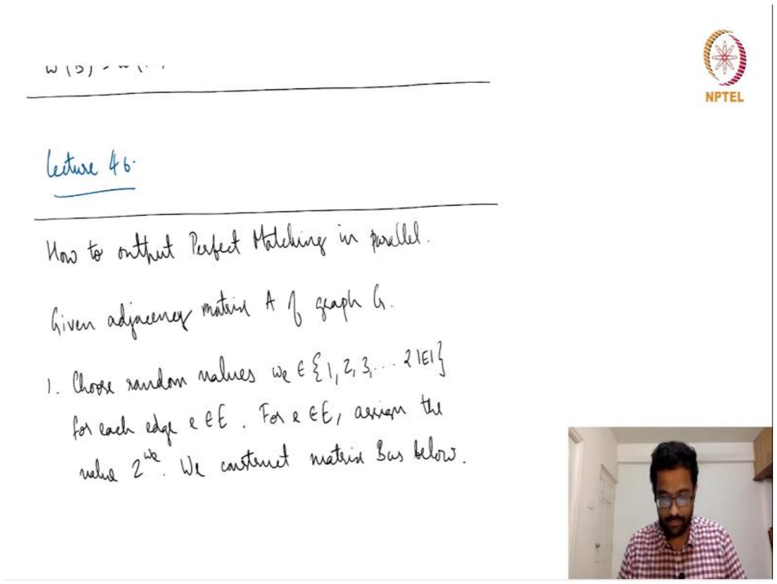
Computational Complexity

Prof. Subrahamanyam Kalyanasundaram
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Lecture - 46
Search Bipartite Perfect Matching is in RNC Part 2

(Refer Slide Time: 00:15)



Lecture 46.

How to output Perfect Matching in parallel.

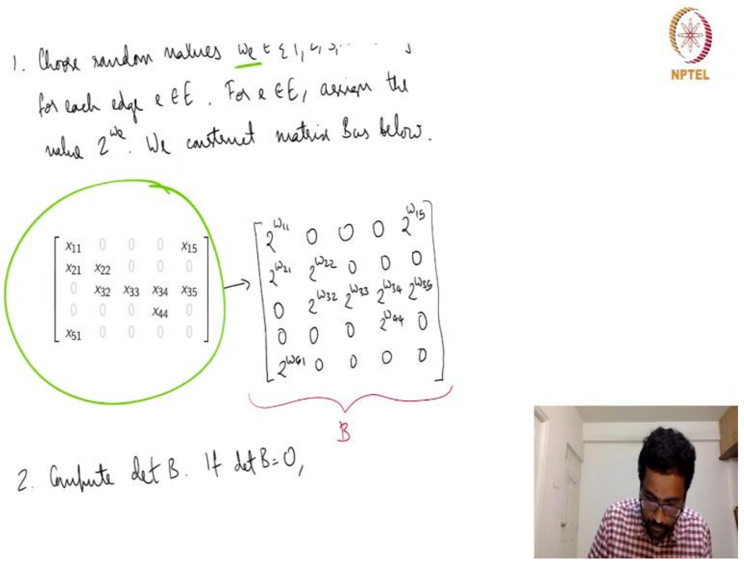
Given adjacency matrix A of graph G.

1. Choose random values $w_e \in \{1, 2, 3, \ldots 2|E|\}$ for each edge $e \in E$. For $e \in E$, assign the value 2^{w_e}. We construct matrix B as below.

Hello and welcome to lecture 46 of the course computational complexity. In the previous lecture we saw isolation lemma which gave us a way to assign weights to elements of a set. In such a way that there is a unique minimum weight set in some family. So, consider any family of subsets of that and whatever be the family isolation lemma tells us that if you assign weights in a certain uniformly and randomly from a range.

There is a good probability of finding a unique minimum weight set in the family. So, in this lecture we will see how to use isolation lemma to output a perfect matching of a bipartite graph in parallel. So, as I said before there are lots of other residual algorithms for computing the perfect matching and outputting them, this is a parallel algorithm. This algorithm is by Mulmuley Valiant and Vazirani who also discovered the isolation lemma as part of their paper.

(Refer Slide Time: 01:31)



1. Choose random values $w_e \in \{1, 2, 3, \ldots\}$ for each edge $e \in E$. For $e \in E$, assign the value 2^{w_e}. We construct matrix B as below.

$$\begin{bmatrix} x_{11} & 0 & 0 & 0 & x_{15} \\ x_{21} & x_{22} & 0 & 0 & 0 \\ 0 & x_{32} & x_{33} & x_{34} & x_{35} \\ 0 & 0 & 0 & x_{44} & 0 \\ x_{51} & 0 & 0 & 0 & 0 \end{bmatrix} \rightarrow \underbrace{\begin{bmatrix} 2^{w_{11}} & 0 & 0 & 0 & 2^{w_{15}} \\ 2^{w_{21}} & 2^{w_{22}} & 0 & 0 & 0 \\ 0 & 2^{w_{32}} & 2^{w_{33}} & 2^{w_{34}} & 2^{w_{35}} \\ 0 & 0 & 0 & 2^{w_{44}} & 0 \\ 2^{w_{61}} & 0 & 0 & 0 & 0 \end{bmatrix}}_{B}$$

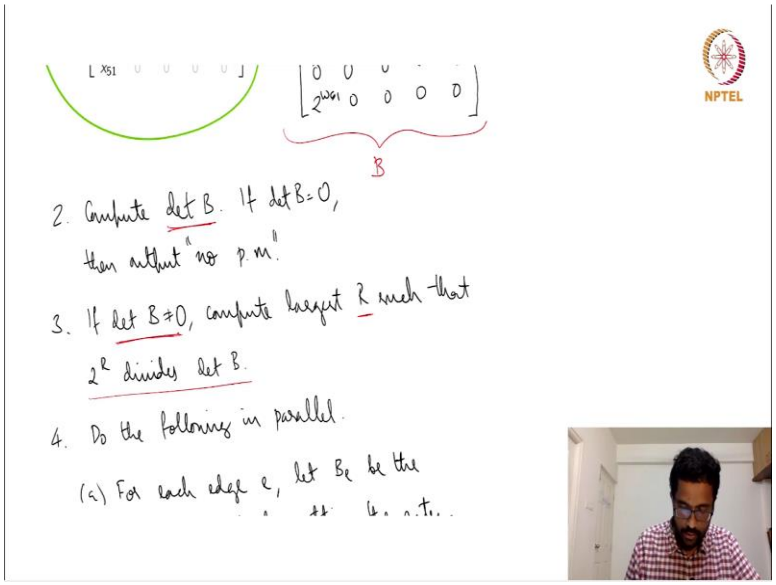
2. Compute det B. If det B = 0,

So, some of the elements of this proof we already saw in the previous week. So, given the adjacency matrix we saw we could make the tut matrix. But in this proof, we will not make the touch matrix, we will use a different matrix. So, we start with the tut matrix or the adjacency matrix, suppose this is a the one that I have circled here in green is the tut matrix. What we do is consider the elements x 11 x 15 etcetera. as some elements of a set.

Now corresponding to x 11, we will introduce a weight w 11 corresponding to x 15 will introduce away w 15 and so on. But each of these weights so each of these entries x 11 x 12 etcetera are entries corresponding to certain edges, zeros correspond to non-edges. So, corresponding to each of these weights we pick random values. So, I will denote them w subscript e and we pick them from the range 1 2 3 up to 2 times the size of e.

So, there are 100 edges in the graph. We go up to the range 200, 1 2 3 up to 200. So, twice the number of edges and then what we do is we pick random weights like this and then you assign in the matrix b you assign 2 power that weight. So, not just that weight we need to assign 2 power that weight. So, 2 power w 11 will correspond to the edge e 11 2 power w 15 will correspond to the edge between 1 and 5 and so on and this gives us the matrix b.

(Refer Slide Time: 03:22)



$[\,x_{51} \quad 0 \quad 0 \quad 0 \quad 0\,]$

$$\underbrace{\begin{bmatrix} 0 & 0 & 0 & & \\ 2^{w_{51}} & 0 & 0 & 0 & 0 \end{bmatrix}}_{B}$$

2. Compute det B. If det B = 0, then output "no p.m."

3. If det B ≠ 0, compute largest R such that 2^R divides det B.

4. Do the following in parallel.

(a) For each edge e, let B_e be the

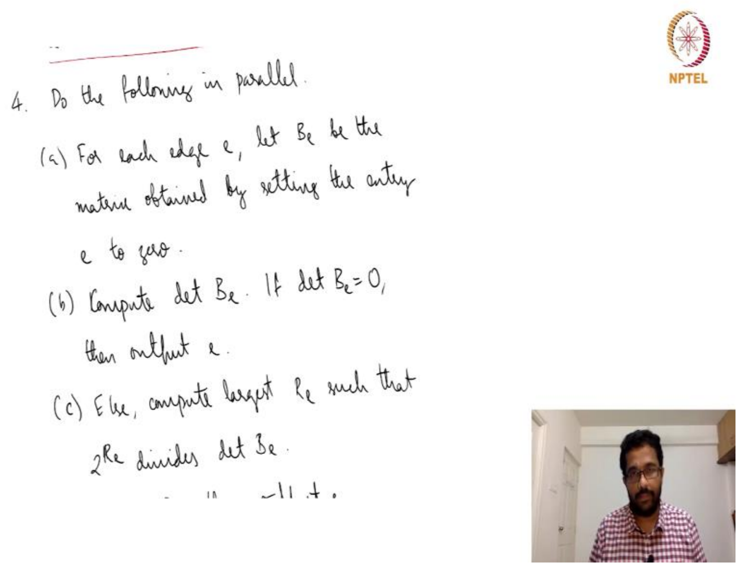
As we saw in the last lecture if the matrix b is non-zero that means there is a perfect matching. But there is some possibility that there is a perfect matching but still somehow some terms cancel. So, that the determinant of the matrix I think I misspoke before, if the determinant of the matrix is 0, if the determinant of the matrix is non 0, then there is a perfect matching. But it could somehow so happen that there is a perfect matching but still the determinant could become 0 because of some terms cancelling each other.

And what we hope to achieve or accomplish using isolation lemma is that the weights will be assigned in such a way that if there is a perfect matching then the chances of cancellation are very very low and that will be assured using isolation lemma. So, what we do is we compute the determinant, so this is first we describe the algorithm and then I will explain why this algorithm makes sense or why this algorithm works. So, compute the determinant of this matrix b.

This is the same matrix above one with the 2 power etcetera. If the determinant is 0 then you output there is no perfect matching. So, this algorithm can still have errors so there could be a perfect matching and we still could output 0. But we can limit the error or we can even compute the probability of error. So, we will see when that happens if determinant is non-zero then we know for sure that there is a perfect matching.

Now the goal is not to decide whether there is a perfect matching or not, the goal is to output the perfect matching. So, let us see how we do that now compute the determinant so of course we already compute the determinant now you check the largest number R such that 2 power R is a factor of determinant b. so, compute the largest number R such that 2 power R divides determinant of b. So, this point it is done serially

(Refer Slide Time: 05:40)



4. Do the following in parallel.

(a) For each edge e, let B_e be the matrix obtained by setting the entry e to zero.

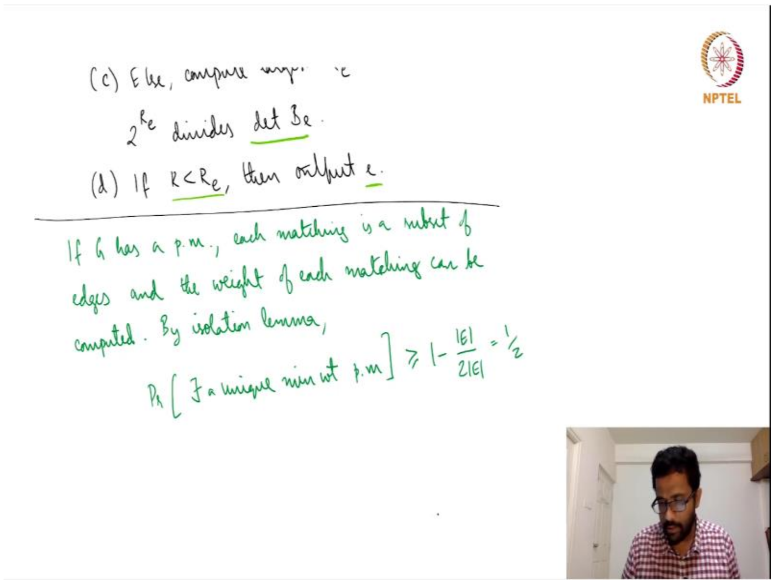
(b) Compute det B_e. If det $B_e = 0$, then output e.

(c) Else, compute largest R_e such that 2^{R_e} divides det B_e.

Now what we will do is the rest will be in parallel. So, rest can be done separately by one processor per edge of the graph. Now what can be done is for each edge in the graph consider. So, what you do is you let us say you consider this edge for instance h 51. So, now you replace edge 51 with 0. So, this h this entry instead of 2 power w 51 will become 0 and this so for the edge e we call be as the matrix obtained by setting that entry to 0.

Now you compute the determinant of B e. If the determinant of B e is 0 you output e so what we will do is we will do many parallel outputs. So, this this branch will only output e or not output e

(Refer Slide Time: 06:45)


(c) Else, compute [illegible] R_e [illegible]

2^{R_e} divides $\det B_e$.

(d) If $R < R_e$, then output e.

If G has a p.m., each matching is a subset of edges and the weight of each matching can be computed. By isolation lemma,

$$\Pr[\exists \text{ a unique min wt p.m}] \geq 1 - \frac{|E|}{2|E|} = \frac{1}{2}$$

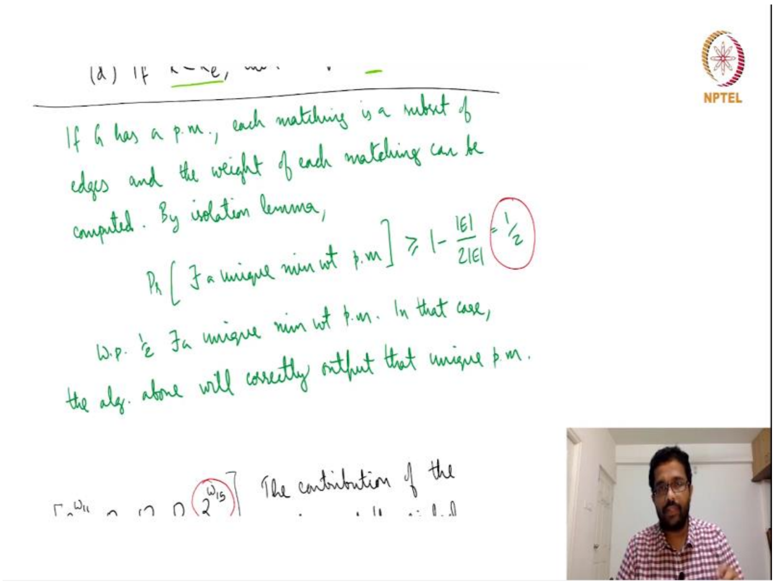
Or it will say no matching at all. In fact, no matching will already have said been said at the earlier part. So, if the determinant of B e is non-zero then output it outputs e, sorry if the determinant of B e is 0 it outputs e, else what we do is you compute again we do the same thing. We compute the largest R e, so this looks kind of like an l such that 2 power R e divides determinant of B e. So, I am just replacing the e's to actually look like these.

Now it looks sometimes it looks more like l than e. So, compute the largest R e such that 2 power R e is a factor of determinant B e. Now so recall we earlier computed this R which was the biggest, so it was where B e was this original matrix that we have here. And B e is the matrix corresponding to where each edge entry or the 1 edge entry is replaced by 0. And then we again computed or earlier we computed R such that 2 power R is the largest factor of B e.

Or the largest R e such that 2 power R is a factor of B e. Now compute the largest R e such that 2 power R e is a factor of B e. Now compare R and R e. And if R is strictly smaller than R e, then you output e there are 2 cases where you output e. One is when the determinant of B e is 0, or when the R is a strictly smaller number than R e. So, notice that from where from in step 4 from where I started step 4, I am just focusing on 1 specific edge.

And this will be done in parallel by different processors for independent processors for each edge. And the claim is that each edge will correctly each processor at the corresponding edge will correctly output e even only if e is part of the perfect matching. So, which perfect matching? There could be multiple perfect matching's. So, here is where isolation lemma helps. Maybe I will just write here. I will just write a bit here to explain.

(Refer Slide Time: 09:18)



(d) If

If G has a p.m., each matching is a subset of edges and the weight of each matching can be computed. By isolation lemma,

$$\Pr\left[\exists \text{ a unique min wt p.m}\right] \geq 1 - \frac{|E|}{2|E|} = \boxed{\tfrac{1}{2}}$$

w.p. $\frac{1}{2}$ $\exists$ a unique min wt p.m. In that case, the alg. above will correctly output that unique p.m.

$2^{w_{15}}$] The contribution of the

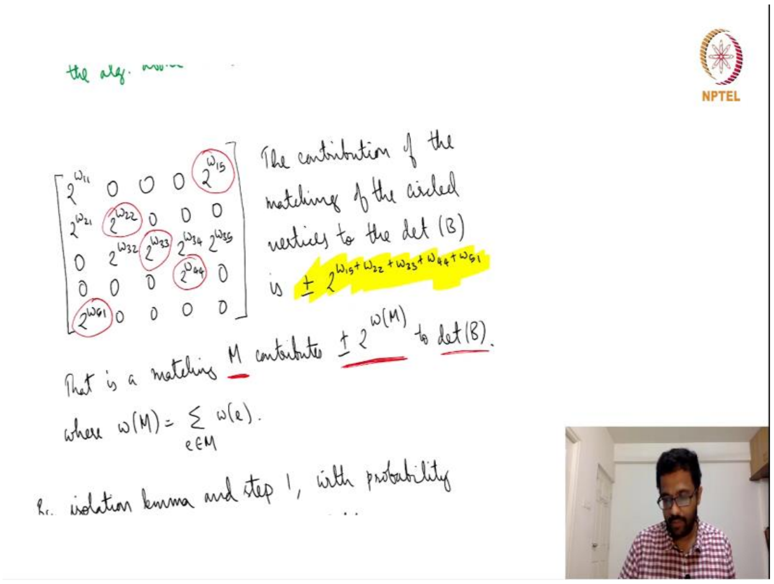
So, what is happening here is that if G has a perfect matching, then there is some then only then the rest of it makes sense. But now each matching can be thought of as a subset of edges and you can compute the weight for them, and the weight of each matching can be computed. So, the base set here has size of e many elements. It is a base set. In the case of isolation lemma number, it was x. It was the size of x was n small n.

Here the size of e is well it is just the size of e and the range here is going up to 2 e. So, by isolation lemma, probability of a unique line, there is a unique mean weight perfect matching is at least 1 - size of e which is the number of elements in the size edge divided by 2 times e which is simply half. So, with probability half there is a unique min weight perfect matching. So, that is with probability half there is a unique, sorry unique min weight perfect matching.

And if there is such a minima, in that case the algorithm above will correctly output that unique perfect matching. So, what the algorithm will do? We will see the proof now; the algorithm will output this unique min wait perfect matching. So, let us see how that happens. So, if the isolation lemma works as an if there is a unique min weight perfect matching this algorithm will correctly output the unique min weight perfect matching that matching.

If it does not work meaning if it does not have a unique mean weight perfect matching then there is no guarantee. So, in this particular assignment the probability of that happening is half but this need not you could choose the weights from a bigger range instead of 2 e you could choose it from 10 e or something. So, the probability will be instead of half it will be 9 by 10 in that case so you can improve the probability of doing that.

(Refer Slide Time: 13:18)



the alg. ...

$$\begin{bmatrix} 2^{w_{11}} & 0 & 0 & 0 & 2^{w_{15}} \\ 2^{w_{21}} & 2^{w_{22}} & 0 & 0 & 0 \\ 0 & 2^{w_{32}} & 2^{w_{33}} & 2^{w_{34}} & 2^{w_{35}} \\ 0 & 0 & 0 & 2^{w_{44}} & 0 \\ 2^{w_{51}} & 0 & 0 & 0 & 0 \end{bmatrix}$$

The contribution of the matching of the circled vertices to the det (B) is $\pm 2^{w_{15}+w_{22}+w_{33}+w_{44}+w_{51}}$

That is a matching M contributes $\pm 2^{w(M)}$ to $\det(B)$.

where $w(M) = \sum_{e \in M} w(e)$.

By isolation lemma and step 1, with probability

...

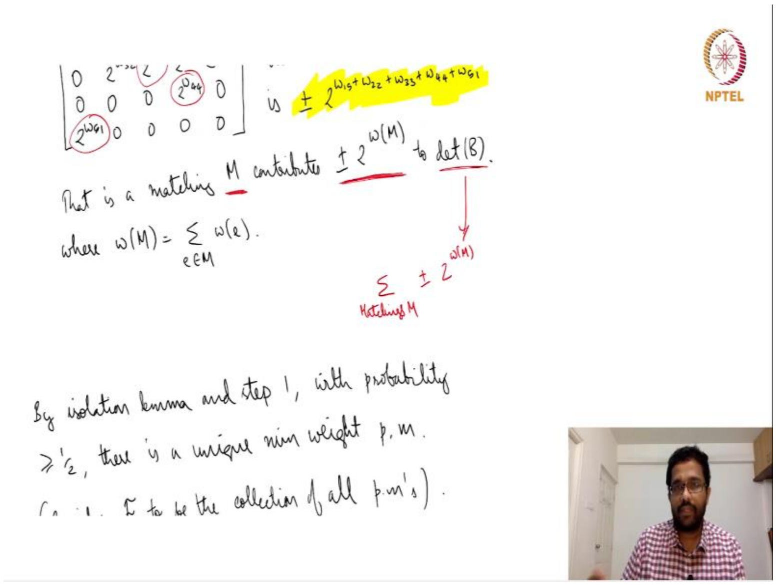
So, now the rest of it, please assume that there is a unique min weight perfect matching and then we will see how it outputs the same matching. So, consider this matrix here, this is the same matrix B and consider the circled edges here. There are five circled edges and these form a perfect matching, now this contributes to the determinant. This is one permutation and the contribution the term.

That is contributed to the determinant by this perfect matching is 2 power w 15 multiplied by 2 power w 22 multiplied by 2 4 w 33 multiplied by 2 power w 44 multiplied by 2 power w 15 or sorry 51 this could be + or - depending on whether this permutation is a odd or even permutation. So, the contribution is this highlighted quantity 2 + or - 2 power w 15 multiplied by 2 power w 22.

So, the product of these 2 power something can be written as 2 power the you can add up the exponents. So, it is 2 power w 15 + w 22 and so on. So, the instead of multiplying the 2 powers you could just 2 power, you could write it as 2 power the sum summation. And so basically it is what is the summation? What is exponent here? The exponent is nothing but the weight of the matching.

So, way a matching m, contributes a weight + or - 2 power the weight of the matching. That is the contribution to the determinant. So, the determinant is just a collection of such matches or the collection of such sums.

(Refer Slide Time: 15:05)



$$\begin{bmatrix} 0 & 2^{w_{32}} & \cdots & & \\ 0 & 0 & 0 & 2^{w_{44}} & 0 \\ 2^{w_{51}} & 0 & 0 & 0 & 0 \end{bmatrix} \text{ is } \pm 2^{w_{15}+w_{22}+w_{33}+w_{44}+w_{51}}$$

That is a matching $\underline{M}$ contributes $\underline{\pm 2^{w(M)}}$ to $\underline{\det(B)}$.

where $w(M) = \sum_{e \in M} w(e)$.

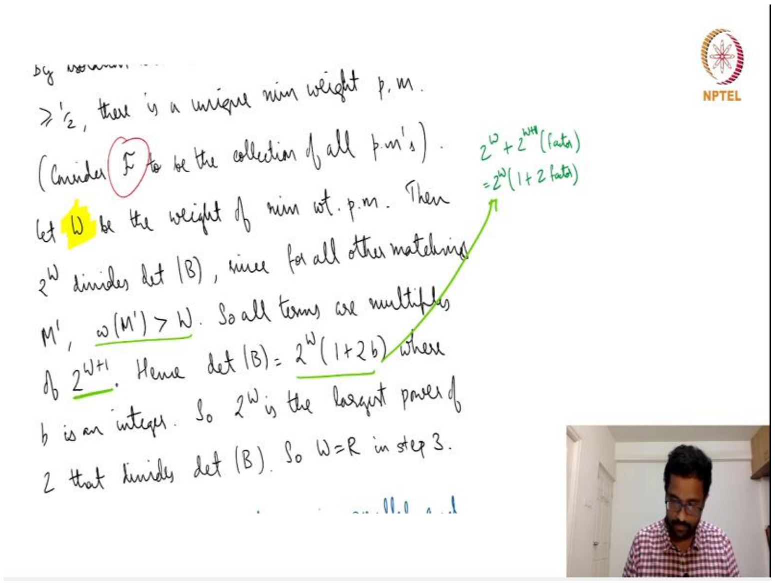
$$\sum_{\text{Matchings } M} \pm 2^{w(M)}$$

By isolation lemma and step 1, with probability $\geq \frac{1}{2}$, there is a unique min weight p.m.

[illegible] Σ to be the collection of all p.m's).

So, just to elaborate a bit further so what I am saying is determinant is, summation of matching m + or -. So, depending on the sign of the matching or sign of the permutation corresponding to the matching 2 power weight of m, this is what the determinant is, where weight of m is just the sum of the weight of the edges.

(Refer Slide Time: 15:38)



by isolation

$\geq \frac{1}{2}$, there is a unique min weight p.m.

(Consider $\mathcal{F}$ to be the collection of all p.m's).

Let W be the weight of min wt. p.m. Then

2^W divides det (B), since for all other matchings

M', $\underline{w(M') > W}$. So all terms are multiples

of $\underline{2^{W+1}}$. Hence det $(B) = \underline{2^W(1+2b)}$ where

b is an integer. So 2^W is the largest power of

2 that divides det (B). So $W = R$ in step 3.

$2^W + 2^{W+1}$(factor)

$= 2^W(1 + 2$ factor)

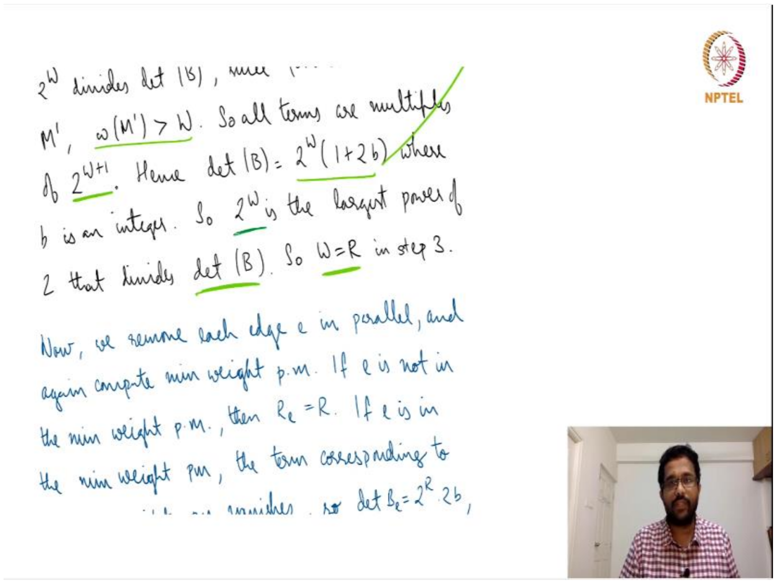
And this is something I already said by isolations lemma and step one with probability at least half there is a unique mean weight perfect matching. Just to just see you can consider F as the collection of all the perfect matching's, now let w be the weight of the min weight perfect matching W Capital W. In that case if you look at this sum over here if you look at this sum that I have circled over here it is a collection of things it is a summation of 2 power weight of some things.

2 power now w, capital W is the smallest rate, that means for all the other matching's m prime. So, let w be the matching m and for all the other matching's m prime the weight of m prime is strictly greater than w. So, again we are in this case where there is a unique min weight perfect matching. So, all the other matching's n prime the weight of w the weight of m prime is strictly greater than w, so that means all the other matching's contribute at least 2 power w + 1.

There could be 2 power w + 2 or 2 + w + 3 or 2 power w + 100 or something, but at least 2 power w + 1. So, the determinant of b, so there could be + - of all this. So, there will be one 2 power w term and the rest all will be at least you should be able to take out. So, what I am saying here is the determinant will be 1 2 power w term + the rest of it all will be multiple of 2 power w + 1. And some factor it will it could be a positive negative something.

Which means you could take out the 2 power w and then say 1 + 2 times the factor which is exactly what I have written here. So, determinant of b is you can take out the 2 power w and then 1 + 2 b, that b is some b is the same thing as this factor and b will be an integer, it could be negative it could be positive.

(Refer Slide Time: 17:54)



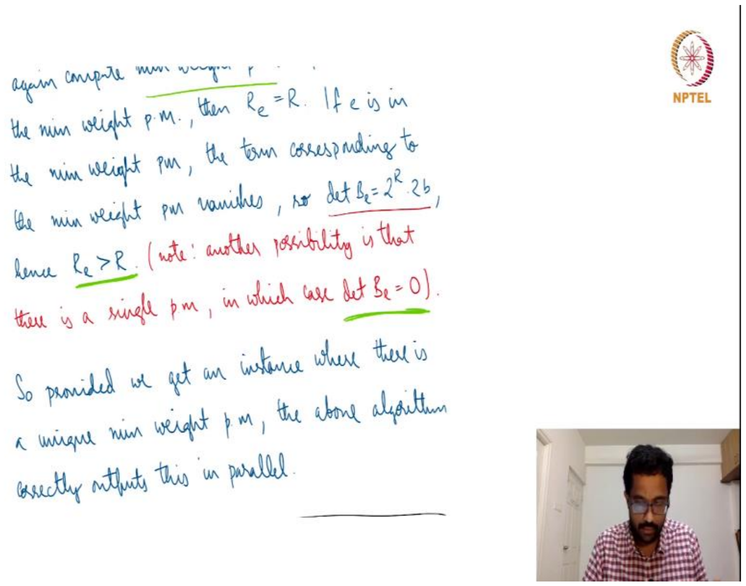
2^W divides det (B), [illegible]

M', $\underline{w(M') > W}$. So all terms are multiples of $\underline{2^{W+1}}$. Hence det $(B) = \underline{2^W(1+2b)}$ where b is an integer. So $\underline{2^W}$ is the largest power of 2 that divides $\underline{\det(B)}$. So $\underline{W=R}$ in step 3.

Now, we remove each edge e in parallel, and again compute min weight p.m. If e is not in the min weight p.m., then $R_e = R$. If e is in the min weight pm, the term corresponding to [illegible] vanishes. so det $B_e = 2^R \cdot 2b$,

This means so this 1 + 2 b is an odd number which means 2 power w is the largest power of 2 that divides determinant of B. And here if you look at the algorithm, we computed the largest R that such that 2 power R is a factor of determinant B. So, this B, this w that we that we computed or the R that we computed is actually the weight of the min weight perfect matching unique mean weight perfect matching. So, till now it is clean now let us see how the magic happens in the parallel steps.

(Refer Slide Time: 18:45)


again compute min weight p.m. If e is not in the min weight p.m., then $R_e = R$. If e is in the min weight pm, the term corresponding to the min weight pm vanishes, so $\det B_e = 2^R \cdot 2b$, hence $R_e > R$. (note: another possibility is that there is a single p.m, in which case $\det B_e = 0$).

So provided we get an instance where there is a unique min weight p.m, the above algorithm correctly outputs this in parallel.

So, the parallel step we are supposed to each edge does something on its own or the processor corresponding to each edge does something on its own. And automatically that is supposed to output that edge if and only if that edge is in the unique min weight perfect matching. So, now in the parallel operations we remove each edge and compute the minimum weight perfect matching. Suppose e is not in the min weight perfect matching.

So, for instance in this matrix here for instance this edge 2 1 is not in the perfect matching. So, if you remove 2 1, the old unique min weight perfect matching still persists. So, the w will be the unique min weight perfect matching weight and then still if you look extracted the largest factor or largest power w for which w 2 power w is a factor you will still get the same thing. So, if e is not in the mean weight; perfect matching.

So, again this looks like l so I will write e then R e will be equal to R, because the unique mean weight perfect matching is retained. If e is in the unique mean weight perfect matching, then that unique min weight perfect matching is now losing an edge. So, in this summation over here now this 2 power w will go because that one edge that is part of this; 2 power w is now being removed. So, this will now become two power w + 1 into the factor.

So, because of it because of this if e is in the unique mean weight perfect matching the term corresponding to this vanishes. So, the determinant of B e will be actually be 2 power R into 2 b.

So, which is what I have written here 2 power w + 1 or you have 2 power R into 2 b or maybe 2 power w into 2 b I have already said that R and w are the same. So, the R e will be at least R + 1. Another possibility is that if you remove e there is no edge there is no matching at all in which case determinant of B e is 0.

In both these cases when R e is greater than R or when determinant of B e is 0 the algorithm determinant of B e = 0 here, and R e is greater than R over in the step d in both the cases e is output. So, in both these cases we output e, and this is exactly the cases when e is part of the unique mean weight perfect matching, R e greater than R when there is a e is part of the unique mean weight perfect matching and there are other matching’s other perfect matching’s.

And determinant of B e is 0 when the edge e is part of all the perfect matching so if you remove e then the graph does not have a perfect matching. So, if we get an instance so what this is saying is that this correctly outputs the edge e if and only if it is part of the unique mean weight perfect matching. If it is not part of the unique mean weight perfect matching, it will not be output. Because it will that basically this 2 power w term will still remain.

Because it is, it may be part of no other matching or it may be part of some other matching, which is not the unique way perfect matching. So, that will not change the 2 power w quantity here. So, it will not change anything else. It will not be output so if we get an instance isolation lemma the randomness is such that we get an instance where there is a unique minimum weight perfect matching. In that case this algorithm correctly outputs this perfect matching in parallel.

I will just say this is all so this algorithm is correct and notice that from here on. This step four is entirely parallel; each processor is only looking at that corresponding to a particular edge e. It removes that edge basically it computes; it removes that meant that entry from the matrix you makes it makes it 0. And then computes the determinant. And then computes the largest power of 2 which divides the determinant.

And then compares with whatever was already computed whatever R was already computed. So, that is all that it has to do and then magically this leads to the correct answer. So, there is a probability of error but that probability of error is universal meaning. If the isolation lemma does not work if, we get with half probability we are supposed to get a unique min with perfect matching.

If you do not get that then it will not output the correct answer but if we get that it will output the correct answer and if you are not happy with the probability, we could we could increase the range instead of 2 times e we could we could choose let us say 10 times e or something. So, this will give us a better probability of success anyway. So, this shows that how we can compute or output the perfect matching in a bipartite graph if it has one in the randomized NC.

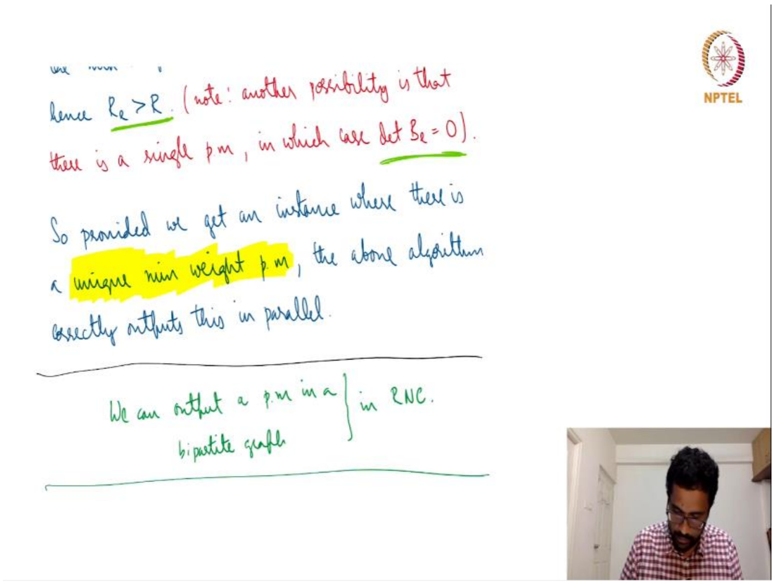
So, all that you do is you replace you get random weights for each of the edges. You replace the terms of the adjacency matrix with 2 power the random weights. And then you compute the determinant of this matrix, call it B. If the determinant is 0, then there is no perfect matching. If the determinant is non-zero you compute the largest R such that 2 power R divides determinant of B. Notice that if there is a unique mean weight perfect matching the determinant of B will necessarily be non-zero.

Because we already saw that if there is a unique min with perfect mention the determinant of B will be 2 power w into an odd number. So, it cannot be 0 this odd number is an odd number so it cannot, this means that if there is a unique minimum perfect matching it cannot cancel. And determinate of B cannot be 0 and then you compute the largest R and then you do it for each it is in parallel you remove the entry corresponding to that edge.

And then compute the determinant you call it B e determinant of B e then for each be you compute the R e the largest R e, R e for which or such that 2 power R e divides B e determinant of B e. And then you if either R is strictly greater than R or if determinant of B e is 0 then you output e. And this works whenever the isolation lemma has worked. If there is a unique min with perfect matching this algorithm correctly outputs in parallel each edge that is in the perfect matching.

So, each the processor that corresponds to each edge will either output the edge or not output the edge. And the point is that if there is unique min with perfect matching the correct edge will be the correct matching will be output by the individual processors.

(Refer Slide Time: 26:48)



hence $R_e > R$. (note: another possibility is that there is a single p.m, in which case $\det B_e = 0$).

So provided we get an instance where there is a unique min weight p.m, the above algorithm correctly outputs this in parallel.

$$\left.\begin{array}{c}\text{We can output a p.m in a}\\ \text{bipartite graph}\end{array}\right\}\text{ in RNC.}$$

And this shows that we can output a perfect matching in a bipartite graph in RNC. And with that we will conclude this lecture 46. So, in lecture 45 and 46 together we showed that the decision is not the decision the search version for of outputting a perfect matching is in a randomizing.