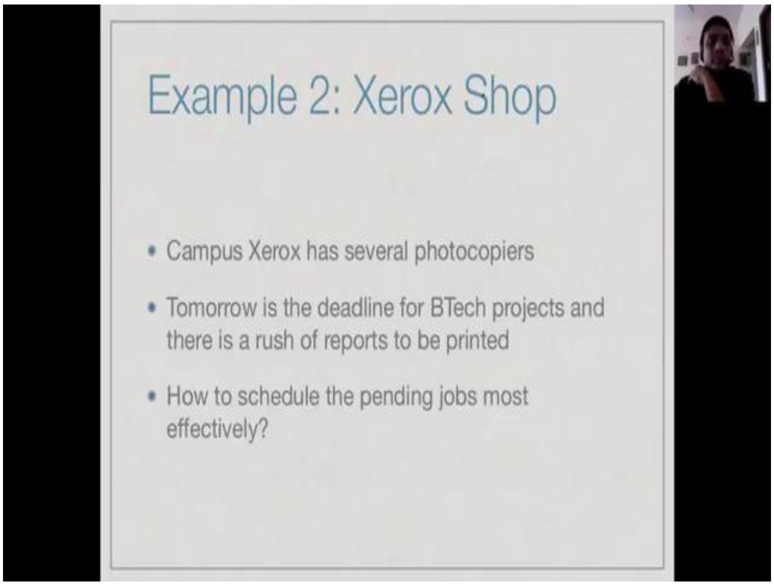
Design and Analysis of Algorithms
Prof. Madhavan Mukund
Chennai Mathematical Institute

Week - 01
Module - 03
Lecture - 03

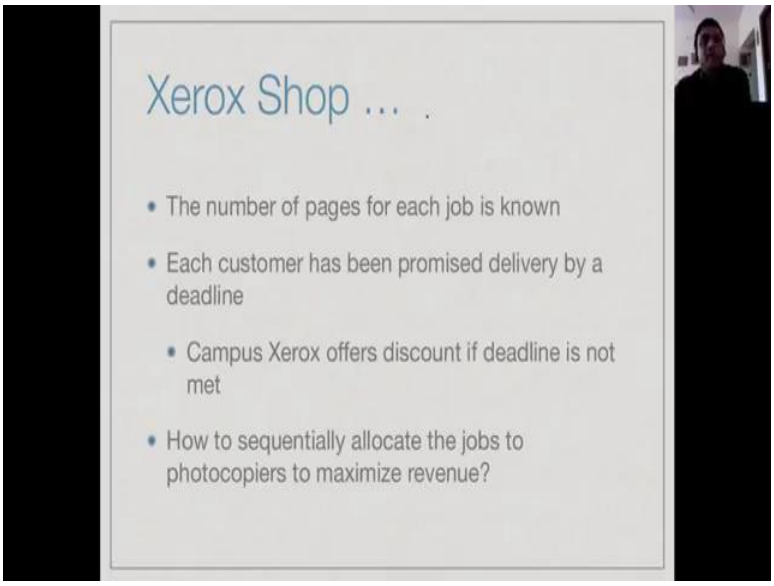
So, our first example was to do with airlines, scheduling, network and using the graph. Now, let us look at its different part.

(Refer Slide Time: 00:08)



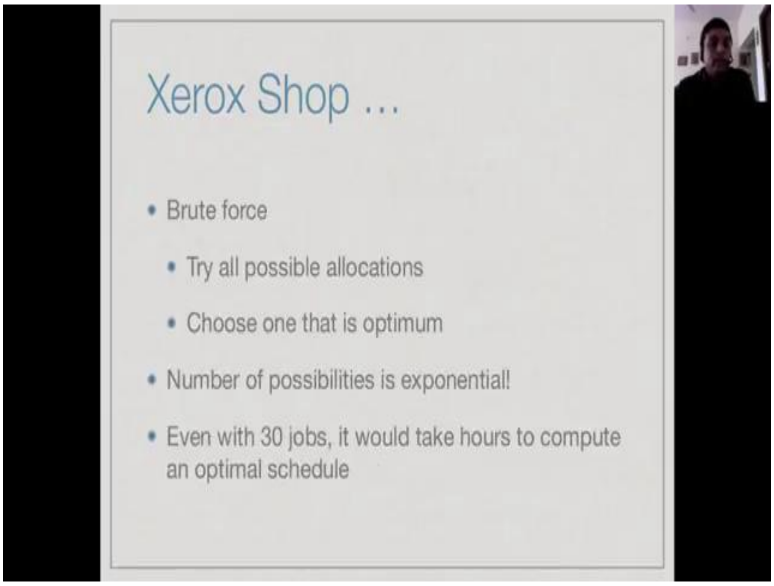
So, suppose we have a photo copy shop, campus Xerox inside the university campus. The deadline for projects is approaching and bunch of student want their projects copied urgently. So, they all go and submit their reports to the photo copying shop, and say we want so many copies made within such a time. So, now the question for the shop is how best to schedule this job. Now, there are many different ways in which this problem can be phrased. So, let us look at one of them.

(Refer Slide Time: 00:42)



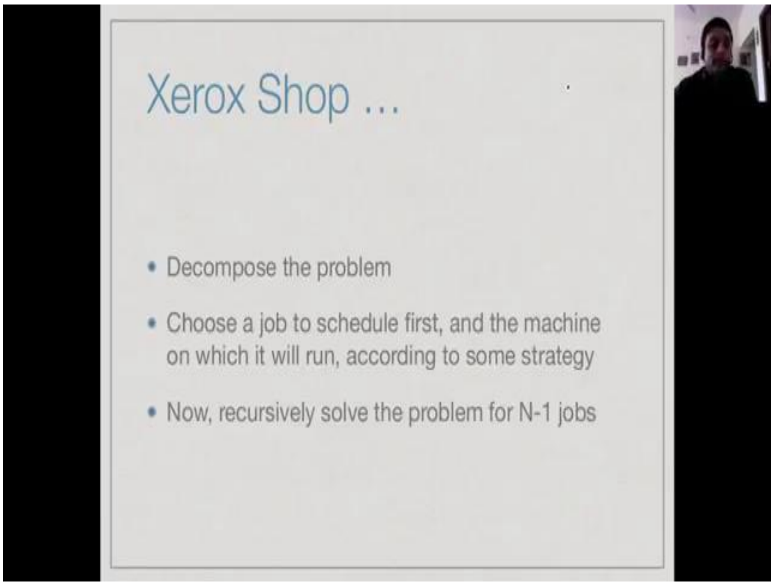
So, suppose the students are submitted their jobs and this shop campus Xerox is competing against some rivals. So, it is offering a special deal. So, it says that it will give each customer a promise delivery time and if does not meet the schedule like a pizza shop, it will give a discount. So, I will promise your report within 6 hours and if you do not get within 6 hours, you pay less. Now, in this time frame there are of course some bigger jobs and some smaller jobs. So, some photo copying jobs will be finished faster, some will take longer, but at the same time they all have to run on the same machines that the Xerox shop has. So, now you can reorder things. So, you could take something which came later and put it earlier on the machine and hope to finish it within its deadline. Therefore, not to have to give discount and take something which is going to take longer and postpone it saying, anyway you are not going to meet the dead line and give up the discount on that job, right. So, the job, the problem the Xerox shop has is how to do this schedule, right.

(Refer Slide Time: 01:50)



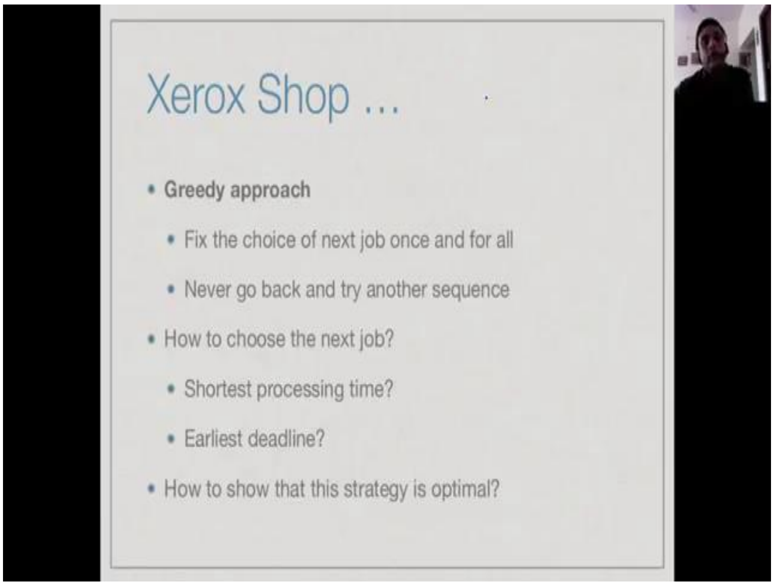
So, there is always at the background what is called group force approach. You can say now I have to allocate these photo copying jobs to the machines in some order. So, let me try every possible order and choose the one which gives me the best return. The problem with this is that it will take a very large amount of time to do this because number of possibilities is exponential. Even if you have just 30 requests pending, it could take several hours to find an optimized schedule, and that several hours might have gone ahead and done some work, so that you got the jobs done and perhaps not optimally at least got some money for it. So, here is where we come to the idea of decomposition, right. So, we can solve this problem by reducing it to a simpler problem.

(Refer Slide Time: 02:30)



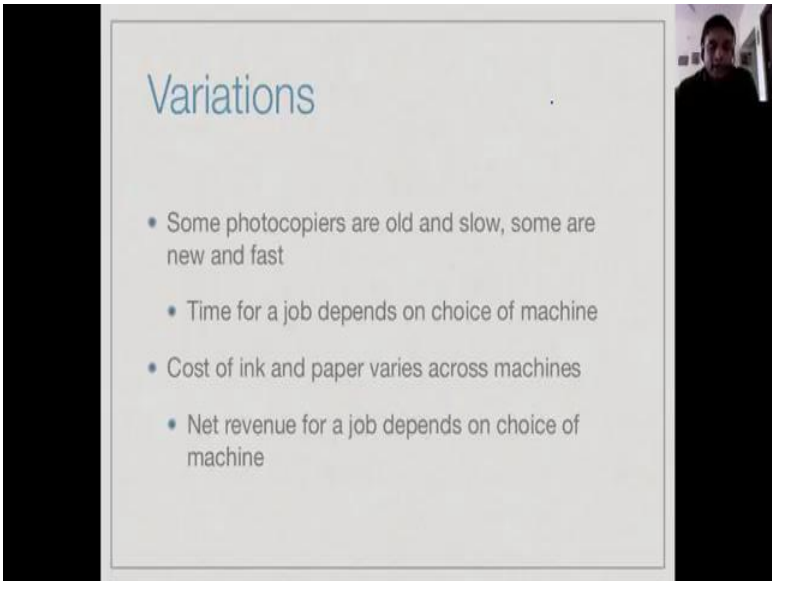
So, supposing we fix one job to run first. If we fix this job to run first, we are left to the remaining jobs of the remaining jobs are smaller in numbers. So, if there was a way to optimally solve for n minus 1 job, then we can pick each of the first jobs, each of the first jobs to be the first job and for each of them determine how much time it takes efficiently if we can do the remaining n minus 1 and choose the best one.

(Refer Slide Time: 03:13)



So, this would give us the kind of recursive solution, pick one and solve the rest and then add the time for this. Another option is to just come up with a strategy. Looking at all the jobs which are yet to be done, we find some criteria by which we choose one to do next. Now, we could have different criteria for which we could choose the one to do next which has the least number of pages that you take the shortest time to process, or we could take the one to do next which is closest to it dead line that is one for which we are most likely to miss finishing it in time and having to give a discount. So, for each of these, we could have a strategy which would tell us which job to do next without looking at all the possibilities of doing the other jobs, but then we have to justify to the strategy that we have chosen is actually optimal. Is it better to choose shortest or (( )) earliest deadline or there is yet another criterion, and how do this criteria justify the choice. Will I always get the best possible return on machine by choosing this strategy.

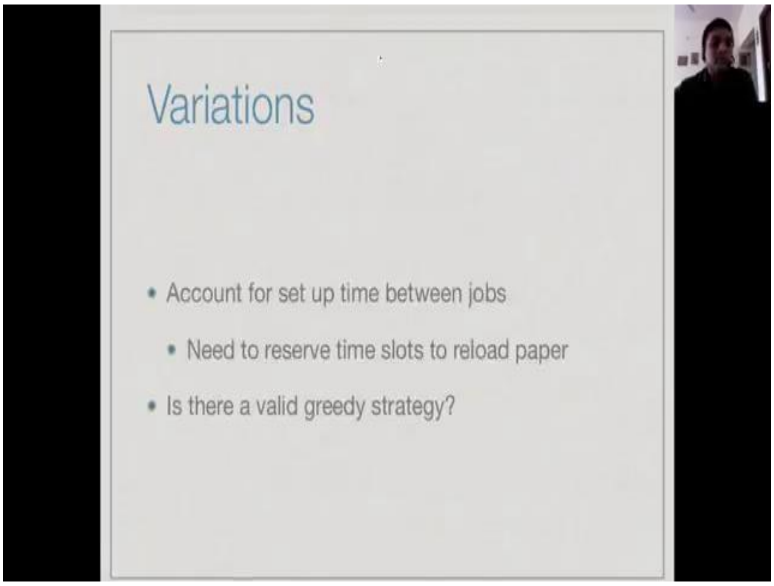
(Refer Slide Time: 04:17)



Now, as we saw with the airline network problem, the basic problem has many different variations which are possible. For instance, if we assume that the shop has many photo copiers, it is reasonable to assume that some are new and some are old. So, some that are new one may work faster than the ones are that are old. Therefore, the time that will take to finish a job depends on which machine we put the job on. So, if you have this additional competition, thus the strategies that we chose for all types of machines that are uniform, still old, then we have to look at the different strategy.

The other factor is of course that the cost of doing something varies across machines. So, if we use a machine; we use some resources, we use some ink, use paper, we use electricity and this cost may vary from one machine to another. So, now the question becomes related to the first question, the previous question which is that now if I split my job across machines, it might not only take more or less time, it also may cost the shop more or less. So, the actual revenue of the shop realizes maybe more or less depending on which machine it chooses. The other thing that we might want to keep in mind is that a machine cannot run indefinitely without having to be stopped for some time for maybe some maintenance, for loading paper, for something. So, we cannot realistically assume that every machine is continuously available.

(Refer Slide Time: 05:29)



Now, under all these situations, it is still a valid greedy strategy or we have to do something else. So, you see the general idea. The general idea is there is a basic problem with some constraints which you want to solve, but that problem can be amplified or made more realistic by adding several new features. By the time you add a new feature, you have to see whether the solution that you have for the simpler problems still works or the new feature demands radically to new approach and if so how you should get that.