### AL\_week4\_lecture\_2-20240923

说话人1 00:00  
Argument, but this formal argument is just a bit formal. The height of idea is the same. So basically, here we use a function fd to map edges from the algorithm, two edges in the optimal solution. It is fp is leave this map. So basically, what does that mean? Fe means that the edges in optimal solution that share a vertex with he for example, if he is this one, the fe is these two edges. Fe also, if e is this one, then fe is these two yellow edges. Basically, the optimal solution, which edges touch the fp and those edges will be fp all right.

Now, there are complaints. Two claims. The first one is for every edge in optimal solution, there must be one edge selected by your algorithm such that e star is efe what does it mean? It means that you just take out one little edge. You can guarantee there is at least one or maybe exactly one red edge touching it. That's the first thing. In other words, it's not possible to have a red edge or have a yellow edge, which have no connection to red eggs. That's not possible.

Why is this true? Let me draw something yet, but I don't have a yellow color. I just pretend this is yellow. Okay? This is yellow. This is a yellow edge, an optimal solution. What's the red? I do? Have a red. There's some red edge somewhere, but it does not touch the yellow dash. It's quite good. So it's not possible that one edge multiple solution, it does not touching, ravage, selected by your characters. It's not possible. Why? Now, what is the argument? Is the following.

Let's look at the moment when this note comes into, this note comes right. According to your algorithm, you should map or match this node to a partner on the right. You should match this one to somebody on the right? But it turns out you didn't match it, right? Because if I match, there's one red edge coming up, right? Then done. The result is the proof, right? But it turns out that in your area didn't match this person to some to anyone on the right. Why is that? It must be because of what? When it comes, right? This look, that comes, but you didn't match this person to anyone on the right. The reason, by the way, there do exist one edge like this, right? But you didn't choose it. Okay? You also didn't choose someone anyone else, but at least you didn't choose this person, right? You didn't choose it. It didn't match it. Why? Yeah, so it must be the case that this guy is already matched, right? Taken by some other people.

Then what does it mean by this time taken by some other people? There is a direction like this. Taken means taken by the algorithm, right? So this person is taken by somebody already. And then that means there is a red s so your conclusion is, if there is no red edge going up from this point, there must be a red edge coming back from this one. Anyway, there must be some red edge touching your wretch, right? Touching optimal solution. Think about it. Basically, optimal solutions, edge, right? It has to somehow overlap or connect with somebody. Otherwise, there will be contradiction, right? Otherwise, the algorithm is not performing, is not really is not really the right thing.

This is the first point, the second time, by the lemma, which lemma, the lemma on the previous page, which means selecting one edge by your algorithm, will only stop you from selling into edges or not, right? Which is one red edge, will only one red edge. It was one red edge, will only touch, at, most, two yellow edges, right? Oki want to touch two yellow edges. That means this fe is less than equal to two. The fe means red edge map to yellow edge. This fe is m os two.

Then you can have some deduction like this. They say optimal solutions, size. How many edges are matching? Optimal solution is at most? Fb the summation of fb although all the edges chosen by the algorithm, right? Because I already show yellow edge must touch red edge, right? Every yellow edge must touch red edge. So I only to look at those red edges, which yellow edges and touch, right? That would be all the optimal solution. You will never miss any yellow edge.

This one, because fe is, at most two, I replace fe by two. It becomes even larger, right? Okay? And then what's this? How many edges do you have in the algorithm? It's just algi right? For all the algorithm edges, every one of them, they will contribute, but two right ok altogether, it becomes 2 times the solution of everything. That tells you optimal solution is almost 2 times everything. In other words, the ratio is larger than half. You do. Everyone divide by optimal? It becomes 1 over 2 larger than equal to 1 / 2.

Right now, as I told you, this is just aa little bit formal proof compared to the high level up here. We both agreed right in the last lecture. What's make it closer? This is the end of the proof. Now, after this, what do you think I would do in the next slide? We have one algorithm, which is very good. 0 . 5 us, v 40 . 5 competitive. What's next? Yeah, show. You cannot do better than that, right? So you need to show something about. All right. So how do you know about? Here are two vertices, right? B one, b two. This is a job I have, right? Now we need to design some bad example for all the possible algorithms, right? What are all the possible algorithms for this case?

Now, let me give you one of the first. You want. You want to connect to both, the one and each. So what can everyone do? How many cases for everyone you ever can do this? Can do that? Can do. So how many different actions? So everything can do if two, you paper order, you want to be one or you want to teach it right now. Two is a good answer, but i'm not a 100 % correct answer.

The airport we also do some thing else. Zero. So everyone knows they have no match. You want to any one of them, right? This is also one possible action from the area, but this option, this option actually is not good. Why? Because if you do not match it to any b then our performance is very bad. If you are the adversary, you just stop the instance.

Now, you say, in this instance, optimal is one edge matching, but you ever only match zero, right? Ratio is arbitrary back zero. This first case, no, good how the second is matched. You want to be one. Another case that will be on the right match, you want to be. Okay? So if you want, it's nice to be one. Then what will you do? If you design the batteries? That's what we do. You need to introduce another you, right? I introduce another you like this, right? Because you match, you want to be one. So I make my second node only connect to be one. And then you cannot match anything else, right? So basically, you are designing something which will make the african regret about its earlier decision, right?

In this case, also as well. Optimal. Sorry, I think too long optimus one, this right? Ok two x but you have one match one edge. What would I do? 0 . 5, right? And similarly, if you it's on the right, it's just symmetric, right? Because we won't be true. They are indeed symmetric. If you choose to match, you want to be true. Are you just connect u two to d two? Again? You cannot match anything else, right? Only describe edge. The optimal solution, again, has 2 days, right? Optimal solutions, I guess. One is your algorithm. Two is optimal solution. Again, it's one of the two. No matter which case, right? Basically, we have the three choices for each choice. I have a future note coming in, making the optimal solution be two, right? And then go out with only one. So you cannot be better than 0 . 5, no matter whichever, right? This finishes the low down proof, actually, very easy, right? This is a very easy example. So much about matching.

Now, next, we'll talk about the third and also the last deterministic water. And the last classic example. Okay. The last example of scheduling, we need a bit more preparation. So there's 1 page having a notation about scheduling. It's worth paying attention to this, because we have another lecture about scattering. If these symbols that will be easier for you to understand those things. So basically schedule each one, it means that you have a bunch of machines.

Okay, I guess of these machines, ok and then you have some jobs, the jobs will always change ok you need to assign jobs, machines, some jobs, and rather change subject to the machine. Okay? This is the basic scenario of schedule assigning jobs and chips.

Now, every job has a so called size or processing time for pj for example, this job one job can last one hour, and the job is 1/2 an hour. Is that right? This is the jobs, execution, time of science. In the general case, each machine has a speed as mine, so which means it's a job. The size of 1 hour is put on a machine with c two, then it only needs half an hour, right? Because the machine has to be too. Okay. That's why I see the actual processing time of job j on machine. M is pj over si the size of the job divided by the the speed of the machine.

That's the actual running time. Okay? Now the schedule will assign jobs to machines, and then you have a way to compute the load of a machine. What's the load of the machine? The local machine is all the jobs assigned to the machine. You add their signs together. This sum divide by the machine speed. That's the load of the machine. Okay? All right, just like this. All right. The summation of all pj such that you match this job change machine I okay? And divide by the speed of machine I everybody has a little. Now, what do you care about the largest load? For example, this machine told us 2 hours. This is 3 hours. This is 1 hour. Now the 3 hours will be bottleneck. That's the longest time one machine will take to finish the workload, right? If you take the last one, that is the so called make span. I think that special work for make span, give make span.

Now, certain problem has many objectives that makes sense will be one very important objective. And that's the objective we are talking about today. Okay, so let's see. This is basically what we explain, right? So let's see some examples. Again, here is a so called simple, really area, but this time is the truth is not basically not really imagine ok so this is simply add to this one, but before you look at this one, I higher, permanent. What do you think the greedy will be? For example, if jobs are like 1 × 1, right? This company, you have to assign it, by the way, you cannot wait for 5 hours to arrive and then assign. Can I do that? Okay? One job arrived assignments on the chain, second job arrived assignments on the chain. This is what you need to do. Now, what's that reading? Can anyone find whatever you have a new job coming in? Which machine do you assign it fast? Or I look at smallest order, you will assign the job to the smallest load machine, which means you assign to the change, finish the insist of earliest ok so let me show you the picture ok so basically, at the very beginning, it doesn't matter which you assign a job to.

And then the second job comes, there will be assignment she want. Why? Because she has nothing on it. A machine has something on it. Machine one is ready earlier than 22. If you assign a second job to machine one, and then we have to assign a third job here again, one or 21, I guess the local machine one is shorter, right? Assign a new job to machine one, right? And then how about the next, by the way, we are talking about the ip machine? All the machines have speed one. So size is the same as the process as the processing time they got the same.

Now. What's the next job? Yeah, two, very good. What's the next job? One, good, and then two books, sorry. And then two, and then one, right? At any time, you just check which machine finishes. It's already assigned jobs early, and then that she will be the one. You assign a new job to, right? Good. Now hear about the algorithm, right? So now let's see how good this kind of case and this one will be given to l one, right? Okay. Now, it's a show this algorithm can achieve two minus one over m competitive issue.

Now, whenever you try to prove some racial, the first, very important thing is to figure out how bad optimal solution. It doesn't need to compare your efforts solution with optimal solution. But here, what is optimal solution is not very straightforward. You don't know what optimal solution is, right? You need to provide about for the optimal solution.

Now, let's guess examples. This is a picture consisting of all the jobs, right? Now, the first, very easy, but for optimal cities, this is a high size optimal statement. C pi star is a big stand in optimal schedule. I claim this optimal solution is at least one over m the summation of all the pj why is that? Look at this picture. Look at this picture. C this is c pi star. Now, suppose this optimal solution. I told, you imagine, right? Or pretend this is optimal solution, optimal solution, c pi side there, because it makes time. What is the submission of pj easy, right? Especially pj just. So submission of these and these, right? Right? You need to finish everything right. Now, from this future, what can you say? This machine? It makes sense, fully utilized, because the job just finishes here. But for the first machine and third machine, there's still some gap there. Then some item machines are not doing jobs, I think, not doing jobs.

You can see very clearly c pi star times the number of machines is not an unequal level. These right, tj summation, c j c pi star, multiply m right? Multiply m is at least the summation of all the teacher. Is that, right? So basically, you can just understand it in a way that your optimal solution, the best you can do is to split all the workload very easily across all the genes. This is basically, as long as you cannot split like this, your baseband will be a bit larger, right? Average workload is a very ideal, low about for the optimal solutions makes them right. This is one thing. How do we need? Another? One thing is not sufficient. We need another thing now, although that is not bonus, I hope you didn't look at the slide yet. So what's your guess? We need another robot optimal services. There is at least something else. Was there something else? Or in other words, this fund sometimes will be useless. So when will this fund be useless or will be not that effective? Any good one?

Now remember just what we say. If you can ideally spend all the clothes to and machines, then this will be quality, right? When this is not equality, it means what you cannot ideally split, right? There's another extreme about not idealistic. Yeah, very good. So there's a job very, very large ok and there's another job very small. You have two machines. What can you do? The only thing you do is put the large job on 2 months, a small job to do, but it's far from the average, right? Ok this problem will be almost useless. Therefore, we need another gun, which is this. The big span is at least the largest process in time or largest size of china, right? Because the big the largest job you cannot split it, right? You cannot split into different machines. Okay? Yeah, with these two bounds are going to prove an easier thing, although the bound 9 years to - 1:00 0 m i'm going to prove the case when two machines and it wasn't easy.

So i'm going to prove when this part is not here, i'm going to prove to a conversation to approximation.

We're too competitive. All right? That's easier for sure, right? But let's take the one machine which gives you the largest finishing time, which is an expense, right? And remove this job. So I I don't need this purple job. This purple job at the time when it is added into the solution, meaning, meanwhile, in this time now, at this time point, the purple job is given to machine two. Why is it given to machine two? Because of the finishing time. Here is earlier than this and earlier than this, right? I think that's why you put the purple job here. So you may spend, I computed in some way like this. Yeah. This part. Li pi minus pj skill, the expand or the logo machine I minus this pj right? This part is less than or equal to average workload. Is that correct?

If I take this time count, I tell you this time point. If I multiply by the number of machines, then it's less than equal to the total workload. Is that right? Because every machine is busy now. At this point, every machine is busy. So the workload here is part, right? Is fully utilized. It's less than your total workload, right? It's your less than a total sum of the pj because no machine is either, no machine is being have free. Before this time back. What is submission of p submission piece plus this? And to this, right? Plus everything here is submission of p this part, this time, this one, multiply m is, at most, the total worker for this one. Any problem? The special moment, right? This special moment is less than or equal to the average local. It's okay, right? Because after this moment, all the machines are the same way that we just, but they didn't finish yet. So the job, the jobs being done by the machine are not yet the total workload, right? Isn't finished.

Now, with this, it means you may spend, which is equal to li pi minus pj plus pj maybe this will pass. This is just an expand. This is the longest machine time, c part, the main span of this solution, is equal to li hat. And then I multiply and subtract the pj and plus pj it's the same, right? Because this is zero. And then this part is, at most average, what is the average? That is the most optimal solution? How about this part dj part? Pg is less than this, right? Okay. This one is definitely the optimal solution. It's less than able to do part one plus part two. Each part is the most optimal solution. So they act together these animals 2 times up in the solution, right? So that means your next bend, your solution is almost twice the optimal solution. That means the ratio is too right. The approximation rate of competitive ratio is too right.

So much about the easy proof. During the break, you can figure out how to get this minus one over n it's not difficult because here we are throwing away many jobs, right? At least we throw away this job. We didn't calculate. If you calculate the scheme, probably this one. Okay? Any questions here about the half of them?

Now, if no question, remember, as usual, we are going to prove water, right? You cannot do better than something, right? You cannot do better than something. Well, we are using m equal to two as an example, two machines, two machines. You see, this theorem tells you, you can do 2 minus half, which is 1 . 5. Competitive. Right? So now on the next page, they are going to show you cannot do better than 1 . 5. Competitive. Okay? How to do that? So again, it's not difficult. Basically tastes, for example, I can do two jobs of unit plans. Next one ok side one, I doing two jobs. What can you offer? What can you ever do when you see these two jobs?

Now you may challenge, you told us this one by one, right? George come on about ac to george wellman. Let me give you an algorithm, a a little bit benefit, right? A little bit more power and allow you to see, too. Okay. That would be, what can everything? This is everything we ask. How many questions? Two choices. Remember, the two machines are the same. There are two choices. What choice is you put them on the same machine? Right? Second choice. Yeah, different. Good, very nice. Let's see the case of the same. If the algorithm for the two jobs on the same team, what would you do? If you are atmosphere? The bedtime normal jobs? Because if I do not give you any more jobs, your solution is too makes sense, too. The ultimate solution is what? One. Put them on different sheets. It's only one, right? So you already have a very bad issue, too. Okay, good. Now you can also, so this one we already analyzed, right for this issue. Achiever rate, can we achieve ratio? Too?

Now, if you are everything, put them on two different sheets, now remember our philosophy. You try to make the algorithm regret, right? Meaning that globally, this solution is not good, but if this solution is not good, which one should be good? It's a different machine solution by the algorithm. It's not good. Then which one is good? Same is good, right? But how can you do the same? Machine solution? Good. You need to give the owners a large job, right? You give the owner a large job. Like this, you give it a job with that, too.

Okay. Now in this case, you see optimal solution definitely will do for the job that to here right here and put this job up. That will be give you an experiment too, right? But there is only 31. That was 233, 1 + 2. Because this new job, too, it can only put either on g 1122, no matter which machine is 33 ÷ 2, this is 1 . 5, right? Remember, I we said there are only two possibilities for reality. One gives ratio two. The other gives ratio 1 . 5. Nobody can do better than 1 . 5. This is the the end of part one. Any question? I believe you might have a question. For example, how are you going to be tested return of 5 % about this? That's a very tough question for me, either.

Anyway, I believe for this kind of simple analysis, you should be able to do it, right? I think everyone agrees. This one is simple. The match. One, the lower bound is also simple, right? It's just 1 page. It's very easy to understand. For these days you should be able to know, but i'm not telling you, but I will not ask you. These examples. I will test you some new scenario, but analysis should not be too difficult. This is a some potential customer have the answer. Okay, all right. Now I guess we will not explain the second part. All right? And to let you know what we are going to talk about in the coming. 1 hour. Okay? Now the second part is about the online decision making when you have some other uncertainties.

Now, the very famous problem is called secretary problem. What is secretary problem? Secretary problem is that your company decided recruit, hire some secretary, right? So there were candidates coming in. You can interview them. You can decide whether to give them. All right. We won't have one voter, only give one person. All right. But different from the real world in the real world, usually I will arrange interview after the interview. I have all the results. I choose the best one. Okay, so that's very good. Now, although this is a real case, but it's not very practical. What does it mean? Now? The advocates sometimes come from about support or getting some water right.

Now, maybe the first candidate, if you tell the candidate, please wait until december the 30. The first, when we tell you the result, maybe in the middle, that candidate will get some other offer, and even that he will give a leave, right? He will not leave it at all. Ok so a safer way will be to give offer right after the interview. So you interview somebody, you need to tell the candidate right away whether you take the candidate or not.

This is something that will also happen in reality. All right. Now, maybe about interview, it is less persuasive. Then let's use one another example where so let me see which is established. I give you. So suppose you are trying to, actually, this is a famous example. Suppose you are trying to find an important that they the important other via some by some experience.

And then we need to decide after you spend some days with cable there, right? Okay? You cannot say I do multiple candidates at the same time. That's not good. Ethically, not good, also, practically also not feasible. So it's sequential order. All right. Now, up to one, you need to decide whether you consider the good fit, right? Same number, like secretary. Right? After interviewing one candidate, you need to decide, by the way, whether you will hire this candidate as a secretary. Right?

Now, another more virtual scenario is like you walk in the forest, and then there are some treasures along the way, but you only have one bag. You can only take one treasure along the way. All right? And then you cannot go back and say because I think the earlier on this matter, can I go back to take the earlier one or not? Do that? You go back earlier, and then you don't have sufficient time to walk out of the forest. You have a little time only keep walking, so no time to go back. All right. So again, it's 1/5 out of my scenario where after the decision you cannot revoke it, you cannot say. Because I see a better candidate, I have to fix the earlier offer. No, you cannot do that. Okay? Then her reputation, all right, so much about the background. Now, what about the problem? The problem usually, in the all night problem, we talk about the so called adversary, right? We try to design a very bad sequence for the hour. But here, if you are allowed to design bad sequence, then probably algorithm cannot do much.

Now, one first thing. Here we assume you have n tennis right? Now suppose you do not know this n for example, n is 101,000 ethanol, usually in practice, you don't know crap. But if you don't know, the adversary can let the algorithm behave very bad. Now, although in the security problem, is to see that better, just to see that better. So if the atmosphere is allowed to change this end, then no matter which candidate the areas of peace, the adversary can always find a better one. Right? Which arrives later. And then the algorithm will regret. So that means if you do not know the value of n then the l the industry can be for our decision as bad as possible. Right? Now, suppose you do know the same. Will the s should be weaker now weaker, but not too much weak. If the adversary can control the arriving order, remember, we say these candidates they are radical, which means you can imagine there are numbers, 1234567.

You want to see that one, which is larger, which has a large area right now. But if the adversary can change the data and change the order, as you wish, then again, your effort cannot do much. Why? So? Basically, the album will see as somebody, right? The evidence you have somebody, but then adversely can set the candidate, some of the candidate, not yet by the algorithm to be very good, to be much better. And then because you have a busy idea, so again, your average performance will be very bad, right? Okay? Now, in order to have some meaningful results about this problem, we need to do some or make some assumption about the atoms. What kind of adversary are we assuming? Here? We have the less powerful atoms? This adversary, firstly, we'll let you know what n is. So this will tell you how many candidates will do. All. Right? Second, the adversary cannot control the order of arrival. The order of arrival of these candidates are random. Any reputation has equal chance, equal probability of showing up.

For example, we have three candidates then with 1 / 6 probability, you see 123, 1 / 6. You see 132213231212321. In this case, you can have some meaning with us. Right? Now, before we present online algorithms, you first look at what is offline optimal algorithm. Offline of optimism is very easy, right? You just choose the best one. So not much to say here. The online algorithm here, it's a bit different from the earlier class, because now the arriving order is actually is random. Now we are trying to maximize the probability that the error of them chose the best candidate.

So think about it. Maybe you have 40 % chance to the best, the other 60 % I choose the worst. Don't worry. I don't care about the worst part. I won't tell you a question. 40 % best, 60 % worst. It's better than 30 % best, 70 %, second best. Remember, we only care about best candidate session, probability. I don't care other and other scenarios, choices. Even if you see that second, guess, it doesn't mean it's better than the worst.

So although it's somewhat not so ideal, right? Because for me, if I have 30 % selected best, 70 %, second best, I will say it's pretty good, right? But this problem is not because that this kind of operation is trying to maximize the probability of seattle, the best candidate, other security situations. Let's see one possible strategy. This strategy is very easy and performance. It's not good, but it's not also not extremely bad.

All right. So this is really says, take the first candidate. I know that you will never say this. Some chinese idiots will come into the mind. Language is the first candidate, right? Okay, choosing first candidate. What's the probability of getting the best one? Very good. One over n because every position, the best one showing up in every position, has equal probability showing up in the first is also one on the right people of it. Good. Now we at least have a guarantee, right? The threshold. One of them is the at least something you can do, right? Okay? But that's not so good, right? It's not so good. That's why we need to think about others. Right? Now, the second strategy is really interesting. It actually is used in research as well. So there's advanced characters using this kind of thinking as well.

So what's the second strategy? The second strategy says, i'm going to split the candidates in 2 groups, group one, group, I to be group one. And then I will have a rough estimation about how good these candidates might be, right? And then what is equal to? I will compare them with group one. Are you better than group one? If you are better at it, you can see that actually, there is a good chance that if this is true, it might be best for about only. Right? So there will be a good probability that you see that the best candidate.

Now, the straightforward splitting, what happened? Ha ha. Basically, it says candidates coming one by one, interview, first one, second one, third one, until n over two to see their quality. All right. You do not see that anymore. They are unlucky. They are only test. There are only test data. They will not be chosen. And then second half, whenever you see somebody better than the first half, better than everyone in the first half, you see it.

Right? Now this is, yeah. You worry about what's the probability of getting the best? The probability getting the best is at least. This is not advice. What's these two parts? First of all, you need to make sure the best candidate is the second half. All right? Otherwise you lose it, right? Because it didn't take anyone from the first half. So first of all, you need to guarantee best candidate in the second half. Next thing is not a must, but it's good. The second best candidate is in the first half, by the way, require this. Yeah. Very good. Basically, the second best in the second half, you have a chance to choose the second best, right? But if the second best is the first half, the only one that can beat the first half will be the best.

So that is, if both conditions are satisfied, for sure, you will see that the best way. Then, what's the probability of these two? The best candidate is second half, and is 1 / 2. Why? Because the other case is best candidates, this part, right? Saving chance. This is the half. How about this? Second lesson goes half game over two, right? Half life, because a little counterpart is second best, the second half, right? The same problems, right? So it's half times half. Wonderful. Now, the 25 % chance, you get the best handed, right? That's very good. Now remember, this is a larger than or equal to why not equal to? Just now we said, if you fulfill both conditions, for sure you select the best one, right? But if you do not fulfill these two, prefer, to example, only fulfill this one, not this one, then because you have a chance to see that the best, right? So that's it. Let's take 10 minutes break before we come back to analyze. Okay, that's great.