PROFESSOR: So we spent the last few lectures discussing Szemerédi's regularity lemma. So we saw that this is an important tool with important applications, allowing you to do things like a proof of Roth's theorem via graph theory.

One of the concepts that came up when we were discussing the statement of Szemerédi's regularity lemma is that of pseudorandomness. So the statement of Szemerédi's graph regularity lemma is that you can partition an arbitrary graph into a bounded number of pieces so that the graph looks random-like, as we called it, between most pairs of parts.

So what does random-like mean? So that's something that I want to discuss for the next couple of lectures. And this is the idea of pseudorandomness, which is a concept that is really prevalent in combinatorics, in theoretical computer science, and in many different areas. And what pseudorandomness tries to capture is, in what ways can a non-random object look random?

So before diving into some specific mathematics, I want to offer some philosophical remarks. So you might know that, on a computer, you want to generate a random number. Well, you type in a "rand," and it gives you a random number.

But of course, that's not necessarily true randomness. It came from some pseudorandom generator. Probably there's some seed and some complex-looking function and outputs something that you couldn't distinguish from random. But it might not actually be random but just something that looks, in many different ways, like random.

So there is this concept of random. You can think about a random graph, right, generate this Erdos-Renyi random graph. Every edge occurs independently with some probability.

But I can also show you some graph, some specific graph, which I say, well, it's, for all intents and purposes, just as good as a random graph. So in what ways can we capture that concept? So that's what I want to discuss. And that's the topic of pseudorandomness.

And of course, well, this idea extends to many areas, number theory and whatnot, but we'll stick with graph theory. In particular, I want to explore today just one specific notion of pseudorandomness. And this comes from an important paper called "Quasi-random graphs."

And this concept is due to Chung, Graham, and Wilson back in the late '80s. So they defined various notions of pseudorandomness, and I want to state them. And what it turns out-- and the surprising part is that these notions, these definitions, although they look superficially different, they are actually all equivalent to each other.

So let's see what the theorem says. So the set-up of this theorem is that you have some fixed real p between 0 and 1. And this is going to be your graph edge density.

So for any sequence of graphs, Gn-- so from now, I'm going to drop the subscript n, so G will just be Gn-- such that the number of vertices-- so G is n vertex with edge density basically p. So this is your sequence of graphs.

And the claim is that we're going to state some set of properties. And these properties are all going to be equivalent to each other. So all of these properties capture some notion of pseudorandomness, so in what ways this is graph G or really a sequence of graphs. Or you can talk about a specific graph and have some error parameters and error balance. They're all roughly the same ideas.

So in what ways can we talk about this graph G being random-like? Well, we already saw one notion when we discussed Szemerédi's regularity lemma. And let's see that here. So this notion is known as discrepancy. And it says that if I restrict my graph to looking only at edges between some pair of vertex sets, then the number of edges should be roughly what you would expect based on density alone.

So this is basically the notion that came up in epsilon regularity. This is essentially the same as saying that G is epsilon regular with itself where this epsilon now is hidden in this little o parameter.

So that's one notion of pseudorandomness. So here's another notion which is very similar. So it's almost just a semantic difference, but, OK, so I have to do a little bit of work.

So let me call this DISC prime. So it says that if you look at only edges within this set-- so instead of taking two sets, I only look at one set-- and then look at how many edges are in there versus how many you should expect based on density alone, these two numbers are also very similar to each other.

So let's get to something that looks dramatically different. The next one, I'm going to call count. So count says that for every graph H, the number of labeled copies of H in G-- OK, so labeled copies, I mean that the vertices of H are labeled. So for every triangle, there are six labeled triangles that correspond to that triangle in the graph.

The number of labeled copies of H is-- so what should you expect if this graph were truly random? You would expect p raised to the number of edges of H plus small error times n raised to number of vertices of H. And just as a remark, this little o term, little o 1 term, may depend on H.

So this condition, count, says for every graph H, this is true. And by that, I mean for every H, there is some sequence of decaying errors. But that sequence of decaying errors may depend on your graph H. OK.

The next one is almost a special case of count. It's called C4. And it says that the number of labeled copies of C4, so the fourth cycle, is at most p raised to power of 4-- so again, what you should expect in a random setting just for cycle count alone.

I see, already, some of you are surprised. So we'll discuss that this is an important constraint. It turns out that alone implies everything, just having the correct C4 count.

The next one, we will call codegree. And the codegree condition says that if you look at a pair of vertices and look at their number of common neighbors-- in other words, their codegree-- then what should you expect this quantity to be? So there are n vertices that possibly could be common neighbors, and each one of them, if this were a random graph with edge probability p, then you expect the number of common neighbors to be around p squared n.

So the codegree condition is that this sum is small. So most pairs of vertices have roughly the correct number of common neighbors. So codegree is number of common neighbors.

Next, and the last one, certainly not the least, is eigenvalue condition. So here, we are going to denote by lambda 1 through lambda G the eigenvalues of the adjacency matrix of G. So we saw this object in the last lecture. So I include multiplicities. If some eigenvalue occurs with multiple times, I include it multiple times.

So the eigenvalue condition says that the top eigenvalue is around pn and that, more importantly, the other eigenvalues are all quite small. Now, for d regular graph, the top eigenvalue-- and it's fine to think about d regular graphs if you want to get some intuition out of this theorem. For d regular graph, the top eigenvalue is equal to d, because the top eigenvector is d. It's the all-one vector.

So top eigenvector is all-one vector, which has eigenvalue d. And what the eigenvalue condition says is that all the other eigenvalues are much smaller. So here, I'm thinking of d as on the same order as n.

OK, so this is the theorem. So that's what we'll do today. We'll prove that all of these properties are equivalent to each other. And all of these properties, you should think of as characterizations of pseudorandomness.

And of course, this theorem guarantees us that it doesn't matter which one you use. They're all equivalent to each other. And our proofs are actually going to be-- I mean, I'm going to try to do everything fairly slowly. But none of these proofs are difficult. We're not going to use any fancy tools like Szemerédi's regularity lemma.

In particular, all of these quantitative errors are reasonably dependent on each other. So I've stated this theorem so far in this form where there is a little 1 error. But equivalently, so I can equivalently state theorem as-- for example, have DISC with an epsilon error, which is that some inequality is true with at most epsilon error instead of little o.

And you have a different epsilon for each one of them. And the theorem, it turns out that-- OK, so the proof of this theorem will be that these conditions are true, so all equivalent, up to at most a polynomial change in the epsilons.

In other words, so property one is true for epsilon implies that property two is true for some epsilon raised to a constant. So the changes in parameters are quite reasonable. And we'll see this from the proof, but I won't say it again explicitly. Any questions so far about the statement of this theorem?

So as I mentioned just now, the most surprising part of this theorem and the one that I want you to pay the most attention to is the C4 condition. This seems, at least at face value, the weakest condition among all of them. It just says the correct C4 count.

But it turns out to be equivalent to everything else. And there's something special about C4, right? If I replace C4 by C3, by just triangles, then it is not true. So I want you to think about, where does C4 play this important role? How does it play this important role?

OK. So let's get started with a proof. But before that, let me-- so in this proof, one recurring theme is that we're going to be using the Cauchy-Schwarz inequality many times. And I want to just begin with an exercise that gives you some familiarity with applying the Cauchy-Schwarz inequality.

And this is a simple tool, but it's extremely powerful. And it's worthwhile to master how to use a Cauchy-Schwarz inequality. So let's get some practice. And let me prove a claim which is not directly related to the proof of the theorem, but it's indirect in that it explains somewhat the C4 condition and why we have less than or equal to over there.

So the lemma is that if you have a graph on n vertices such that the number of edges is at least pn squared over 2, so edge density basically p, then the number of labeled copies of C4 is at least p to the 4 minus little o 1 n to the 4th. So if you have a graph with each density p-- p's your constant-- then the number of C4s is at least roughly what you would expect in a random graph.

So let's see how to do this. And I want to show this inequality as a-- well, I'll show you how to prove this inequality. But I also want to draw a sequence of pictures, at least, to explain how I think about applications of the Cauchy-Schwarz inequality.

OK. So the first thing is that we are counting labeled copies of C4. And this is basically but not exactly the same as number of homomorphic copies of C4 and G. So by this guy here, I really just mean you are mapping vertices of C4 to G so that the edges all map to edges.

But we are allowing not necessarily injective maps, C4 to G. But that's OK. So the number of non-injective maps is at most cubic. So we're not really affecting our count. So it's enough to think about homomorphic copies.

OK. So what's going on here? So let me draw a sequence of pictures illustrating this calculation. So first, we are thinking about counting C4s. So that's a C4. I can rewrite the C4 count as a sum over pairs of vertices of G as the squared codegree.

And what happens here-- so this is true. I mean, it's not hard to see why this is true. But I want to draw this in pictures, because when you have larger and bigger graphs, it may be more difficult to think about the algebra unless you have some visualization.

So what happens here is that I notice that the C4 has a certain reflection. Namely, it has a reflection along this horizontal line. And so if I put these two vertices as u and v, then this reflection tells you that you can write this number of homomorphic copies as the sum of squares.

But once you have this reflection-- and reflections are super useful, because they allow us to get something into a square and then, right after, apply the Cauchy-Schwarz inequality. So we apply Cauchy-Schwarz here. And we obtain that this sum is at most where I can pull the square out.

And I need to think about what is the correct factor to put out here. And that should be-- so what's the correct factor that I should put out there?

AUDIENCE: 1 over n squared.

PROFESSOR: OK, so 1 over n squared. So I don't actually like doing these kind of calculations with sums, because then you have to keep track of these normalizing factors. One of the upcoming chapters, when we discuss graph limits-- or in fact, you can even do this.

Instead of taking sums, if you take an average, if you take an expectation, then it turns out you never have to worry about these normalizing factors. So normalizing factors should never bother you if you do it correctly. But just to make sure things are correct, please keep me in check.

All right. So what happened in this step? In this step, we pulled out that square. And pictorially, what happens is that we got rid of half of this picture. So we used Cauchy-Schwarz, and we wiped out half of the picture.

And now what we can do is, well, we're counting these guys, this path of length 2. But I can reprioritize this picture so that it looks like that. And now I notice that there is one more reflection.

So there's one more reflection. And that's the reflection around the vertical axis. So let me call this top vertex x. And I can rewrite the sum like that.

OK. So once more, we do Cauchy-Schwarz, which allows us to get rid of half of the picture. And now I'm going to draw the picture first, because then you see that what we should be left with is just a single edge. And then you write down the correct sum, making sure that all the parentheses and normalizations are correct. But somehow, that doesn't worry me so much, because I know this will definitely work out.

But whatever it is, you're just summing the number of edges. So that's just the number of edges. And so we put everything in. And we find that the final quantity is at least p raised to 4 n to 4.

So I did this quite slowly. But I'm also emphasizing the sequence of pictures, partly to tell how I think about these inequalities. Because for other similar looking inequalities-- in fact, there is something called Sidorenko's conjecture, which I may discuss more in a future lecture, that says that this kind of inequality should be true whenever you replace C4 by any bipartite graph. And that's a major open problem in combinatorics.

It's kind of hard to keep track of these calculations unless you have a visual anchor. And this is my visual anchor, which I'm trying to explain. Of course, it's down to earth. It's just the sequence of inequalities. And this is also some practice with Cauchy-Schwarz. All right. Any questions?

But one thing that this calculation told us is that if you have edge density p, then you necessarily have C4 density at least p to the 4th. So that partly explains why you have at most, then, here. So you always know that it's at least this quantity. So the C4 quasi randomness condition is really the equivalent to replacing this less than or equal to by an equal sign.

So let's get started with proving the Chung-Graham-Wilson theorem. So the first place that we'll look at is the two versions of DISC. So DISC stands for discrepancy.

So first, the fact that DISC implies DISC prime, I mean, this is pretty easy. You take y to equal to x. Be slightly careful about the definitions, but you're OK. So not much to do there.

The other direction, where you only have discrepancies for a single set and you want to produce discrepancies for pairs of sets-- so this is actually a fairly common technique in algebra that allows you to go from bilinear forms to quadratic forms and vice versa. It's that kind of calculation.

So let me do it here concretely in this setting. So here, what you should think of is that you have two sets, x and y, and they might overlap. And what they correspond to in the-- when you think about the corresponding Venn diagram, where I'm looking at ways that a pair of vertices can fall in x and/or y-- so if you have x and y.

And so it's useful to keep track of which vertices are in which set. But what the thing finally comes down to is that the number of edges with one vertex in x and one vertex in y, I can write this bilinear form-type quantity as an appropriate sum of just number of edges in single sets.

And so there are several ways to check that this is true. One way is to just tally, keep track of how many edges are you counting in each step. So if you are trying to count the number of edges in-- yeah, so let's say if you're trying to count the number of edges in-- with one vertex in x, one vertex in y. Then what this corresponds to is that count.

But let me do a reflection. And then you see that you can write this sum as an alternating sum of principal squares, so this one big square plus the middle square and minus the two sides squares, which is what that sum comes to.

All right. So if we assume DISC prime, then I know that all of these individual sets have roughly the correct number of edges up to a little o of n squared error. And again, I don't have to do this calculation again, because it's the same calculation. So the final thing should be p times the sizes of x and y together plus this same error.

So that shows you DISC prime implies DISC. So the self version of discrepancy implies the pair version of discrepancy. So let's move on to count. To show that DISC implies count-- actually, we already did this.

So this is the counting lemma. So the counting lemma tells us how to count labeled copies if you have these epsilon regularity conditions, which is exactly what DISC is. So count is good.

Another easy implication is count implies C4. Well, this is actually just tautological. C4 condition is a special case of the count hypothesis.

All right. So let's move on to some additional implications that require a bit more work. So what about C4 implies codegree? So this is where we need to do this kind of Cauchy-Schwarz exercise.

So let's start with C4. So assume a C4 condition. And suppose you have this-- so I want to deduce that the codegree condition is true. But first, let's think about just what is the sum of these codegrees as I vary u and v over all pairs of vertices.

So this is that picture. So that is equal to the sums of degrees squared, which now, by Cauchy-Schwarz, you can deduce to be at least n times 2 raised to number of edges-- namely, the sum of the degrees-- that thing squared.

So now we assume the C4 condition-- actually, no, we assume that G has the density as written up there. So this quantity is p squared plus little 1 times n cubed, which is what you should expect in a random graph of Gnp.

But that's not quite what we're looking for. So this is just the sum of the codegrees. What we actually want is the deviation of codegrees from its expectations, so to speak.

Now, here's an important technique from probabilistic combinatorics is that if you want to control the deviation of a random variable, one thing you should look at is the variance. So if you can control the variance, then you can control the deviation.

And this is a method known as a second moment method. And that's what we're going to do here. So what we'll try to show is that the second moment of these codegrees-- namely, the sum of their squares-- is also what you should expect as if the random setting. And then you can put them together to show what you want.

So this quantity here, well, what is this? We just saw-- see, up there, it's also codegree squared. So this quantity is also the number of labeled copies of C4-- not quite, because you might have two vertices and the same vertex. So I incorporate a small error. So it's a cubic error, but it's certainly sub n to the 4.

And we assume that the number of labeled copies of C4 by the C4 condition is no more than basically p to the 4 times n raised to power 4. OK. So now you have a first moment. You have some average, and you have some control.

In the second moment, I can put them together to bound the deviation using this idea of controlling variance. So the codegree deviation is upper bounded by-- so here, using Cauchy-Schwarz, it's upper bounded by basically the same sum, except I want to square the summand.

This also gets rid of the pesky absolute value side, which is not nicely, algebraically behaved. OK. So now I have the square, and I can expand the square. So I expand the square into these terms. And the final term here is p to the 4 n to the 6. No, n to the 4.

All right. But I have controlled the individual terms from the calculations above. So I can upper bound this expression by what I'm writing down now.

And basically, you should expect that everything should cancel out, because they do cancel out in the random case. Of course, the sanity check, it's important to write down this calculation.

So if everything works out right, everything should cancel out. And indeed, they do cancel out. And you get that-- so this is a multiplication. This is p squared. Is that OK? So everything should cancel out. And you get a little o of n cubed.

To summarize, in this implication from C4 to codegree, what we're doing is we're controlling the variance of codegrees using the C4 condition and the second moment bound, showing that the C4 condition trumps over the codegree condition. Any questions so far?

So I'll let you ponder in this calculation. The next one that we'll do is codegree implies DISC. And that will be a calculation in a very similar flavor. But it will be a slightly longer but with similar flavor of calculation. So let me do that after the break.

All right. So what have we done so far? So let's summarize the chain of implications that we have already proved. So first, we started with showing that the two versions of DISC are equivalent. And then we also noticed that DISC implies count through the counting lemma. So we also observed that count implies C4 tautologically and C4 implies codegree.

So the next natural thing to do is to complete this circuit and show that the codegree condition implies the discrepancy condition. So that's what we'll do next.

And in some sense, these two steps, you should think of them as going in this natural chain, where C4-- so C4 is like this, C4. Codegree condition is really about that. And DISC is really about single edges. So you can go from-- so double-- if you half, you get much more power. So it's going in the right direction, going downstream, so to speak. So that's what we're doing now, going downstream. And then you go upstream via the counting lemma.

All right. Let's do codegree implies DISC. So we want to show the discrepancy condition, which is one written up there. But before that, let me first show you that the degrees do not vary too much, show that the degrees are fairly well distributed, which is what you should expect in a pseudorandom graph.

So you don't expect the half the vertices, half in degrees, twice the other half. So that's the first thing I want to establish. If you look at degrees, this variance, this deviation, is not too big.

OK. So like before, we see an absolute value sign. We see a sum. So we'll do Cauchy-

Schwarz. Cauchy-Schwarz allows us to bound this quantity, replacing the summand by a sum of squared.

I have a square, so I can expand the square. So let me expand the square. And I get that, so just expanding this square inside.

And you see this degree squared is that picture, so that sum of codegrees. And sum of the degrees is just the number of edges. But we now assume the codegree condition, which in particular implies that the sum of the codegrees is roughly what you would expect. So the sum of the codegrees should be p squared n cubed plus a little o n cubed error at the end.

Likewise, the number of edges is, by assumption, what you would expect in a random graph. And then the final term. And like before-- and of course, it's good to do a sanity check-- everything should cancel out.

So what you end up with is little o of n squared, showing that the degrees do not vary too much. And once you have that promise, then we move onto the actual discrepancy condition. So this discrepancy can be rewritten as the sum over vertices little x and big X, the degree from little x to y minus p times the size of y, so rewriting the sum.

And of course, what should we do next? Cauchy-Schwarz. Great. So we'll do a Cauchy-Schwarz. OK, so here's an important step or trick, if you will. So we'll do Cauchy-Schwarz.

And something very nice happens when you do Cauchy-Schwarz here. OK. So you can write down the expression that you obtain when you do Cauchy-Schwarz. So let me do that first.

OK. So here's a step which is very easy to gloss over. But I want to pause and emphasize this step, because this is actually really important. What I'm going to do now is to observe that the summand is always non-negative. Therefore, I can enlarge the sum from just little x and X to the entire vertex set.

And this is important, right? So it's important that we had to do Cauchy-Schwarz first to get a non-negative summand. You couldn't do this in the beginning. So you do that. And so I have this sum of squares. I expand. I expand. I write out all these expressions. And now the little x range over the entire vertex set.

All right. So what was the point of all of that? So you see this expression here, the degree from little x to big Y squared, what is that? How can we rewrite this expression? So counting little x and then Y squared--

AUDIENCE: Sum over u and big Y.

PROFESSOR: Yeah. So sum of codegree of two vertices in Y, so Y, Y prime, and Y codegree of little y, little y prime. And likewise, the next expression can be written as the sum of the degrees of vertices in Y. And the third term, I leave unchanged.

So now we've gotten rid of these funny expressions where it's just degree from the vertex to a set. And we could do this because of this relaxation up here. So that was the point. We had to use this relaxation so that we get these codegree terms.

But now, because you have the codegree terms and we assume the codegree hypothesis, we obtain that this sum is roughly what you expect as in a random case, because all the individual deviations do not add up to more than little o n cubed. That codegree sum is what you expect. And the next term, the sum of degrees, is also, by what we did up there, what you expect.

And finally, the third term. And as earlier, if you did everything correctly, everything should cancel. And they do. And so what you get at the end is little o of n squared.

This completes this fourth cycle. Any questions so far? So we're missing one more condition, and that's the eigenvalue condition. So far, everything had to do with counting various things. So what does eigenvalue have to do with anything?

So the eigenvalue condition is actually a particularly important one. And we'll see more of this in the next lecture. But let me first show you the equivalent implications. So what we'll show is that the eigenvalue condition is equivalent to the C4 condition. So that's the goal. So I'll show equivalence between EIG and C4.

So first, it implies a C4 condition, because up to-- so instead of counting C4s, which is a little bit actually not-- it's a bit annoying to do actual C4s. Just like earlier, we want to consider homomorphic copies, which are also labeled walks, so closed walks of length 4. So up to a cubic error, the number of labeled C4s is given by the number of closed walks of length 4, which is equal to the trace of the 4th power of the adjacency matrix of this graph.

And the next thing is super important. So the next thing is sometimes called a trace method. One important way that the eigenvalue, so the spectrum of a graph or matrix, relates to other combinatorial quantities is via this trace. So we know that the trace of the 4th power is equal to the fourth moment of the eigenvalues.

So if you haven't seen a proof of this before, I encourage you to go home and think about it. So this is an important identity, of course. 4 can be replaced by any number up here.

And now you have the eigenvalue condition. So I can estimate the sum. There's a principle term-- namely, lambda 1. So that's the big term. Everything else is small. And the smallness is supposed to capture pseudorandomness. But the big term, you have to analyze separately.

OK, so let me write it out like that. So the big term, you know that it is p to the 4 n to the 4 plus little o of n to the 4. OK. So next thing is what to do with the little terms. So we want to show that the contribution in total is not too big.

So what can we do? Well, let me first try something. So first, well, you see that each one of these guys is not too big. So maybe let's bound each one of them by little o of n raised to 4. But then there are n of them, so you have to multiply by an extra n.

And that's too much. That's not good enough. So you cannot individually bound each one of them. And this is a novice mistake. This is something that we actually will see this type of calculation later on in the term when we discuss Roth's theorem. But you're not supposed to bound these terms individually.

The better way to do this or the correct way to do this is to pull out just a couple-- some, but not all-- of these factors. So it is upper bounded by-- you take max of-- in this case, you can take out one or two. But you take out, let's say, two factors. And then you leave the remaining sum intact. In fact, I can even put lambda 1 back into the remaining sum.

So that is true. So what I've written down is just true as an inequality. And now I apply the hypothesis on the sizes of the other lambdas. So the one I pulled out is little o of n squared.

And now what's the second sum? That sum is the trace of a squared, which is just twice the number of edges of the graph. So that's also at most n squared.

So combining everything, you have the desired bound on the C4 count. Of course, this gives you an upper bound. But we also did a calculation before the break that shows you that the C4 bound has a lower bound, as well. So really, having the correct eigenvalue-- actually, no, this already shows you that the C4 bound is correct in both directions, because this is the main term. And then everything else is small.

OK. The final implication is C4 implies eigenvalue. For this one, I need to explore the following important property of the top eigenvalue. So there's something that we also saw last time, which is the interpretation of the top eigenvalue of a matrix interpreted as-- so this is sometimes called the Courant-Fischer criterion. Or actually, this is a special case of Courant-Fischer.

This is a basic linear algebra fact. If you are not familiar with it, I recommend looking it up. The top eigenvalue of a matrix, of a real, symmetric matrix, is characterized by the maximum value of this quadratic form. Let's say if x is a non-zero vector.

So in particular, if I set x to be a specific vector, I can lower bound lambda 1. So if we set this boldface 1 to be the all-one vector in R raised to the number of vertices of G, then the lambda 1 of the graph is at least this quantity over here.

The numerator and denominators are all easy things to evaluate. The numerator is just twice the number of edges, because you are summing up all the entries of the matrix. And the denominator is just n. So the top eigenvalue is at least roughly pn.

So what about the other eigenvalues? Well, the other eigenvalues, I can again refer back to this moment formula relating the trace and closed walks. It is at most the trace of the 4th power minus the top eigenvalue raised to the 4th power. It's the sum of the other eigenvalue raised to the 4th power.

And 4 here, we're using the 4. It's an even number, right? So you have this over here. So having a C4 hypothesis and also knowing what lambda 1 is allows you to control the other lambdas.

See, lambda 1 cannot be much greater than pn. Also comes out of the same calculation. Yep.

AUDIENCE: So [INAUDIBLE] number 1 equal to [INAUDIBLE]?

PROFESSOR: Yeah, thank you. Yeah, so there's a correction. So lambda 1 is-- so in other words, the little o is always respect to the constant density. OK, yeah. Question.

AUDIENCE: You said in the eigenvalue implies C4, you somewhere also used the lower bound to be proved [INAUDIBLE].

PROFESSOR: OK. So the question is in eigenvalue implies C4, it says something about the lower bound. So

I'm not saying that. So as written over here, this is what we have proved.

But when you think about the pseudorandomness condition for C4, it shouldn't be just that the number of C4 count is at most something. It should be that it equals to that, which would be implied by the C4 condition itself, because we know, always, it is the case that a C4 count is at least what it is compared to the random case.

So just one more thing I said was that lambda 1, you also know that it is at most pn plus little n, because-- OK. Yeah. So this finishes the proof of the Chung-Graham-Wilson theorem on quasi-random graphs. We stated all of these hypotheses, and they are all equivalent to each other.

And I want to emphasize, again, the most surprising one is that C4 implies everything else, that a fairly seemingly weak condition, this just having the correct number of copies of labeled C4s, is enough to guarantee all of these other much more complicated looking conditions. And in particular, just having the C4 count correct implies that the counts of every other graph H is correct.

Now, one thing I want to stress is that the Chung-Graham-Wilson theorem is really about dense graphs. And by dense, here, I mean p constant. Of course, the theorem as stated is true if you let p equal to 0. So there, I said p strictly between 0 and 1. But it is also OK if you let p be equal to 0. You don't get such interesting theorems, but it is still true.

But for sparse graphs, what you really want to care about is approximations of the correct order of magnitude. So what I mean is that you can write down some sparse analogs for p going to 0, so p as a function of n going to 0 as n goes to infinity.

So let me just write down a couple of examples, but I won't do all of them. You can imagine what they should look like. So DISC should say this quantity over here. And the discrepancy condition is little o of pn squared, because pn squared is the edge density overall. So that's the quantity you should compare against and not n squared. If you're comparing n squared, you're cheating, because n squared is much bigger than the actual edge density.

Likewise, the number of labeled copies of H is-- I want to put the little o 1 plus little in front, so instead of plus little o of n to the H at the end. So you understand the difference. So for sparse, this is the correct normalization that you should have, when p is allowed to go to 0 as a function of n.

And you can write down all of these conditions, right? I'm not saying there's a theorem. You can write out all these conditions. And you can ask, is there also some notion of equivalence? So are these corresponding conditions also equivalent to each other?

And the answer is emphatically no, absolutely not. So all of these equivalents fail for sparse. Some of them are still true. Some of the easier ones that we did-- for example, the two versions of DISC are equivalent. That's still OK.

And some of these calculations involving Cauchy-Schwarz are mostly still OK. But the one that really fails is the counting lemma. And let me explain why with an example.

So I want to give you an example of a graph which looks pseudorandom in the sense of DISC but has no, let's say, C3 count. It also has no C4 count, but it has no-- has the clean, correct number of triangles.

So what's this example? So let p be some number which is little o of 1 over root n so some decaying quantity with n. And let's consider Gnp. Well, how many triangles do we expect in Gnp? So let's think of p as just slightly below 1 over root n.

So the number of triangles in Gnp in expectation is-- so that's the expected number. And you should expect the actual number to be roughly around that. But on the other hand, the number of edges is also expected to be this quantity here. And you expect that the actual number of edges to be very close to it.

But p is chosen so that the number of triangles is significantly smaller than the number of edges, so asymptotically smaller, fewer copies of triangles than edges. So what we can do now is remove an edge from each copy of a triangle in this Gnp. We removed a tiny fraction of edges, because the number of triangles is much less than the number of edges. We removed a tiny fraction of edges.

And as a result, we do not change the discrepancy condition up to a small error. So the discrepancy condition still holds. However, the graph has no more triangles. So you have this pseudorandom graph in one sense-- namely, of having a discrepancy-- but fails to be pseudorandom in a different sense-- namely, it has no triangles. Yep.

AUDIENCE: Do the conditions C4 and codegree also hold here-- so the issue being from DISC to count?

PROFESSOR: Question, do the conditions C4 and codegree still hold here? Basically, downstream is OK, but upstream is not. So we can go from C4 to codegree to DISC. But you can't go upward. And understanding how to rectify the situation, perhaps adding additional hypotheses to make this true so that you could have counting lemmas for triangles and other graphs and sparser graphs, that's an important topic. And this is something that I'll discuss at greater length in not next lecture, but the one after that.

And this is, in fact, related to the Green-Tao theorem, which allows you to approve Szemerédi's theorem among the primes. The primes contain arbitrarily long arithmetic progressions, because the primes are also a sparse set. So it has density going to 0. It's density decaying, like, 1 over log n, according to prime number theorem.

But you want to do regularity method. So you have to face this kind of issues. So we'll discuss that more at length in a couple of lectures. But for now, just a warning that everything here is really about dense graphs.

The next thing I want to discuss is an elaboration of what happens to these eigenvalue conditions. So for dense graphs, in some sense, everything's very clear from this theorem. Once you have this, theorem, they're all equivalent. You can go back and forth. And you lose a little bit of epsilon here and there, but everything is more or less the same.

But if you go to sparser world, then you really need to be much more careful. And we need to think about other tools. And so the remainder of today, I want to just discuss one fairly simple but powerful tool relating eigenvalues on one hand and the discrepancy condition on the other hand.

All right. So you can go from eigenvalue to discrepancy by going down this chain. But actually, there's a much quicker route. And this is known as the expander mixing lemma.

For simplicity and really will make our life much simpler, we're only going to consider d-regular graphs. So here, d-regular means every vertex is degree d. Same word, but different meaning from epsilon regular. And unfortunately, that's just the way it is.

So d regular, and we're going to have n vertices. And the adjacency matrix has eigenvalues lambda 1, lambda 2, and so on, arranged in decreasing order. Let me write lambda as the maximum in absolute value of the eigenvalues except for the top one.

In particular, this is either the absolute value of the second one or the last one. As I mentioned earlier, the top eigenvalue is necessarily d, because you have all-ones vector as an eigenvector.

So the expander mixing lemma says that if I look at two vertex subsets, the number of edges between them compared to what you would expect in a random case-- so just like in the disc setting, but here, the correct density I should put is d over n-- this quantity is upper bounded by lambda times the root of the product of x and y. So in particular, if this lambda-- so everything except for the top eigenvalue-- is small, then this discrepancy should be small. And you can verify with what we did, that it's consistent, what we just did.

All right. So let's prove the expander mixing lemma, which is pretty simple given what we've discussed so far, relating-- so there was this spectral characterization up there of the top eigenvalue. So we can let J be the all-ones matrix. So let J be the all-ones matrix.

And we know that the all-ones vector is an eigenvector of the adjacency matrix of G with eigenvalue d. So the eigendecomposition of J is also the all-ones vector and its complement. So we now see that A sub G minus d over nJ has the same eigenvectors as AG.

So you can choose the eigenvectors for that. It's the same set of eigenvectors. Of course, we consider this quantity here, because this is exactly the quantity that comes up in this expression once we hit it by characteristic vectors of subsets from left and right. All right.

So what are the eigenvalues? So A previously had eigenvalues lambda 1 through lambda n. But now the top one gets chopped down to 0.

So you can check this explicitly. So you can check this explicitly by checking that if you take this matrix multiplied by the all-ones vector, you get 0. And if you have a eigenvector-eigenvalue pair, then hitting this by any of the other ones gets you the same as in A, because you have this orthogonality condition. All the other eigenvectors are orthogonal to the all-ones vector.

All right. So now we apply the Courant-Fischer criteria, which tells us that the number in this discrepancy quantity, which we can write in terms of this matrix, it is upper bounded by the product of the length of these two vectors, x and y, multiplied by the spectral norm. So I'm not quite using the version up there, but I'm using the spectral norm version, which we discussed last time. It's essentially the one up there, but you allow not just single x but x and y. And that corresponds to the largest eigenvalue in absolute value, which we see that. It's at most lambda. So at most lambda times size of x, size of y square root.

And that finishes the proof of the expander mixing lemma. So the moral here is that, just like what we saw earlier in the dense case but for any parameters-- so here, it's a very clean statement. You can even have done the degree graphs. d could be a constant. If lambda is small compared to d, then you have this discrepancy condition.

And the reason why this is called an expander mixing lemma is that there's this notion of expanders, which is not quite the same but very intimately related to pseudorandom graphs. So one property of pseudorandom graphs that is quite useful-- in particular, in computer science-- is that if you take a small subset of vertices, it has lots of neighbors. So the graph is now somehow clustered into a few local pieces. So there's lots of expansion.

And that's something that you can guarantee using the expander mixing lemma, that you have lots of-- you take a small subset of vertices. You can expand outward. So graphs with that specific property, taking a small subset of vertices always gets you lots of neighbors, are called expander graphs. And these graphs play an important role, in particular, in computer science in designing algorithms and proving complexity results and so on but also play important roles in graph theory and combinatorics.

Well, next time, we'll address a few questions which are along the lines of, one, how small can lambda be as a function of d? So here is this. If lambda's small compared to d, then you have this discrepancy. But if d is, let's say, a million, how small can lambda be? That's one question.

Another question is, considering everything that we've said so far, what can we say about, let's say, the relationship between some of these conditions for sparse graphs but that are somewhat special-- for example, kd graphs or vertex-transitive graphs? And it turns out some of these relations are also equivalent to each other.