bit of luck it could be optimum. But, then there is no guaranty that the 107 that we obtain is indeed optimal.

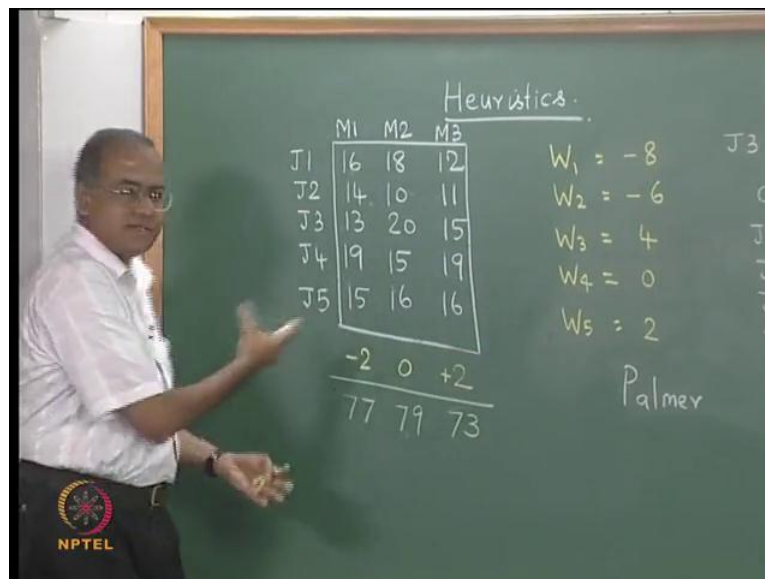
So, we need to find out how good it is, is it good enough or is it not good enough, now in order to do that we also try and find out lower bounds for the make span, we have already seen lower bounds for the make span based on each of the machines. So, we apply those lower bounds to see, what we get as a lower bound for the make span.

(Refer Slide Time: 20:51)



So, lower bound based on M 1 is equal to.

(Refer Slide Time: 20:57)



Now, let us first find out sum of processing times on all the 3 machines 16, 30, 43, 62 plus 15, 77. 18, 28, 48, 63 plus 16, 79. 12, 23, 38, 57 plus 16 is 73. So, the lower bound based on each of these is M 1 all require 77 time units, but then the last job on M 1 has to go to M 2 and M 3. So, the minimum additional time that we require is the minimum of these sums 30, 21, 35, 34, 32, 21 is the minimum, so minimum additional time required is 77 plus 21 ((Refer Time: 22:06)).

So, lower bound based on M 1 is 77 plus 21 which is 98, now lower bound based on M 2 will be the earliest we can start processing on M 2 is the minimum of these which is 13. Because, no job can directly start on M 2, it has to go to M 1 and then only it can start M 2, so the earliest M 2 can begin is the minimum of these numbers which is 13, M 2 itself requires another 79. And then the last job on M 2 also has to go to M 3, so it requires at least an addition which is the minimum of these which happens to be 11.

So, the lower bound will be 13 plus 79 plus 11, so this will be 13 plus 79 plus 11 which is 93, 93 plus 13 is 106 is a lower bound based on M 2. Now, lower bound based on M 3 is no job can start on M 3 directly, so any job that the first job on M 3 should have visited M 1 and M 2. So, the earliest M 3 can begin is these 2 sum to be minimum, so this is 34, 24, 33, 34, 31, 24 is the minimum plus it requires a 73, so lower bound on M 3 is equal to 24 plus 73 which is 97.

Now, out of the 3 lower bounds the maximum one is the best bound, so 106 is a very good lower bound that we have got from using the 3 machines. So, now we have 2 pieces of information, we know definitely based on the lower bound that the make span cannot be less than 106. So, the make span can only be 106 or more, now we also know from here that since we have the optimum make span can be 106 or more based on the lower bound.

Based on this we already have a feasible sequence, with the make span of 107 - therefore, the optimum make span can only be 107 or less. So, the goodness will be $H - L$ by L into 100 percent, now we assume the worst case the optimum is the lower bound, which is the worst case. And therefore, if the optimum were 103 then the goodness of this is $107 - 103$ divided by 103 into 100 percent, so this is 4 by 103 into 100 percent. So, this is let me say approximately 4 percent or less than 4 percent, 3 point something percent we would say 4 percent.

So, if we are willing to accept a solution which is say worst case is 4 percent above the optimum, the reason the worst case 4 percent comes because you do not know the if this optimum is more than L . Actually this should be $H - O$ by O , heuristic minus optimum by optimum, since we do not know the optimum we substituted with a lower bound. So, the optimum is 103 or more therefore, the actual goodness will be 4 or less because this can increase and $H - L$ can actually decrease.

So, in the worst case it can only be 3 plus something, so if we are willing to accept a solution which is within 4 percent, then we can comfortably take this and we need not worry about trying to get to the optimum. But, on the other hand we would like solutions which are better than this, do we have sequences which have make span less than 106. In such a case then we can look at other heuristic algorithms, and try to see if we can get solutions, which are less than 106 make span.

Now, this heuristic is called the Palmers heuristic or Palmers algorithm, and was developed by Palmer in the year 1965. Even though these values of minus 2, 0 and plus 2 appear arbitrary, there is a rationale in choosing these values, if we see very carefully these sequences J 3, J 5, J 4, J 2 and J 1. They were based on a decreasing order of the index, now J 3 had the highest value of the index, while J 1 had the lowest value of the index.

Now, if you see these J 1 is it is very clear it got the lowest value because this 16 is bigger than 12, and J 3 had a higher value positive value because this 13 is smaller than 15. Now, for a 3 machine problem we have put a 0 here, so whatever is there on second machine does not matter, only what is there on the first and third machines matter. So, if a job has a higher processing time on the third machine or on the last machine, and a lower processing time on the first machine like J 3 it will have a higher value of the index.

Another job which has a higher value on M 1 and a lower value on M 3, will have a lower value of the index and therefore, figure later in the sequence. J 3 which has a lower value on M 1 and higher value on M 3, will have a greater than 0 and a higher value of the index and therefore, we will figure early in the sequence. Now, this is very, very close to the Johnson idea that if we go back and see the rule, the machine that has the smallest number on M 1 will appear from the left, which means it will appear early in the sequence while jobs that have smaller values on M 2 will appear from the right, which means it will appear later in the sequence.

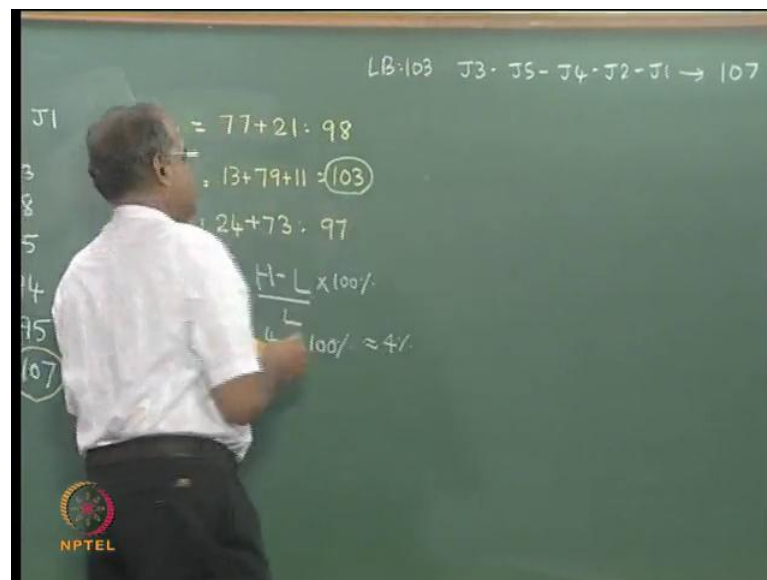
That essence is captured by the weights, in principle that essence is captured by the weight. So the principle should be the middle one has a certain weight, the 1's to the right generalizing it to a 5 machine or a 6 machine problem, the 1 that comes to the right should have positive and increasing weights. And those to the left should have negative

and decreasing as - then the absolute weight will be high, but because of the negative it will be decreasing.

Then we fit this algorithm into the Johnson principle which works, but then there are a couple of limitations here. Because, we put a 0, then the second machine is not contributing at all. And then there are a few other questions like should this be 0 or should this be some constant, some other thing other than 0. Now, should these be the same all these questions can come, but essentially the principle behind computing this index is based on the Johnson idea.

Now, what happens when I have an even number, if I have 4 machines what do I do? So the suggestion was minus 3, minus 1, plus 1 and plus 3 there is no middle machine. So, the two middle 1's will get minus 1 and plus 1 and then a plus 3 on this side and a minus 3 on the other side. So, there were several improvements to this algorithm. But the most important contribution or two things associated with this algorithm is that it is a very early algorithm - one of the earliest. And that the very idea of the weight on the index captures the essence behind the Johnson's algorithm.

(Refer Slide Time: 31:13)

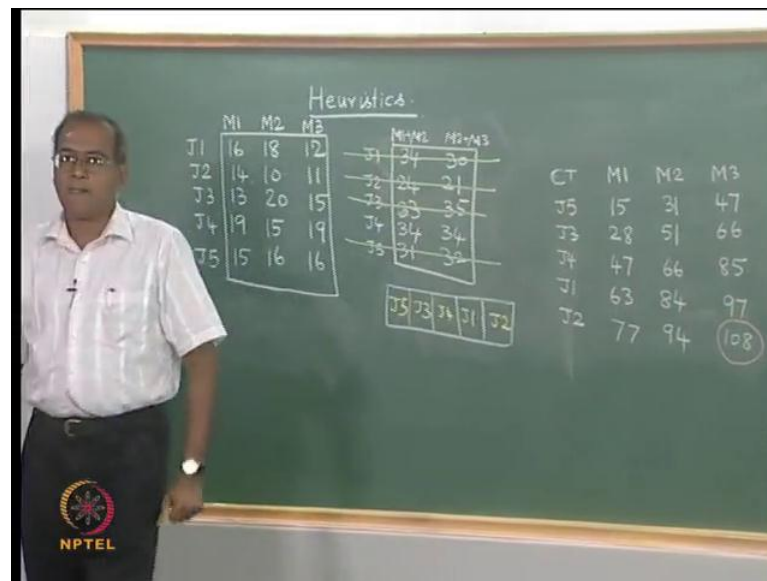


Now, let us go to another algorithm let us simply retain this solution somewhere and let us write this solution here which is J 3, J 5, J 4, J 2, J 1 with make span equal to 107. Now, let us go back to another algorithm and see what we do. Let us also write that our lower bound is 103 that we have, now when we started, we check the Johnson condition

and we said that this does not satisfy the Johnson condition, so that it definitely gives the optimum solution, still we could have used the same algorithm.

Because, if it had satisfied the Johnson condition, then we would have created an equivalent 2 machine problem, and then we would have solved a 2 machine problem optimally using Johnson's algorithm. Now, it does not satisfy the Johnson condition, but nothing prevents us from applying that rule to see what we get, so let us try that.

(Refer Slide Time: 32:31)



So, let us now create an equivalent 2 machine Johnson problem by adding, so this becomes M 1 plus M 2, this becomes M 2 plus M 3, J 1, J 2, J 3, J 4 and J 5. So, this becomes 34 18 plus 16 18, 18 plus 12 30 24 21 33 and 35 34, 34 and 31 32, so now, we make the Johnson sequence for this, the smallest one is 21. So, J 2 goes here this goes the next smallest 1 is 30, so J 1 comes here 30 the next smallest 1 is 31, so J 5 comes here the next smallest 1 is 33. So, J 3 comes here 33 is on the first machine, so it comes from the left and the only 1 available is J 4.

So, when we apply the extended Johnson's algorithm for the 3 machine into this, knowing fully well that it will not or may not give the optimum solution, we get a sequence like this. Now, let us find out the completion time for this sequence, so CT is here the sequence is J 5 first J 3, J 4, J 1, J 2, now the make span associated with this is J 5 starts at 0 and finishes at 15, at 15 it goes to M 2 comes out at 31 15 plus 16 31, at 31 it goes to M 3 takes another 16 and comes out at 47.

Now, J 3 is the next 1, so J 3 starts at 15 on M 1. This 15 takes another 13, so comes out at 28 has to wait till 31 for M 2 to be free, takes M 2 at 31 finishes at 51, 31 plus 20 by which time M 3 is free. So, 51 plus 15 is 66. Now J 4 comes third M 1 is available at 28, so starts at 28 finishes at 28 plus 19 which is 47 waits till 51 for M 2 to be free, goes to M 2 at 51 takes another 15 and finishes at 66. Now, exactly at 66 M 3 is free, so it takes M 3 and then takes another 19 and finishes at 85.

J 1 can start at 47 because M 1 is free at 47 takes another 16, so a 47 plus 16 is 63 - waits till 66 for M 2 to be free, goes to M 2 at 66 takes another 18 and then comes out at 84, but M 3 is available only at 85. So, goes to M 3 at 85 takes another 12 units, so comes out at 97. Now J 2 is the last job it can go to M 1 at 63, starts at 63 takes another 14, so finishes at 77 waits till 84 for M 2 takes another 10, so 94 waits till 97 on M 3 takes another 11. So, gives us a solution with 108, so the make span associated with this sequence is 108. Right?

So, what we have done right now is we have simply applied the Johnson's extension algorithm, to a 3 machine problem. And created an equivalent 2 machine problem, with M 1 plus M 2 and with M 2 plus M 3, now if we go back and revisit what we did in the palmer algorithm, we essentially put a 0 weight on M 2 which means we looked only at M 1 and M 3. So, nothing prevents us from actually using the palmer idea again into it, and take only M 1 and M 3 and try and get an equivalent 2 machine Johnson problem and apply the Johnson's algorithm on it.


(Refer Slide Time: 38:12)

LB:103 J3-J5-J4-J2-J1

	M1	M3
J1	16	12
J2	14	11
J3	13	15
J4	19	19
J5	15	16

	M1	M2	M3
J3	13	33	48
J5	28	49	65
J4	47	64	84
J1	63	82	96
J2	77	92	107

J3	J5	J4	J1	J2
----	----	----	----	----

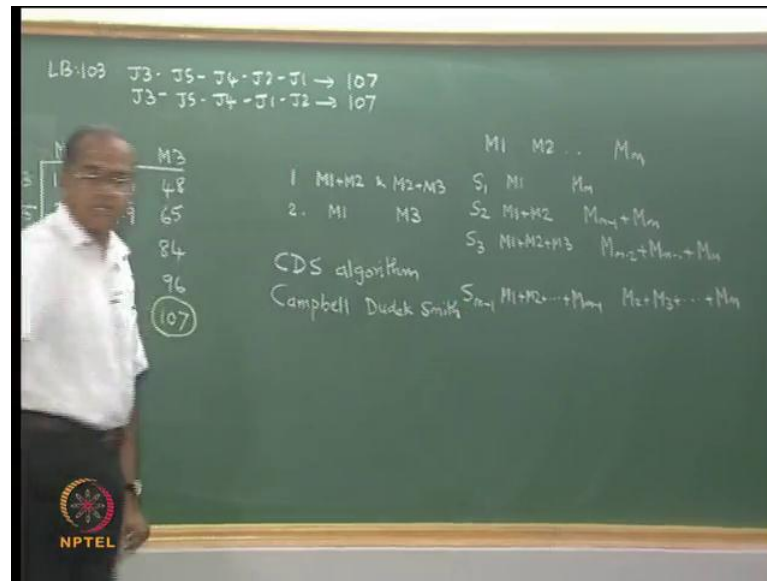


So, right now we will look at only with M 1 and M 3 let us formulate another 2 machine problem here only with M 1 and M 3 let us formulate a 2 machine problem with this. So, once again J 1, J 2, J 3, J 4 and J 5, so we get 16 12, 14 11, 13 15, 19 19, 15 16, now let us make another junction the smallest one is 11. So, J 2 on the second machine, so comes from here J 2, the next smallest is J 1 once again from the second machine, so J 1 comes here.

The third smallest is J 3 which is on the first machine, so J 3 is here the next smallest is J 5 which is here and then J 4 comes here. So, let us quickly find out the make span for this sequence, so we write M 1, M 2, M 3, J 3, J 5, J 4, J 1, J 2, so very quickly writing the completion times. So, J 3 is 13, 13 plus 20, 33, 33 plus 15, 48, J 5 starts at 13 finishes at 28 waits till 33 finishes at 49, M 3 is already available 49 plus 16 is 65. Then we have J 4 starts at 28 plus 19 finishes at 47, waits till 49 takes another 15 which is 64, waits till 65 takes another 19 and finishes at 84.

J 1 starts at 47 takes 16 finishes at 63, waits till 64 takes another 18, so 82, again waits till 84 takes another 12 which is 96. Now, J 2 starts at 63 takes another 14 finishes at 77, waits till 82 takes another 10, 92 waits till 96 takes another 11 which is 107, so what we have done in this algorithm is, now we get another solution whose make span is 107. So, between these two solutions, we could pick the one which is better which has 107, so we essentially did two solutions we evaluated in this algorithm.

(Refer Slide Time: 41:59)



So, one of which is for a 3 machine, the first one we did was a Johnson problem with M_1 plus M_2 and M_2 plus M_3 based on the Johnson extension. The other is we did only M_1 and M_3 , and we took the best out of the two and therefore, we get another solution where J_3, J_5, J_4, J_1, J_2 were 107 again. So, we did not get a solution better than palmer, but then we got the same solution as palmer based on this, but we got a different sequence in this.

Now, for three machines we got 2 solutions and we pick the best, now can we generalize it to m machines. So, if we have m machines, we call them M_1, M_2, M_m , now we create m minus 1 sequences for three machines we created two sequences for general m machines we create m minus 1 sequences. So, let us call sequence S_1 as M_1 and M_m take the first machine, take the last machine do a Johnson out of that, which is this, the second S_2 will be M_1 plus M_2 and M_m minus 1 plus M_m .

Take the first 2 add take the last 2 add do a Johnson out of it, the third sequence will be M_1 plus M_2 plus M_3 and M_m minus 2 plus M_m minus 1 plus M_m take the first 3 and add take the last 3 and add and do it. So, as we keep doing the last sequence S_{m-1} sequence will be M_1 plus M_2 plus M_m minus 1, this will be M_2 plus M_3 plus M_m . So, take the first m minus 1, take the last m minus 1 starting from 2 and get it, we do not find the m 'th sequence because that would mean you add all of them both sides we will get the same, so we do not do that.

So, for a general m machine problem you generalize it to find m minus 1 sequences, so if I have 10 machine problem, irrespective of the number of jobs I will evaluate 9 sequences, based on these 9. Somewhere if each of these has a tie in the Johnson, you could get more sequences, but if you have m machines we will end up doing m minus 1 sequences, and take the best out of that. So, in this case in a 3 machine case we are 2 sequences, and we took the best, if you are 8 machines then you would have 7 sequences and you would take the best.

Now, this still a heuristic algorithm, now this heuristic algorithm is commonly called as a CDS algorithm stands for Campbell Dudek and Smith who are the three authors, and this came in 1977. And a very popular algorithm for use to find out the make span for a general m machine flow shop sequencing problem. It is also generally based on experimentation, it is observed that this algorithm gives better solutions than the Palmer's in general. Palmer's would still evaluate only one sequence CDS would evaluate m minus 1 sequences, and choose the best.

Now, let us look at one more heuristic algorithm, and then we to see whether we can get a solution, which is better than 107. So, the goodness is once again 107 minus 103 by 103, so we have not made too much of an improvement for this particular example the CDS gives the same value of the make span as that of the palmer.

(Refer Slide Time: 47:13)

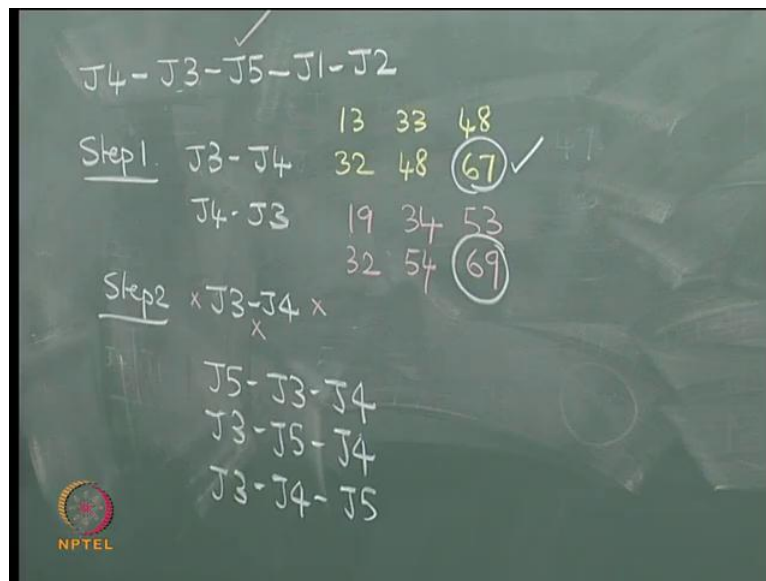
	M1	M2	M3	
J1	16	18	12	46
J2	14	10	11	35
J3	13	20	15	48
J4	19	15	19	53
J5	15	16	16	47

J4-J3-J5-J1-J2
Step 1

Now, we look at one more heuristic very quickly and what we do here is that we start

with, let us find out the sum of processing time of each of the jobs, so 16 plus 18, 34 plus 12 46, 14 plus 10, 24 plus 11 35, 13 plus 20, 33 plus 15, 48, 19 plus 15, 34 plus 19, 53, 15 plus 16, 31 plus 16, 47. So, now we arrange the jobs in decreasing order of the total processing time, decreasing would always mean non increasing order of this. So, the highest is J 3 the next is J 5 I am sorry the highest is J 4, the highest is J 4 with 53, the next is J 3 the 3'rd is J 5 the 4'th is J 1 and the 5'th is J 2.

(Refer Slide Time: 48:42)



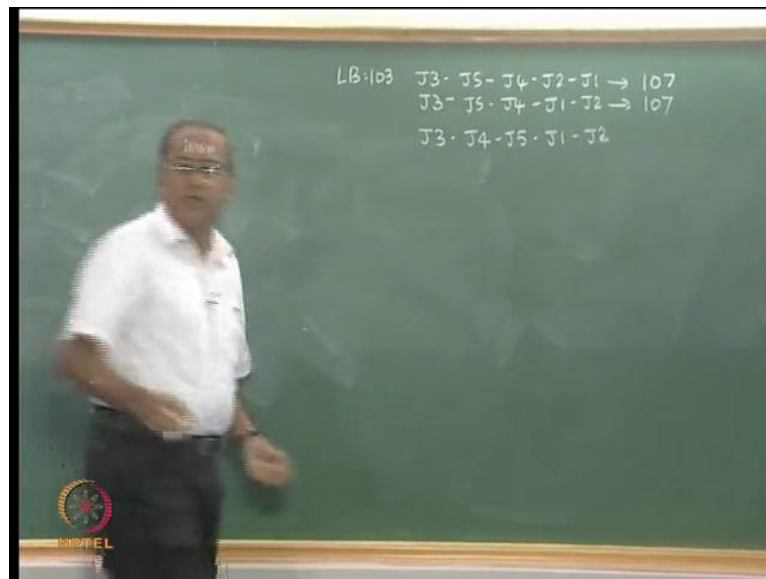
Now, we take the first two of these which is J 4 and J 3, so take, so step 1 take J 4 and J 3, now J 4 and J 3 can be permuted in two ways, which is J 3, J 4 and J 4, J 3. It can be permuted in two ways J 3 and J 4, when I say take the first two if we are taking J 4 and J 3 not necessarily in that order, J 4 and J 3 not J 3 following J 4. Say if we take only J 4 and J 3, there are 2 jobs and they can be permuted in two ways, now find out the make span for the partial sequence J 3, J 4 and J 4, J 3 let us just do that.

So, for J 3, J 4 it will be 13, 33, 48, J 4 is 13 plus 19, 32, 33 plus 15, 48, 48 plus 19, 67 for J 4, J 3 it is 19 plus 15, 34, 34 plus 19 is 53, 13 plus 19, 32, 34 plus 20, 54, 54 plus 15 69. Now, this partial sequence has a make span of 67 this partial sequence has a make span of 69, so choose this sequence, so now, choose J 3, J 4 step 2. So, step 2 choose J 3, J 4, now take the next job here which is J 5, now J 5 can be squeezed in three ways J 5 can come here, J 5 can come here, J 5 can come here. So, you could have three sequences which are J 5, J 3, J 4,- J 3, J 5, J 4,- J 3, J 4, J 5.

Now, find out the make span for all these 3 partial sequences like we did here, and choose the best. So, one of them will be the best at the moment we are not going to show the computation, but one of them will be the best, if for some reason let us say that this is the best for the purpose of argument, after the computation. Then look at the next one J 1 and squeeze it in four positions it could come here, it could come here, it could come here, it could come here.

So, then evaluate four more sequences choose the best, and take the last one and squeeze it in five positions, and then evaluate the best out of these five. So, at the end we will have five sequences and we will choose the best out of these five sequences.

(Refer Slide Time: 52:17)



Now, if we continue with the algorithm then we finally, get a solution 3, 4, 5, 1, 2, so we finally, get a solution 3, 4, 5, 1, 2. Let us compute the make span for this.

(Refer Slide Time: 52:29)

	M1	M2	M3
J3	13	33	48
J4	32	48	67
J5	47	64	83
J1	63	82	95
J2	77	92	106

Insertion algorithm
NEH
Nawaz Enscore Ham

So, we would have M 1, M 2, M 3, J 3, J 4, J 5, J 1 and J 2, now let us quickly compute the make span. So, first 1 is J 3, so 13, 33, 48 followed by J 4 13 plus 19, 32 waits till 33 finishes at 48 goes at 48 finishes at 67. J 5 comes next 15. So, 32 plus 15, 47 waits till 48 takes another 16, 64, waits till 67 takes another 16 and finishes at 83. Then comes J 1 starts at 47 plus 16, 63 waits till 64, 64 plus 18, 82 waits till 83 takes another 12 and finishes at 95. J 2 starts at 63 takes another 14 finishes at 77, waits till 82 takes another 10, 92 waits till 95 takes another 11 and finishes at 106.

(Refer Slide Time: 54:04)

LB:103 J3-J5-J4-J2-J1 → 107
J3-J5-J4-J1-J2 → 107
J3-J4-J5-J1-J2 → 106

m, n	CDS	$m-1$	$\frac{106-103}{103} \times 100\%$
	NEH	n	

$n > m$ $\approx 3\%$

So, the make span here is 106 which is better than the 107 that we got, so we have a make span of 106. ((Refer Time: 54:12)) Now, this algorithm is an insertion algorithm and this insertion algorithm for make span is called the NEH algorithm, stands for Nawaz Enscore and Ham algorithm, which came in the year 1984. Now, this insertion algorithm is found to work very well for make span minimization in a flow shop, and it has been shown to give on an average slightly better results than the Campbell Dudek and Smith algorithm.

Now, if you have m machines, and n jobs CDS will evaluate $m - 1$ sequences and will choose the best, NEH in the end will evaluate n sequences and choose the best, and usually non flow shop problems n is greater than m . So, NEH algorithm would actually evaluate more sequences, than the CDS algorithm either because of that or otherwise it is able to perform slightly better than the CDS algorithm, when we consider the make span minimization.

Now, that we have a solution with 106 then the goodness will become $106 - 103$ by 103 into 100 percent. Now, this is of the order of 3 percent whereas, with 107 it was of the order of 4 slightly less than 3, this will be the slightly less than 4 in the earlier case and slightly less than 3 in this case. So, if we are happy with the solution which is within 3 percent of the optimal, we can take the NEH and use it.

If not we proceed for other methods with the hope that we can get, we may get or we wish to get a solution less than 106. But, then it depends on where the optimum is and for this particular problem instance, optimum is 106 and therefore, NEH has given us the optimum, but then we do not have a way to know that. So, we would still say that NEH has given as a solution, which is within 3 percent of the optimum. So, we have now seen some heuristics to solve the make span minimization problem in a flow shop, earlier in our discussion on sequencing and scheduling, we started with single machine flow shop and then job shop. So, we will look at some aspects of scheduling in a job shop in the next lecture.