

Algorithms for Big Data
Prof. John Ebenezer Augustine
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Lecture - 43
Testing if a Graph is Bipartite

In this segment of our lecture on Random Walk space Algorithms for property testing we are going to look at an algorithm to test whether a given graph is either bipartite or far from being bipartite.

(Refer Slide Time: 00:30)

The slide is titled "LAZY RANDOM WALKS" in blue handwriting. It features a diagram of a vertex v with a self-loop and several outgoing edges. The self-loop is labeled with the probability $1 - \frac{\text{deg}(v)}{2d}$. The outgoing edges are labeled with $\frac{\text{deg}(v)}{2d}$ and $\frac{1}{2d}$. To the right of the diagram, there are handwritten notes in orange: "Consider: • d-regular • possibly bipartite graph. • Rapid mixing time". Below the diagram, there are handwritten notes in red: "Problem: Random walks are periodic Markov chains! How to make it aperiodic?". A small video inset in the top right corner shows a man with a beard and glasses, wearing a blue shirt, sitting in a chair.

We are going to be using random walks, but we need to be a bit careful here because we are dealing with bipartite graphs. Now let us assume that the graph is d -regular and let us possibly be a bipartite graph because one of the two options that we have in the promise given to us is either bipartite or far from being bipartite. And, we also have the promise that a random walk on this graph will mix rapidly, but we need to be a bit careful here because the traditional random walk is actually a periodic Markov chain in this context because the graph could be a bipartite graph.

Now, how do we ensure that it becomes aperiodic? And for that we are not going to use the traditional random walk rather we are going to use what is called the lazy random walk. So, let us consider a vertex v and it has degree d number of edges incident to it. If it is d -regular, the degree will be exactly d , but even otherwise you can think of d as

being an upper bound on the degrees and so the individual degrees can be something between 1 and d . So here is a vertex v with degree d_v number of edges incident added.

So, the question now is how we are going to assign the transition probabilities. What we are going to do is we are going to instead of transitioning with probability $1/d_v$ over the degree of the vertex v we are instead going to transition to neighbors with probability $1/d$ over $2d$ where d is the max degree. So every edge incident at v gets a probability of $1/2d$. But this leaves us with some remaining probability and with that probability $1 - d_v/2d$, so this is the probabilities that have been exhausted by considering all the edges incident that v .

So, one minus of that is the remaining probability and with that probability which is going to remain in the vertex v . You can see why such a random walk is called lazy because there is a probability of at least half that the walk will actually stay in the same vertex, it would not move here with that much probability. And the important thing to keep in mind is that no longer is the Markov chain periodic. You may want to pause for the little while to convince yourself of this a fact.

(Refer Slide Time: 03:44)

BIPARTITENESS TESTING

INPUT : A graph $G = (V, E)$ in bounded degree model.

Queries : What is the i^{th} neighbour of vertex v ?
What is the degree of vertex v ?

Promise : ① Either G is bipartite or $> \epsilon d n$ edges must be removed to make G bipartite. (where $\epsilon > 0$ fixed constant)

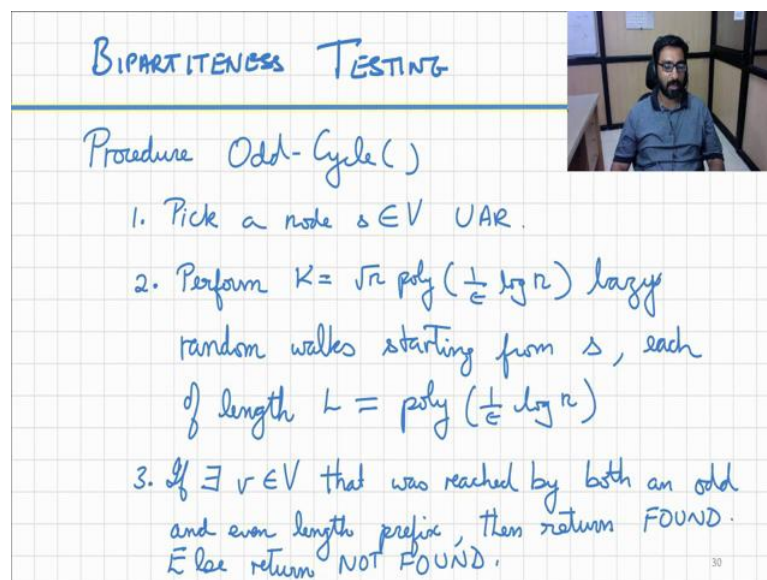
② Lazy random walks mix in $\tilde{O}(\frac{1}{\epsilon} \text{poly}(\log n))$ steps. (more on this later)

We are now ready to dive into the problem so let us make sure we understand the problem statement precisely. The input is a graph G equals V comma E in the bounded degree model. And in this model a queries are of the form what is the i th neighbor of vertex v . Also we may need to also find out what the degree of each vertex v is. So this is

the bounded degree model in property testing for graphs. We are given two promises; the first promise that the graph G is either bipartite or far from being a bipartite. What exactly do we mean by far from being bipartite? More than ϵn edges must be removed to make G bipartite. And here ϵ is some fixed constant greater than 0.

The second promise that we have is that the lazy random walks mix quickly in this graph. In particular the promises that lazy random walk as we described just while ago will mix within O of one over ϵ poly log of n number of steps which recall. What does it mean for a random walk to mix, it just means that the random walk will reach a probability distribution is very close to the stationary distributions within this amount of time.

(Refer Slide Time: 05:31)



BIPARTITENESS TESTING

Procedure Odd-Cycle()

1. Pick a node $s \in V$ UAR.
2. Perform $K = \frac{1}{\epsilon} \text{poly}(\frac{1}{\epsilon} \log n)$ lazy random walks starting from s , each of length $L = \text{poly}(\frac{1}{\epsilon} \log n)$.
3. If $\exists v \in V$ that was reached by both an odd and even length prefix, then return FOUND. Else return NOT FOUND.

So here is algorithm that we have for this problem and you may want to recollect that the graph is bipartite if and only if it has no odd-cycle in it. We are going to really take advantage of that and for that here is the key subroutine in this algorithm there the subroutine is called Odd-Cycle. So how does it go, we first pick a node in the graph and the node let us say chosen uniformly at random. Now we perform some number of lazy random walks. In particular we will be performing squared of n times polynomial in 1 over ϵ poly log n number of random walks starting from s .

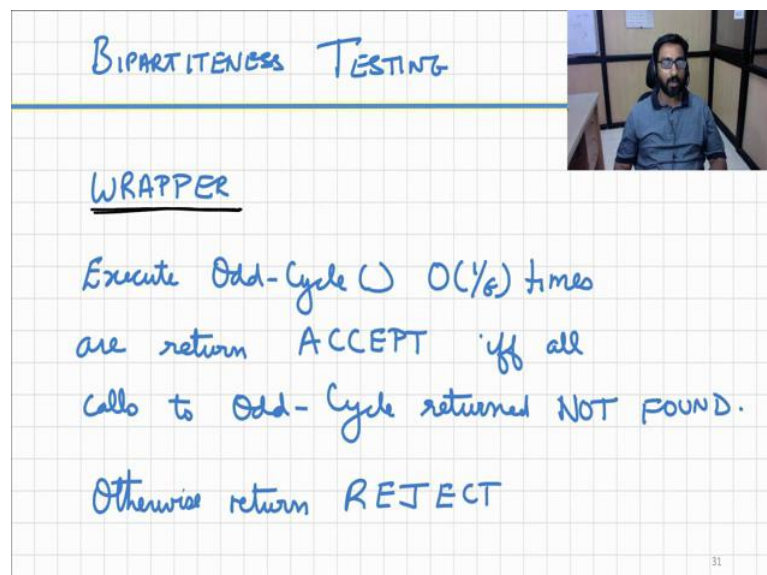
And each of these random walks will be of length L and that L is going to be polynomial n 1 over ϵ times $\log n$. And recall that this is going to be sufficient amount of time

for the random walk to mix. So, we are going to generate k such random walks, and there were observe something about these random walks. We are going to see if there exists some vertex v , such that this vertex was reached by both an odd and an even length random walk. Here something that we need to be a bit clear about. Now let us go back to the notion of lazy random walk, what we are counting as the length of the walk is the actual number of steps taken excluding these self loops that we have over here

So coming back to the algorithm here, the walk that are actually made L steps need not actually be of length L because there could be several self loops invoked. And what we are interested in is the length without the self loops being considered. There are two possibilities either that length can be odd or even. We want to know whether there was a vertex v that was reached by both an odd and an even length prefix. If such vertex v can be found then clearly what we can conclude is that there exists an odd-cycle.

So, we will return same we found an odd-cycle in which case eventually we will reject this graph. If we did not find such a vertex v then we will have to show subsequently that it is likely that this graph does not have an odd-cycle and therefore we will just return for now that there is no odd-cycle.

(Refer Slide Time: 08:58)



BIPARTITENESS TESTING

WRAPPER

Execute Odd-Cycle() $O(1/\epsilon)$ times
are return ACCEPT iff all
calls to Odd-Cycle returned NOT FOUND.
Otherwise return REJECT

31

Now, what we do this we just embed the odd-cycle subroutine within a rapper. All we do in this rapper is that we execute the odd-cycle subroutine some O of one over epsilon times and then we wait to see if there is any one odd-cycle call that comes back with

saying that it found an odd-cycle. If it found an odd-cycle then we will immediately reject. On the other hand, none of the calls to odd-cycle came up with the found return value. That means all of them came with the not found return value from the subroutine then we will accept this graph as being a bipartite graph.

(Refer Slide Time: 10:00)

ANALYSIS

- Will always accept bipartite graphs.
- Focus therefore on proving graph that are ϵ -far from being bipartite will be rejected.

↓
with prob $> \frac{2}{3}$

32

Of course, clearly this algorithms always going to correct the input graph is bipartite. So, moving into analysis all we need to do is focus on proving the graphs that are epsilon far from being bipartite would be rejected, and this we will want to reject with some constant probability let us say to two thirds.

(Refer Slide Time: 10:32)

The slide is titled "LAZY RANDOM WALK" and features a small video inset of a person in the top right corner. The main content is handwritten on a grid background. It includes a graph with two vertices labeled 's' and 'v'. A path is drawn between them, with a loop that is circled and labeled "not counted". Below the graph, it says $L = \text{poly}(\frac{n}{\epsilon})$. To the right, there are several definitions and assumptions:

- P_v = Prob that the r.w will terminate at v .
- Assumption: $\frac{1}{2n} \leq P_v \leq \frac{2}{n}$
- P_v^e = prob the r.w will terminate at v AND walk length is even. (with a note: \rightarrow Ignoring self loops)
- P_v^o = ... odd.
- A boxed equation: $P_v^e + P_v^o = P_v \geq \frac{1}{2n}$

Now, recall that the type of random walk we used is the lazy random walk and we pick the vertex s and we started making a random walk and eventually landed at some vertex v . And along the way we might have taken several self loops which we are simply not going to count in terms of the length of the walk from s to v . Now the length of the random walk, the number of steps taken by the random walk is some polynomial in small type of here polynomial $n \log n$ over ϵ .

So with that let us look at this random walk let us look at the characteristics little carefully. Let us denote by P_v , the probability that the random walk will terminate at v . Remember this we assumed that the random walk was of the lazy type. A quick exercise for you is to convince yourself that such a lazy random walk will actually have stationary distribution that is uniform over all the vertices. Now you may want to pause and convince yourself that is the case.

Now, having convinced yourself that the stationary distribution under the lazy random walk is actually the uniform distribution, and fact coupling that with the fact that our random walks have a walked for mixing time number of steps, we can assume that P_v the probability that the random walk will be at v is close to $\frac{1}{n}$. In particular let us say that it is within half of $\frac{1}{n}$ to at most twice of $\frac{1}{n}$. Now, this P_v we are going to break their up into two parts. Remember we want to eventually argue something about walks that took an even number I mean walks of even length versus odd length. So, what

we are going to do is we are going to break up $P v$ into two parts.

$P v$ with a superscript E refers to the probability that the random walk will terminate at v and the walk length, and this is the another random walk number of steps by the random walk taken by the random walk, but the walking length which is obtained by ignoring the self loops is even. So, this is the probability that you started at s and you may the L length random walk including some self loops and then you reset v , and if you remove all the self loops and just look at the length of the walk from s to v this walk was of even length and you landed in v . This is the probability of that event.

Similarly, you can define $P v$ superscript O for odd which is exactly the same event except that the length of the walk now odd. Clearly $P v$ superscript E plus $P v$ superscript O both are up to $P v$ and we know the $P v$ is at least 1 over $2 n$. Now with this notion of lazy random walks and the notation that go along with it, we are now ready to look at the framework of our analysis.

(Refer Slide Time: 14:28)

ANALYSIS

Consider $\sum_{v \in V} P_r^e \cdot P_r^o$

Bipartite \rightarrow "0"

ϵ -far from being bipartite \rightarrow ①

② "Large"

P_r [odd cycle] returns found \geq a constant \leftarrow

$\geq \sum_{c \in C} \rightarrow$ some const.

So, how are we going to do this analysis? There is going to be a very, very key summation. This is the summation over all the vertices of the product $P v e$ times $P v o$. Now think about this to understand what the summation is about, if the graph is a bipartite graph for each vertex v either $P v e$ is 0 or $P v o$ is 0 because starting at s each other vertex can only be reached by either a even length path or an odd length path. So, one of these probabilities is going to be 0 if the graph is bipartite. So the entire

summation is going to be 0.

However, this summation let us consider what happens when the graph is epsilon far from being bipartite. Now we need to show that this summation is actually is going to be large and by large we mean that it is greater than some epsilon over c prime n. And more over if it is large we need to be able to say then that this function odd-cycle will return a found verdict with some constant probability. So let me repeat that again, when the given graph is far from being bipartite then this summation starts to take a larger value it is certainly not 0 so it start to take a much larger value and in a particular we want to show that this summation take some value is greater than epsilon over c prime n and c prime is some constant.

Given that the summation is so large we need to be able to show that the odd-cycle subroutine will return found verdict with some significant probability and this is the two steps of the analysis that we are going to follow. So without further you do let us start with step 1.

(Refer Slide Time: 17:07)

ANALYSIS: STEP 1

ϵ -far from being bipartite $\rightarrow \sum_v P_v^e \cdot P_v^o \geq \frac{\epsilon}{cn}$

We will show that

$\sum_v P_v^e \cdot P_v^o < \frac{\epsilon}{cn} \wedge \epsilon$ -far from being bipartite
leads to a contradiction!

And recollect what is it that we want to show if the graph is epsilon far from being bipartite then this summation over all these words is P_v^e times P_v^o is at least epsilon over cn , where c is some constant. But, well this is a standard proof technique we will show that if we start with the assumptions that the summation is actually small as in less than epsilon over cn and the graph is far from being bipartite we can work our way to

the contradiction. If you think about this when this leads to a contradiction what we are proved is the negation of this statement that we have over here and the negation of the statement is the negation of this individual statement or this statement. And negation if you work it out it is exactly going to be logically equivalent to this implication.

So, let us move on and just the things that you need to keep in mind is that we are assuming that the summation is small, there is less than epsilon over c n and the graph is epsilon far from being a bipartite, so keep that in mind.

(Refer Slide Time: 18:53)

ANALYSIS: STEP 1

$P_v^e \cdot P_v^o = \frac{1}{4n} \cdot \frac{d_e(v)}{64dn}$

Vertex chosen in step 1 "enforce"

neighbors of v in V_e

Similar expression can also be obtained for $v \in V_o$

$P_v^e \geq \frac{1}{4n}$

$P_u^e \geq \frac{1}{4n}$

$P_u^e(L-1) ?$

Claim: $P_u^e(L-1) \geq \frac{1}{32n}$

Exercise

$V_e = \{v \mid P_v^e \geq P_v^o\}$
 \downarrow
 even

$V_o = V \setminus V_e$
 \downarrow
 odd

$P_v^o \geq \sum_u \frac{1}{2d} P_u^e(L-1)$
 neighbors of v in V_e
 $\geq d_e(v) \cdot \frac{1}{64dn}$

So with that let us start with the analysis source. Of course, in the odd-cycle subroutine we first pick an s and you can think of it has a sort of an enforce step, because the moment you pick s you can then set into two partition of a vertex set into two parts V_e and V_o . This is not just done arbitrarily, but rather this is done in the following manner. So, this is set V_e is the set of all vertices such that the probability P_v^e is greater than or equal to P_v^o . These are the vertices such that you are more likely to terminate at them up within even number of steps.

Remember if it is a bipartite graph every step will take you from one part to the other part. So that makes a lot sense because now you are more likely and even steps to be on the left part and that is at least what we hope if this is the proper way to partition this graph, so this is V_e . And anything other than V_e replace in V_o . These are going to be vertices there are more likely for the random walk to terminate having a walk length that

is of odd length. Or let us now pick a vertex v . This is some vertex v for convenience we place it in V_e , but later we can see that it could be even and V_o .

What do we know about v ? Well, $P_v^{(e)}$ is at least $\frac{1}{2n}$. Why? Well, without the e superscript that itself is going to be $\frac{1}{2n}$. And moreover we know that P_v is equal to P_v^e plus P_v^o and we know that on the left side P_v^e dominates and so if you think about it P_v^e must be at least one fourth of $\frac{1}{n}$ ok Now, consider the neighbourhood of v . Just to let you know where we are going, we are trying to find the product of P_v^e times P_v^o . Now well P_v^e we already have, but to get to P_v^o we are going to try and get an a bound P_v^o by looking at the neighbourhood of v and seeing what is the probability of reaching the neighbourhood even number of steps, because once you reach in neighbourhood within even number of steps then in one step in you will reach v and then that will be a way to reach v with a walk that is of odd length. And so that will give us P_v^o . So that is really where we are going.

So let us go one step at a time. Let us pick a vertex u that is the neighbor of v and let us focus in particular on the neighbourhood v within the set V_e here. These are the neighbors. So, v might have neighbor over here but we are not going to worry about that. And we know that for this under the same reasoning we know P_u^e is at least $\frac{1}{4n}$. And here something we need to prove. This is going to be left as an exercise, but it is nice (Refer Time: 23:02) exercise for you to you have to think about.

What we need to show is the probability that a random walk starting at s will reach u within even number of a walk length that is of even parity. In $L-1$ random walk steps. Here we are adding little primary to this probability, here we are specifying that the number of random walk steps should be $L-1$. This probability we want to understand this. And our claim is that probability is not too be small it is at least $\frac{1}{32n}$. So think about this.

The hint that I would like to give is that basically we know this P_u superscript is at least $\frac{1}{2n}$. Now you will reach u having taken several self loops. So the probability of reaching having the same walk structure, but achieving and that in $L-1$ random walk steps is you can achieve that by just skipping one of the self loops. So, exploiting that we want to get a bound on this probability and we want to be able to say that that is at least $\frac{1}{32n}$. So, this is your exercise.

And, for now let us proceed with this bound. Now notice that with this bound what that means is we have reached u with some probability $\frac{1}{32^n}$ and we have reached u with walk length that is of even length up to u , which means that with $\frac{1}{2^d}$ probability we will step into v and by doing so we would have reached v with the walk length that is odd. So that gives us therefore a bound for this probability $P_{v \rightarrow o}$. So, $P_{v \rightarrow o}$ now we look at one particular u , but now we have to sum over all neighbors of all u they are the neighbors v , but in restricted to this $v \in e$. Summation of over all such u , this is the probability that we already have over here that we will reach u within even length walk in $L - 1$ random walk steps. And then with probability $\frac{1}{2^d}$ we will transition in to v and summed over u we will get a bound for lower bound for $P_{v \rightarrow o}$.

And now remember we have a notation d_e of v for the number of neighbors of v that are in this set $v \in e$. Therefore, this is simply d_e of v times and now we are going to plug-in this inequality of $\frac{1}{32^n}$ over here this 2 here will make this as 64 so we get this bound over here. Notice now that we have a bound for $P_{v \in e}$ and we have bound for $P_{v \rightarrow o}$. These are two components that we need to multiply and we are interested in $P_{v \in e}$ times $P_{v \rightarrow o}$. So all we need to do is, well this is nothing but $\frac{1}{4^n}$ times this is that basically $P_{v \in e}$ times this bound over here d_e over 64^n . Just to be clear this must not be an equal to but rather in inequality

One last comment here, this exact same argument can be repeated for vertices see on the right side as well. So, the same expression except now we would not have d_e , but will have d_o because it is now if the vertex v was chosen on the right side then we will be interested in neighbors that are on the right side, so that will be d_o of v . (Refer Time: 28:06) that change this expression will remain exactly the same. We can actually put this expression back into the larger summation.

(Refer Slide Time: 28:16)

ANALYSIS : STEP 1

$$\sum_{v \in V} P_v^e \cdot P_v^o \geq \sum_{v \in V_e} \frac{1}{4n} \frac{d_e(v)}{64dn} + \sum_{v \in V_o} \frac{1}{4n} \frac{d_o(v)}{64dn}$$

$= \frac{1}{c'd n^2} \left[\sum_{v \in V_e} d_e(v) + \sum_{v \in V_o} d_o(v) \right]$

$\geq \frac{\epsilon}{c'n}$

of violating edges for partition $(V_e, V_o) > \epsilon dn$

Remember we were interested in the summation over all the vertices $P_{v \in e} \times P_{v \in o}$, and this summation is at least the sum of these individual summation. The first summation is simply based on what we have over here, and the second summation is what we would have if we considered the vertex v to be drawn from here. So that is essentially what we have over here, v drawn from $v \in e$ and $v \in e$ and $v \in e$ and $v \in o$, but the expression inside the summation is pretty much identical except that we are using $d_o(v)$ here (Refer Time: 29:02) this number of neighbors of v and $v \in o$ given that v is in $v \in o$.

So, now pulling out the common terms we get 1 over some constant $c' d n^2$ the n square outside and then we have the summation over vertices in $v \in e$, summation over vertices in $v \in o$, the neighbors of $d_{v \in e}$ and the neighbors of $v \in d_{v \in e}$ and in the set $v \in e$ and here in the set $v \in o$. What exactly is the summation? Well, this is as the turns out simply number of violating edges for the partition $v \in e$ and $v \in o$. Why is that? Well, think about this is if that partition considers the partition $v \in e$ and $v \in o$ these are the neighbors within the same partition. So, these are the violating edges and this number of violating edges must be more than ϵdn that is the assumption that we started off with.

So, if we plug this is ϵdn over here we will end up getting a bound of ϵ over $c' n$. This dn will cancel out with this dn and one of these n 's so we will be left with this summation being at least ϵ over some $c' n$. But keep in mind our assumption was that this joint summation is actually less than $\epsilon c n$. And this

is our assumption that we started off with. Now if c prime here is small then this quantity is going to be larger compared to this ϵ / cn and that would lead to a contradiction because on the one hand our assumption is that this summation is less than the smaller quantity. On the other hand what we are worked out is that the summation is greater than this larger quantity. So, this is the contradiction that we have an established.

With this contradiction that the first step in our analysis is over and we need to move on to the second step. Just remind ourselves conclusions on first step is that, if the graph is ϵ far from being bipartite then this summation will be more than some ϵ / cn . And given that it is large now our second steps to prove that the function odd-cycle will actually return saying that an odd-cycle is found with the probability at least some constant. And vice constant that means we are say that it is bounded away from 0.

(Refer Slide Time: 32:14)

The slide contains the following content:

ANALYSIS: STEP 2

Given: $\sum_{v \in V} P_r^e \cdot P_r^o \geq \frac{\epsilon}{cn}$

$\eta_{ij} = \begin{cases} 1 & \text{if } i^{\text{th}} \text{ and } j^{\text{th}} \text{ walk from } s \text{ terminated at } v \text{ AND one was even length while other was odd.} \\ 0 & \text{otherwise} \end{cases}$

$\Pr(\eta_{ij} = 1) = \sum_{v \in V} 2 P_r^e \cdot P_r^o = E[\eta_{ij}]$
↳ Two odd/even combinations

So now let us move into a step 2 of the analysis. Just to remind ourselves, we are given that the this submission overall $\sum_v P_v^e \times P_v^o$ is greater than or equal to ϵ / cn , and we need to show that the odd-cycle will find the bipartite and find with good probability find an odd-cycle therefore our safe return that is found 1. Now, let us define an indicator random variable. This is indicator random variable η_{ij} will be set to 1 if the i th and the j th random walk from s terminated at v . And one of them was of even walk client while the other was odd.

So, clearly you can see why this indicated a random variable is very important, because

if η_{ij} is 0 that means we have found a candidate vertex v which gives us a witness to the fact that there is an odd-cycle and therefore we will be able to reject this graph. So, η_{ij} takes on the value 1 under that condition and otherwise it takes on the value 0. So, what is the probability that η_{ij} equal to 1 that is nothing but, well for η_{ij} to be 1 there must be some vertex v such that 1 of the probabilities let us pick a vertex v with the probability P_v the random walk will reach v having an walk length that is even, with P_v odd it will with this probability it will reach vertex v with a walk length that is odd.

Let us say i th random walk follow this probability and the j th random walk follow this probability what you end up having is an evidence of an odd-cycle. The other option is that the j th random walk follows this probability and i th random walk follows this probability. So that is why we have this two, because there is two ways which these can combine. And that is for one vertex v . And we have to therefore sum over all possible destination vertices v . And this summation is gives us the probability that η_{ij} equals 1. And this is nothing but the expectation on the value of η_{ij} as well. That is simply be follows from the fact that η_{ij} is an indicator random variable.

(Refer Slide Time: 35:52)

ANALYSIS: STEP 2

Notice that the algorithm will fail to reject if $\sum_{i < j} \eta_{ij} = 0$ → We need an upper bound on prob of this event

Exercise. Hint Use Chebyshev's

$E \sum_{i < j} \eta_{ij} = \sum_{i < j} E[\eta_{ij}] > 1$

none of the pairs of walks terminate at the same vertex with different parity lengths.

Total # of walks $K = \text{poly}(\frac{1}{\epsilon} \log n) \cdot \sqrt{n}$

⇒ # of pairs is $\Omega(K^2) = \Omega(n/\epsilon)$

And now notice that at the end, let us count all the η_{ij} 's all the possible η_{ij} 's. If the summation over all η_{ij} 's where i is strict (Refer Time: 36:09) j . If this summation is equal to 0, what that means is none of the pairs of walks terminated at the same vertex with different parity lines. So we not been able to find any evidence that the graph is not

bipartite, which means that we will fail to reject and number where of course this whole analysis is under the assumption that the graph is actually far being from bipartite so this would be the bad event. If this event were to occur then our algorithm is incorrect. And so we need to ensure the probability of this event is small.

Let us consider the probability of the complementary event. The total number of walks member is some polynomial in $1/\epsilon$ times $\log n$ the whole times are squared of n . I mean the most important term to keep in mind is this square root of n over here. This square root of n random walks means that the total number of pairs is at least k^2 , big omega of k^2 , which means that the number of pairs if you are considering that the k square root of n is big omega of n/ϵ . So there is also this also this (Refer Time: 37:47) ϵ over here.

With that let us and consider the expectation of the sum of η_{ij} 's. That is of course by linearity of expectation nothing but the summation over all $i < j$ expectation of η_{ij} . Of course, the expectation of η_{ij} if you recall is this quantity $2 P_{ve} \times P_{vo}$ summed over all vertices, and remember then this is now we are assuming that this is large. So at least $\sum \epsilon / c n$. With that assumption this expectation is going to be ϵ of the order of $\epsilon / c n$, but then we are summing over all pairs and that is going to be big omega of n / ϵ .

So, some of the term we n in the ϵ will cancel out and this expectation will be some θ of one term. So it is going to be greater than 1 assuming the constants are bipartitely chosen. But this does not give us enough strength, because notice that this only says expectation is greater than 1. What we want is to ensure that the probability with which it will actually be this e this summation η_{ij} will be more than 1 basically the event the complementary to this event is it should large. So, the part of the left as the exercise now is to show using Chebyshev's inequality that probability of this event is actually is small.

So, over which implies that the probability of the summation of η_{ij} is exceeding 1 is greater than or equal to 1 is sufficiently large. And remember that if that happens algorithm will in fact reject the graph. So, with that we really come to end of the second step in the analysis and therefore also the end of the overall analysis. If all of this workout then the summation η_{ij} 's will be greater than or equal to 1 with some good

probability which means that we have detected a collation of (Refer Time: 40:37).

So, that brings us to the end of our analysis of the property testing algorithm of a bipartite testing that used random walks.