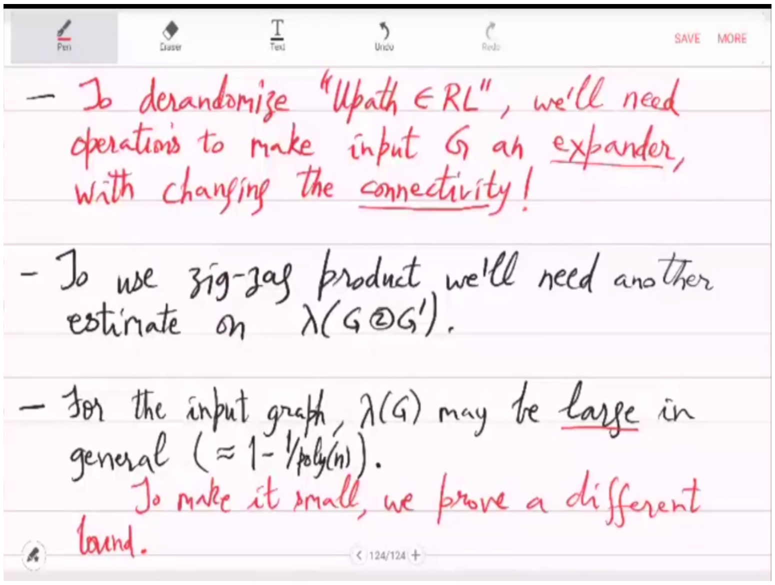
Randomized Methods in Complexity
Prof. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology - Kanpur

Lecture – 16
Undirected Graph Connectivity in Logspace

(Refer Slide Time: 00:20)



- To derandomize "Upath ∈ RL", we'll need operations to make input G an expander, with changing the connectivity!

- To use zig-zag product we'll need another estimate on $\lambda(G ⓩ G')$.

- For the input graph, $\lambda(G)$ may be large in general ($\approx 1 - 1/poly(n)$).
To make it small, we prove a different bound.

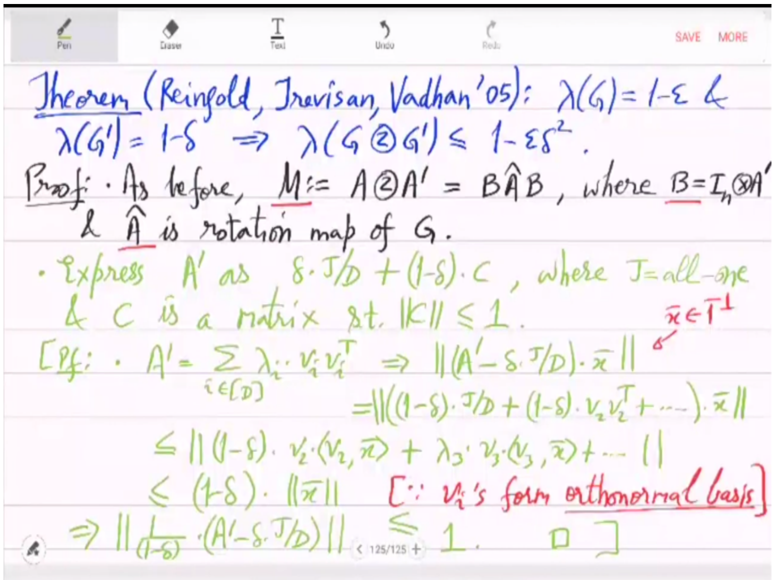
Last time we finished the analysis on zig-zag product. So, we saw that using these 3-4 types of products we can construct strongly explicit constant, constant expanded family which means that the degree is constant and second largest eigenvalue is also constant and hence it is away from 1, so spectral gap is constant. We saw this construction. So, that solves our one major problem.

The second major problem now that we will solve using expanders is undirected graph connectivity Upath. So, we have already seen that random walk solves you path in randomized logspace. So, we will now use expander construction or we will actually use these products to show that input graph connected graph can be made an expander and hence if there is a path from s to t, it will become very small, it will become logarithmic.

So, this we will achieve by various products. So, for the input graph this $\lambda(G)$ maybe large. We only proved a bound of $1 - 1/poly(n)$, so that is not good enough. So, spectral gap is $1/poly(n)$, it is not constant, so we want to make it constant. So, to make it constant now we will use the product systematically and increase the spectral gap to constant from 1 over poly. So, from inverse poly we will go to constant.

So, the thing is the theorems you saw about spectral gap, right. So, you saw this last theorem said that if the expanders are $\lambda(1)$, $\lambda(2)$ then from that you get $\lambda(1) + 2\ \lambda(2)$. So, these kinds of operations will not give you inverse poly to constant. So, inverse poly will remain inverse poly. So, we actually need a different bound to achieve this. So, to make it small we prove a different bound. Let us start that.

(Refer Slide Time: 03:42)



<u>Theorem</u> (Reingold, Trevisan, Vadhan '05): $\lambda(G) = 1-\varepsilon$ & $\lambda(G') = 1-\delta \Rightarrow \lambda(G \textcircled{z} G') \leq 1-\varepsilon\delta^2$.

<u>Proof</u>: • As before, $\underline{M} := A \textcircled{z} A' = B\hat{A}B$, where $\underline{B} = I_n \otimes A'$ & $\underline{\hat{A}}$ is rotation map of G.

• Express A' as $\delta \cdot J/D + (1-\delta)\cdot C$, where J = all-one & C is a matrix s.t. $\|C\| \leq 1$.

[Pf: • $A' = \sum_{i\in[D]} \lambda_i \cdot v_i v_i^T \Rightarrow \|(A' - \delta \cdot J/D)\cdot \bar{x}\|$ $\overset{\bar{x}\in \mathbb{1}^{\perp}}{}$

$= \|((1-\delta)\cdot J/D + (1-\delta)\cdot v_2 v_2^T + \cdots)\cdot \bar{x}\|$

$\leq \|(1-\delta)\cdot v_2 \cdot \langle v_2, \bar{x}\rangle + \lambda_3 \cdot v_3 \cdot \langle v_3, \bar{x}\rangle + \cdots\|$

$\leq (1-\delta)\cdot \|\bar{x}\|$ [$\because$ v_i's form orthonormal basis]

$\Rightarrow \|\frac{1}{(1-\delta)}\cdot(A' - \delta \cdot J/D)\| \leq 1$. □]

So, this is again a theorem by Reingold, Trevisan and Vadhan and the previous estimate was by Reingold, Vadhan and Wigderson. So, that told you a b goes to $A + 2B + B^2$ this will be slightly different. So, $\lambda(G) = 1 - \varepsilon$ & $\lambda(G') = 1 - \delta \Rightarrow \lambda(G \textcircled{z} G') = 1 - \varepsilon\delta^2$. So, this is the regime wherenow we will work.

So, we will work with a G whose λ is close to 1, so it is like $1 - \varepsilon$. It is ε close to 1 and G' similarly or you can think in terms of the spectral gap. So, spectral gap you started with this ε and G' we will use to reduce the degree so that has a spectral gap δ. Then the spectral gap of the zig-zag product will be $\varepsilon\delta^2$. So, it is not immediately clear how this will help, but it will.

So, let us first prove this theorem. So, basically you can think of the difference being in the earlier theorem being additive and this being multiplicative. So, the previous spectral analysis gave you this additive behavior

A + B and what this result is giving you is in the spectral gap multiplicative and that will somehow be helpful. Let us first prove this estimate. As expected, this will go via matrix analysis, so eigenvectors and eigenvalues and norms matrix norm spectral norm that we have defined.

So, let us restart. So, as before this $M = A \textcircled{z} A' = B\hat{A}B$, this is the normalized adjacency matrix of the zig-zag product which can also be written as like we did before $B\hat{A}B$, So, B again refers to this inside the cloud, step, then outside the cloud is $\hat{A}$ which will follow the graph G and again inside the cloud is B which is following G'. So, B we have seen is tensor product. $B = I_n \otimes A'$

So, this is just a blown up version of the graph G' and $\hat{A}$ is rotation map, $\hat{A}$ is the rotation map of G. So, recall these three things that is the zig-zag product. Now, expressA' as A' is this dot G' coming from G'. So, this you express as again you remember this main term error decomposition, which we did in the previous proof. $Express\ A'\ as\ \delta.J/D + (1 - \delta).C\ , where\ J = all - one\ \&C\ is\ a\ matrix\ s.t\ ||k|| \leq 1$. So, this you can show as follows. So, recall that $A' = \sum_{i\epsilon[D]} \lambda_i .. v_i v_i^T$. A' was a D / D matrix and its eigenvalues are$\lambda_1 to\ \lambda_D$in decreasing order of magnitude and v $_i$, v $_1$ to v$_D$ these are the corresponding eigenvectors which form an orthonormal basis.

So, if you look at $||A' - \delta.J/D||.\bar{x} = ||(1 - \delta).J/D + (1 - \delta)v_2 v_2^T$ +.....) $\bar{x}||$ then so on that is the expression you get. So, $v_1 v_1^T$ gives you J / D and then$v_2 v_2^T$ the eigenvector corresponding eigenvector $\lambda_2 = 1 - \delta$ and so on. So, you can see that $||A' - \delta.J/D||.\bar{x} = ||(1 - \delta).J/D + (1 - \delta)v_2 v_2^T$ +.....) $\bar{x}||$. So, you remember how did we define the matrix norm in the previous proof that is the definition of matrix norm.

$||A|| = max||Ax||/||x||$or in other words the shrinkage when you multiply by A ,a unit vectors orthogonal to $\bar{1}$. So, we are taking here $\bar{x}$ to be orthogonal to 1. So, with that assumption J $\bar{x}$ actually vanishes. So, what you are left with is $v_2 v_2^T$ and so on. So, what can you say about this? Now,

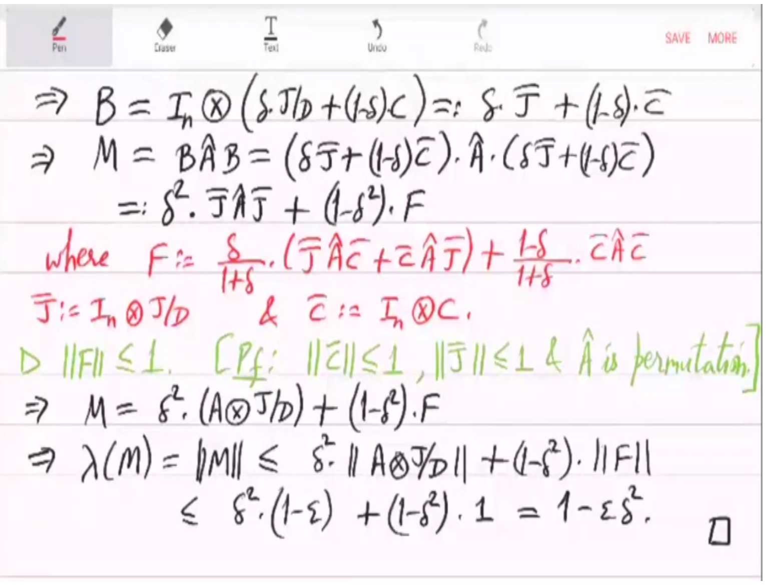
$||A' \ - \ \delta.J/D||.\bar{x} = ||(1 \ - \ \delta).J/D \ + \ (1 \ - \ \delta)v_2 v_2^T \ \ +.....) \ \bar{x}|| \le ||(1 \ - \ \delta).v_2 < v_2, \bar{x} >+ \lambda_3.v_3 < v_3, \bar{x}$

this calculation also we have kind of seen before if you recall.

Again, this calculation in green that we did in this slide it is similar calculation. So, from this type of calculation what you will get is that this is $\le \ (1 \ - \ \delta).\ ||\bar{x}||$. So, we are also using here the fact that $(1 \ - \ \delta)$ is the biggest value amongst λ_2, λ_3 and so on. And the remaining things cannot give you something bigger than the norm of x where we use the fact that v $_i$'s form orthonormal basis.

In particular, it means that $||\frac{1}{(1-\delta)}(A' \ - \ \delta.J/D)|| \le 1$ that is the thing we have proved. So, which is why talking about C as the error term is correct because we know that its spectral norm is less than or equal to 1. So, let us now begin with this decomposition, main term plus error term decomposition of A' and substitute this in B.

(Refer Slide Time: 14:52)



$$\Rightarrow \ B = I_n \otimes (\delta . J/D + (1-\delta) C) =: \delta . \bar{J} + (1-\delta) . \bar{C}$$

$$\Rightarrow \ M = B\hat{A}B = (\delta\bar{J} + (1-\delta)\bar{C}) . \hat{A} . (\delta\bar{J} + (1-\delta)\bar{C})$$

$$=: \delta^2 . \bar{J}\hat{A}\bar{J} + (1-\delta^2) . F$$

$$\text{where } \ F := \frac{\delta}{1+\delta} . (\bar{J}\hat{A}\bar{C} + \bar{C}\hat{A}\bar{J}) + \frac{1-\delta}{1+\delta} . \bar{C}\hat{A}\bar{C}$$

$$\bar{J} := I_n \otimes J/D \quad \& \quad \bar{C} := I_n \otimes C .$$

$$\triangleright \ ||F|| \le 1 . \quad [\underline{Pf}: \ ||\bar{C}|| \le 1 , ||\bar{J}|| \le 1 \ \& \ \hat{A} \text{ is permutation}]$$

$$\Rightarrow \ M = \delta^2 . (A \otimes J/D) + (1-\delta^2) . F$$

$$\Rightarrow \ \lambda(M) = ||M|| \le \ \delta^2 . ||A \otimes J/D|| + (1-\delta^2) . ||F||$$

$$\le \ \delta^2 . (1-\varepsilon) + (1-\delta^2) . 1 = 1 - \varepsilon\delta^2 . \qquad \square$$

So, $B \ = I_n \otimes \ (\delta.J/D \ + \ (1 \ - \ \delta).C) \ =:\delta\bar{J} + \ (1 \ - \ \delta) \ \bar{C}$ So, $\bar{J}$ is essentially this blown up version of J/ D tensor with I $_n$ and $\bar{C}$r is blown up version of C tensor with I $_n$. So, remember that definition. And now we can work with M, So, M = $B\hat{A}B$=$(\delta\bar{J} + \ (1 \ - \ \delta) \ \bar{C}).\hat{A}(\delta\bar{J} + \ (1 \ - \ \delta) \ \bar{C})$

$$=\delta^2\bar{J}\hat{A}\bar{J} + \ (1 - \delta^2).F$$

Where,$F: = \frac{\delta}{1+\delta}(\bar{J}\hat{A}\bar{C} + \bar{C}\hat{A}\bar{J}) + \frac{1-\delta}{1+\delta}.\bar{C}\hat{A}\bar{C}$

$\bar{J}: = I_n \otimes J/D$ &$\bar{C}: = I_n \otimes C$

. We have written M as main term plus error term and error term F is just a combination of three matrices and you can also see that it is a convex combination. So, the ratios are delta over 1 + delta the same and then 1 – delta upon 1 + delta. If you sum it up, you will get 1. So, it is a convex combination of these three errors that is also what we are preserving.

And the claim here is again $||F|| \leq 1$, why is that? So, it is because we have already shown that $||C|| \leq 1$ is and that would mean $||\bar{C}|| \leq 1$. $||\bar{J}||$ will depend on J / D which you can see is also 1, you know $||\bar{J}|| \leq 1$ because, so we have this you can check. It is an easy check,.

So, what about a hat? So, $\hat{A}$ it is a permutation matrix. So, from these three conditions you should be able to deduce that this $\bar{J}\hat{A}\bar{C}$,$\bar{C}\hat{A}\bar{J}$ and $\bar{C}\hat{A}\bar{C}$all of them have norm at most 1 and that means the convex combination also has norm at most 1. So, F has norm at most 1. So, that means $M = \delta^2(A \otimes J/D) + (1 - \delta^2).F$, so you further continue this breakup mean term plus error term breakup.

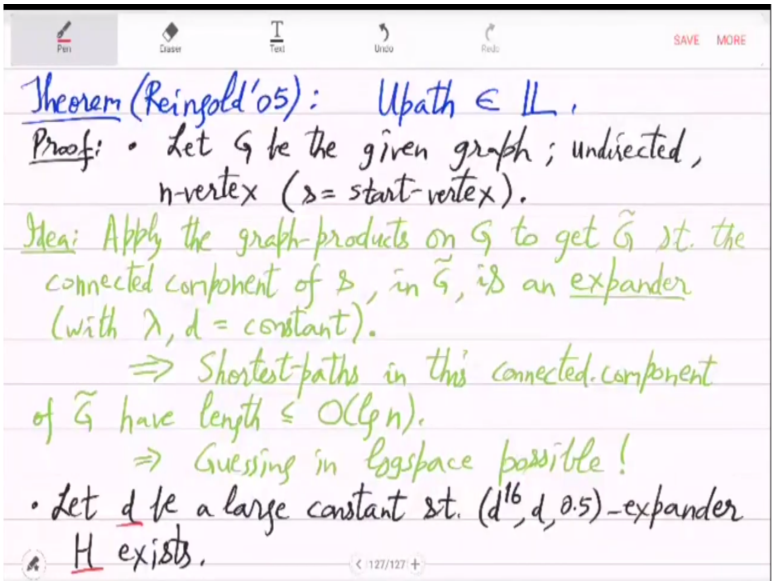
$\lambda(M) = ||M|| \leq \delta^2||A \otimes J/D|| + (1 - \delta^2).||F||$, you get this. If you apply the trivial inequalities, then you will just get $\delta^2 + 1 - \delta^2$ which is only 1. So that will be no fun. So, what you want is away from 1 and for that you have to use the hypothesis on A.

So, you know that hypothesis for A is 1 – , so use that that will give you something nontrivial $\lambda(M) = ||M|| \leq \delta^2||A \otimes J/D|| + (1 - \delta^2).||F|| \leq \delta^2(1 - \varepsilon) + (1 - \delta^2).1 = 1 = \varepsilon\delta^2$ that is what we wanted to show.

So, we have shown that this zig-zag product matrix if you start with εδ, then you get 2 $\varepsilon\delta^2$you get this multiplicative behavior. So, remember that. Now using this we can solve undirected graph reachability problem in logspace. They kept saying that this is main term error term, but if you carefully look at this, $(1 - \delta^2).$ 1this term in fact is very close to 1 this is quite large and so this is the part which is actually of interest. $\delta^2(1 - \varepsilon)$ This is the negative part.

So, in fact it was the opposite, this main thing which we were trying to understand that is the negative part. This is what tells you how much below 1 you can get and we wanted to get this thing. So, close to part 1 we have separated that is kind of easy to handle and the negative part is what we wanted to understand So, that we have understood to be $-\varepsilon\delta^2$. So, maybe that helps you in going through this proof again.

(Refer Slide Time: 24:12)



Theorem (Reingold'05): Upath ∈ L.

Proof: • Let G be the given graph; undirected, n-vertex (s= start-vertex).

Idea: Apply the graph-products on G to get $\tilde{G}$ s.t. the connected component of s, in $\tilde{G}$, is an expander (with λ, d = constant).

$\Rightarrow$ Shortest-paths in this connected component of $\tilde{G}$ have length $\leq O(\lg n)$.

$\Rightarrow$ Guessing in logspace possible!

• Let d be a large constant s.t. $(d^{16}, d, 0.5)$-expander H exists.

So, this is a simple statement but a very advanced proof that Upath $Upath\in L$e. So, of towns or places in a town in a city a big map, what this theorem is saying that even if you have a very small amount of memory, just logarithmically small amount of memory still you can go from place to place. This is an amazing result. It is deterministic and it is not very old, relatively recent. So let G be the input graph.

It is undirected n-vertex, s=start vertex and you are interested in the connected component of s and the idea is that apply the graph products on G to get $\overline{G}$ such that the connected component of s, so basically you are moving from transforming G to another graph $\overline{G}$ which will be bigger. It has better expansion and better connectivity and vertex s will have in fact for vertices s and t there will be corresponding vertices s and t in $\overline{G}$.

And there will also be a connected component of s that will be an expander. So, with λ which is the second biggest eigenvalue of $\overline{G}$ and d is the degree both of them are constant. Which means so if the degree is constant and λ is constant, then the spectral norm is also constant and if you recall the random walk analysis then you know that in log n steps, n is the number of vertices in $\overline{G}$, in log n steps you will reach almost everywhere.

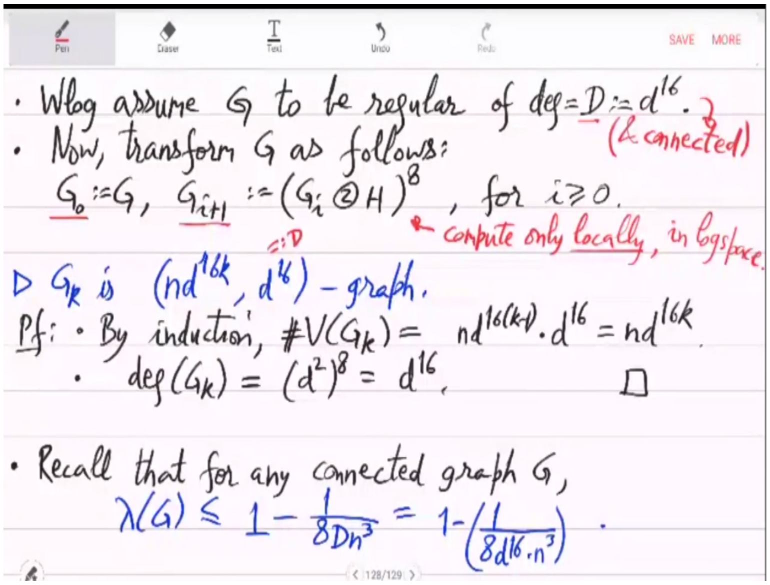
You will reach everywhere with almost the same probability in $\overline{G}$ in log n steps, which means that obviously the shortest distances in the graph $\overline{G}$ that is at least the connected component of $\overline{G}$ the diameter is log. In this connected component have length $\leq O($ log n) that is the key idea that you have made the paths very small. Once the paths are very small, then you can actually and also the degree is constant.

So, you just have to guess a path in terms of neighbors doing this walk d times log n many bits. In fact log d times log n many bits which is constant times log n. So, this you can easily guess in logspace. So, you guess it so which means that you will actually get this path from s to t, you have solved connectivity logspace. So, all I have to do now is I have to give you the sequence of products getting you from G to $\overline{G}$.

So, let d be a large constant such that (d^{16}, d, 0.5)-expander exists. Why these parameters? So just note that the size of the graph is quite big, it is d^{16} compared to the degree being just d. So, you can actually show that if you pick d to be a big enough constant like 10 or 20 or 30, then random graphs of this type are actually expanders and expanders with spectral gap 0.5.

So, they exist, so just fix d to be such a constant. So, this H is a constant size, constant degree, constant spectral norm expander graph which you have designed somehow or you have just found it by a probabilistic process. So, it is a one-time investment and this is what now we will use again and again on the input graph G.

(Refer Slide Time: 31:23)



- Wlog assume G to be regular of $deg = \underline{D} := d^{16}$. (& connected)
- Now, transform G as follows:
  $\underline{G_0} := G$, $\underline{G_{i+1}} := (G_i \textcircled{z} H)^8$, for $i \geq 0$.
  ← compute only locally, in logspace.

▷ G_k is $(nd^{16k}, \overset{=:D}{d^{16}})$ – graph.

Pf: • By induction, $\#V(G_k) = nd^{16(k-1)} \cdot d^{16} = nd^{16k}$.
• $deg(G_k) = (d^2)^8 = d^{16}$. □

- Recall that for any connected graph G,
  $\lambda(G) \leq 1 - \frac{1}{8Dn^3} = 1 - \left(\frac{1}{8d^{16} \cdot n^3}\right)$.

So, assume G to be regular, this we can assume because if it was not regular then you can make it by these small gadgets that we saw in the very beginning and in fact the degree also you can make whatever you want just at least 3. So, let us make it of degree D =d^{16} because as you can guess we want to use this zig-zag product with H. So, H has size d ^{16}, your input graph G has degree d ^{16}.

So that degree will be reduced to D by when you take the zig-zag product or will be reduced to d^2 so we will see that. Now transform. $G_0 := G, G_{i+1} := (G_i \textcircled{z} H)^8$. So that is the key transformation, zig-zag product to reduce the degree and then path product to reduce λ and then again repeat the same.

$G_k\ is\ (nd^{16k}, d^{16}) - graph.$

So, let us first look at the vertices and degree how many vertices does G_k have?

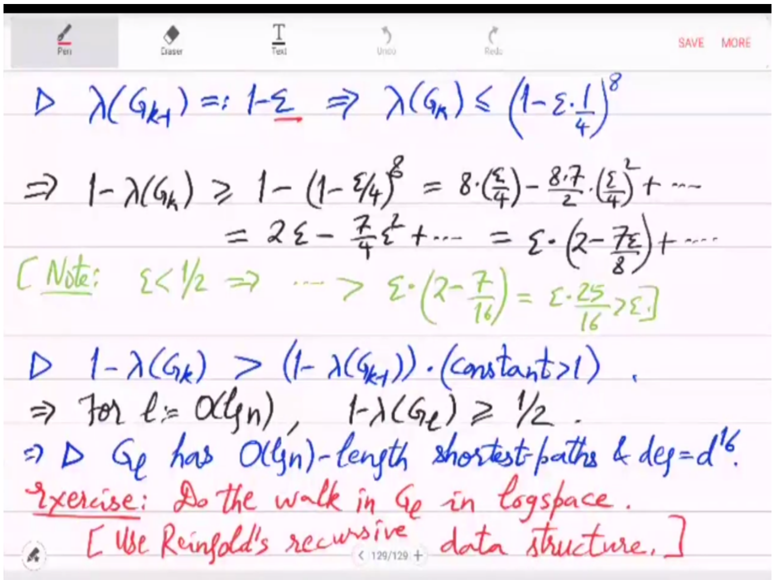
$By\ induction,\ \#V\ (G_k) = nd^{16(k-1)}.d^{16} = nd^{16k}$

$deg(G_k) = (d^2)^8 = d^{16}$

So seems to be a big graph, big degree, but since d is constant it is technically a constant, so constant times, I mean in K iterations every time you are multiplying by a constant. So, you get n times constant to the K and that is the size and the degree is just constant no matter what K. Now recall that for any connected graph, in fact we can also assume here without loss of generality that it is connected.

Just assume this because we are interested in one connected component of G, we do not care about the disconnected components. So, G is a connected regular graph and for a connected graph recall that we have a bound forλ. So, λ(G) is away from 1. $\lambda(G) \leq 1 - 1/8Dn^3 = 1 - (1/d^{16}n^3)$.

remember that small d is a constant. So, this is like 1 – 1/ n^3. So spectral gap you already are starting with the 1/ n^3, but you want to increase it to constant. So, how will that be done? Let us do that.



▷ $\lambda(G_{k-1}) =: 1-\varepsilon \Rightarrow \lambda(G_k) \leq \left(1-\varepsilon\cdot\frac{1}{4}\right)^8$

$\Rightarrow 1-\lambda(G_k) \geq 1-(1-\varepsilon/4)^8 = 8\cdot\left(\frac{\varepsilon}{4}\right) - \frac{8\cdot7}{2}\cdot\left(\frac{\varepsilon}{4}\right)^2 + \cdots$

$= 2\varepsilon - \frac{7}{4}\varepsilon^2 + \cdots = \varepsilon\cdot\left(2-\frac{7\varepsilon}{8}\right) + \cdots$

[Note: $\varepsilon < 1/2 \Rightarrow \cdots > \varepsilon\cdot\left(2-\frac{7}{16}\right) = \varepsilon\cdot\frac{25}{16} > \varepsilon$.]

▷ $1-\lambda(G_k) > (1-\lambda(G_{k-1}))\cdot(\text{constant}>1)$.

$\Rightarrow$ For $\ell := O(\lg n)$, $1-\lambda(G_\ell) \geq 1/2$.

$\Rightarrow$ ▷ G_ℓ has $O(\lg n)$-length shortest-paths & deg $= d^{16}$.

Exercise: Do the walk in G_ℓ in logspace.

[Use Reingold's recursive data structure.]

(Refer Slide Time: 38:21)

$\lambda(G_{k-1}) =: 1 - \varepsilon \Rightarrow \lambda(G_k) \leq (1 - \varepsilon\frac{1}{4})^8$.

So you will get this. So, what is happening here is when you go from $1 - \varepsilon$ to 1 –ε/ 4, you went closer to 1 so which is making it worse. You get closer to 1 but then you multiply it 8 times and you are again far away from 1. So, you take some bad steps, but then many good steps that is what has happened. So let us now analyze this. So, what is the spectral gap $1 - \lambda(G_k) \geq 1 - (1 - \frac{\varepsilon}{4})^8 = 8.(\frac{\varepsilon}{4}) - \frac{8.7}{2}.(\frac{\varepsilon}{4})^2 +$............

$=2\varepsilon - \frac{7}{4}\varepsilon^2 +$........ $= 2\varepsilon(1 - \frac{7\varepsilon}{8}) +$.......

So, what we see here is if we take epsilon to be small what should we compare this with? We should compare with epsilon because you started with epsilon, you started with this gap.

So, let us compare this clearly with epsilon. $1 – \lambda(G_k) \geq 1 - (1 - \frac{\varepsilon}{4})^8 = 8.(\frac{\varepsilon}{4}) - \frac{8.7}{2}.(\frac{\varepsilon}{4})^2 +$………… $=2\varepsilon - \frac{7}{4}\varepsilon^2 +$…….. $= 2\varepsilon(2 - \frac{7\varepsilon}{8}) +$……. so this is ε times some fraction which also depends on epsilon, but nevertheless you note that if your goal is to go above epsilon you want to improve the spectral gap. So, therefore $\varepsilon < 1/2$, the above is $>\varepsilon(2 - \frac{7}{16}) = \varepsilon.\frac{25}{16}$which is; the problem is I am ignoring the other terms. So, I have to be a bit more careful here, anyways so this will be increasing by a lot.

So, this is something like this. So, basically if <1/2, then this $1 – \lambda(G_k)$ is a good improvement on ε. You started with ε as the spectral gap but of ε will be multiplied by some factor which is bigger than ε. So, what you learn is $1 – \lambda(G_k) > 1 – \lambda(G_{k-1})$ times a constant. So, every time the spectral gap is improving by a constant factor and obviously > 1.

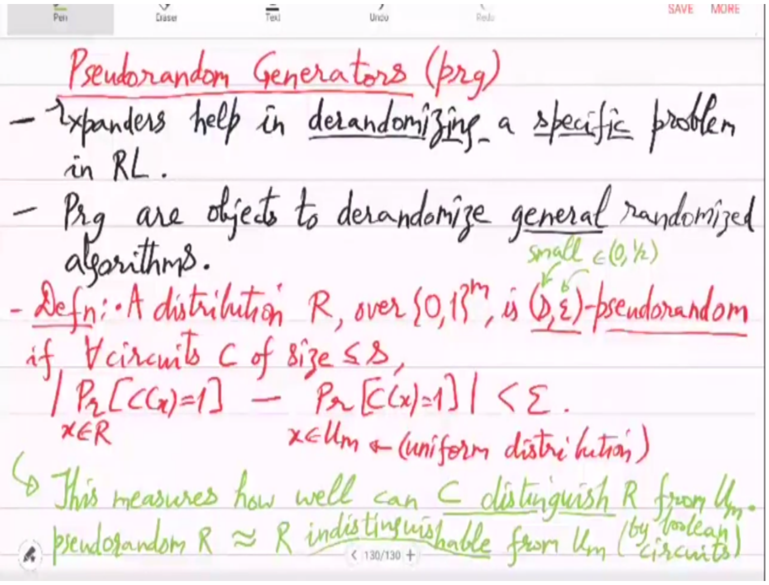
It is always increasing. So, to go from 1/ n or $1/n^3$ to a constant it will be achievable in log n many creations. $for\ l = O(logn),\ 1 - \lambda(G_l) \geq 1/2$ So, $\lambda(G_l)$ will be smaller will be at most 0.5 that is it. So, you have made a general graph and expander with constant parameters, constant degree, constant expansion.

So, $G_l\ has\ O(logn)$ length shortest paths and degree d^{16} which is also a constant. So, now you can do path search s to t in this graph $G_l\ has\ O(logn)$ in logspace. So, what I leave as an exercise now is do the walk in G_l So, you have to do all these operations locally. You cannot store the whole thing, you do not have space or time for that. So, you have to just compute these functions in a very local way. So, you need some kind of a recursive data structure and given that you have to do the walk. So, there is some implementation details required here. So, this is to be compute only locally and then that is possible in logspace.

Do not try to compute everything because that is not possible, it is a very big graph compared to the space that you have. So, there is some technicality, data structure is to be invented and applied here, but the expander part I hope is absolutely clear that is the main idea to make this general graph and expander. And then the problem kind of trivializes because the shortest paths are very short.

So, that finishes the topic of expanders. You can go back and read this again, these are beautiful objects. So, I hope you enjoyed this. Next thing will be in a way even more better.

(Refer Slide Time: 48:51)



Pseudorandom Generators (prg)

- Expanders help in derandomizing a specific problem in RL.
- Prg are objects to derandomize general randomized algorithms.
- Defn: A distribution R, over $\{0,1\}^m$, is (s,ε)-pseudorandom if $\forall$ circuits C of size $\leq s$,

small $\in (0, \frac{1}{2})$

$$\left| \Pr_{x \in R}[C(x)=1] - \Pr_{x \in U_m}[C(x)=1] \right| < \varepsilon .$$

$\leftarrow$ (uniform distribution)

↳ This measures how well can C distinguish R from U_m.

pseudorandom R ≈ R indistinguishable from U_m (by boolean circuits)

So next topic or next objects that we will study are called pseudorandom generators, so we call it prg. So what are prg's? So what was the use of expanders? If you see it in terms of the use, expanders you use to derandomize logspace and not even the whole logspace just a specific problem, so in derandomizing a specific problem in RL. So, expanders are these extremely nice objects that help you in derandomizing the random walk algorithm for a specific problem.

Which is the undirected graph connectivity at the level of logspace. So, you can ask the question what would it take to derandomize all problems take the whole complexity class? So, prg are objects to derandomize general randomized algorithms. So, if there is a randomized algorithm it is asking for random bits or coin tosses, then prg would be an algorithm.

You can first think of it in terms of an algorithm or a function easy to compute which will simulate the coin toss very fast efficiently and then give this quote unquote coin toss to your randomized algorithm. And when you look at your algorithm together with this subroutine of prg you will see a deterministic polynomial time algorithm. So prg's are that general. So, how is such a thing defined and what can we prove about these objects and then can we construct these objects?

So you notice that construction of expanders was not easy. This required a lot of theory, a lot of matrix analysis. That work is not yet complete for prg's So we do not know constructions for prg, but there is a lot of initial theory connecting prg's with other objects, other objects with prg, implications of prg and that is what we will discuss in this chapter over the next several weeks. So first we define a distribution.

So, a distribution R over$\{0,1\}^m$ is (s, ε) $-$pseudorandom. So, first we define the key word pseudorandom. So, you already hopefully know what is random. Random just it is a mathematical notion, we do not know how to do it in practice, how to implement it in practice, although a coin toss is an approximation to that in practice. Mathematically, we just say that we have a box full of balls and we just without looking at the balls we randomly pick a ball.

And then what is the probability of picking a certain ball? So, you look at the number of favorable cases divided by the number of all cases. So, it is just a mathematical formalism definition, certainly a mathematical concept, but what is the mathematical concept of pseudorandom that also the word pseudo also we have to define. So, we define it in terms of two parameters s, epsilon, s is the kind of the resource or the algorithmic power against which you want to prove randomness.

So, in our case s will be circuit size, ε will be just a fraction close to 0 denoting the error away from uniform probability. So we say that a distribution R is (s, ε) $-$ pseudorandom for all circuits C of size $\leq$ s, $Pr_{x \epsilon R}[C(x) = 1] - Pr_{x \epsilon \{0,1\}^m}[C(x) = 1]$ So, what is this difference? How far is the distribution from the uniform distribution with respect to circuits of sizes?

And we are talking about Boolean circuits,$Pr_{x \epsilon R}[C(x) = 1] - Pr_{x \epsilon U_m}[C(x) = 1] < \varepsilon$. so what is the idea of this? So, this measures how well can C distinguish R from U m. So, is there a small circuit by small v mean s that can distinguish the distribution R from U_m ? If there is none if R is indistinguishable from U_m then we call R pseudorandom.

So, is R indistinguishable from U_m up to small circuits that is what this above definition means. If you are seeing it first time it may look complicated, but there are multiple interpretations. This is the best interpretation that for two distributions I mean for distribution R we will call it pseudorandom if small circuits are unable to distinguish. So, this goes back to this protocol Arthur-Merlin protocol that we studied.

So, essentially this R will look random to Arthur because Arthur will be a restricted computing machine. So, maybe for Merlin it is not, Merlin will be able to distinguish it from U_m, but for Arthur it looks pseudorandom. So, this you can think of as small and ε you can think of as close to 0, it is a fraction. So based on this, we will define prg's.