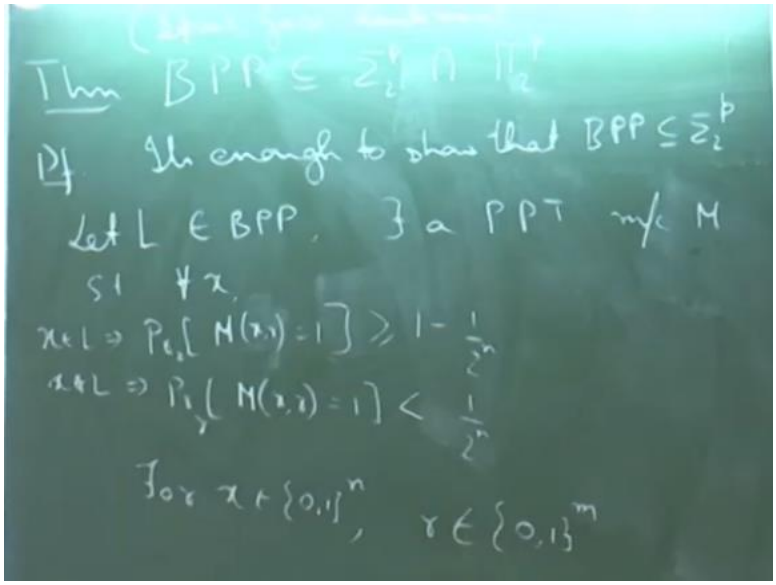
Computational Complexity Theory
Prof. Raghunath Tewari
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Module No # 05
Lecture No # 22
BPP in PH and Logspace Randomized Classes

Last time we say that BPP is computable by polynomial size circuits in other words it is p by poly so we will look at an, another important result concerning BPP.

(**Refer Slide Time: 00:30)**

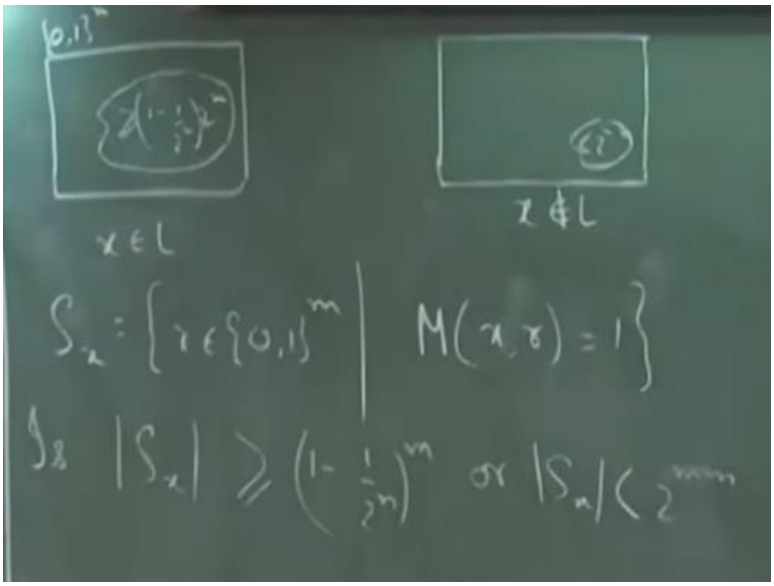


So what we will show today is that BPP is also contained in sigma 2 inter section Pi 2. So remember that when we first introduced BPP we discussed that why we do not know whether BPP is contained in n p or not. Because of this 2 sided error thing so that is I mean that is still an open problem whether BPP is in n p or whether BPP is in co n p or not. But this is actually known so and this was due to the following people so Sipser.

I think in the late seventies we proved that BPP is in the polynomial hierarchy and then this result was due to Gac's and Laute man in 83. So Gac's actually gave the first proof that BPP is in sigma 2 intersect by 2 but Laute man subsequently gave a simpler proof in fact the proof that we will discuss now is due to Laute man. So I mean so BPP is closed under complement it is enough to show that this is in just sigma 2 because BPP is equal to co BPP.

So the idea is as follows so the idea is to, basically and let us also assume that the random string that this machine uses has learned some f. So suppose we are looking at for x and 0 1 to the power n r belongs to 0, 1 to the power n. So since the machine that we are considering is a polynomial time machine m cannot be more than polynomially larger as a function of n.

(Refer Slide Time: 04:50)



So the idea is as follows so let us look at the space of all random strings. So this is the set of all strings of length m how many good strings do we have in this strings? Suppose we have a instance which in the language for that instance how many strings are there which makes the machine to give a correct answer? So is that question clear? So basically I mean r can be any of these 2 to the power n strings.

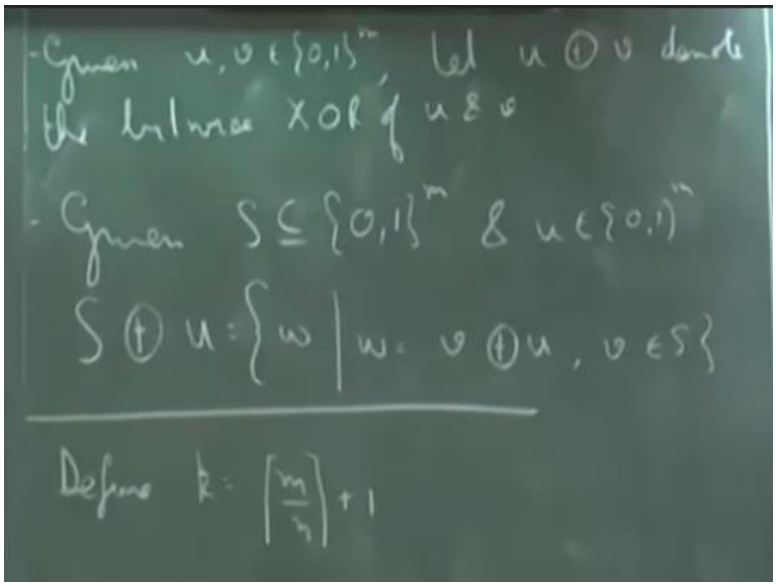
So suppose if you pick a string in the language then what fraction of these 2 to the power n strings will make m accept x correctly at least 1 – 1 over 2 to the power n. So basically it is quite a large fraction so this fraction so this has size at least 1 – 1 over 2 to the power m times 2 to the power m correct. On the other hand so this is for correct let me say that for x in l this is the set of random strings which we make it give a correct answer or it will make it accept.

On the other hand suppose if you have an instance which is not in the language then how many strings will need m access? It is less than 1 over it is some small fraction which is smaller than 2 to the power m – n for x naught in l. So the idea is to basically distinguish between 2 cases. So

suppose I can construct the sigma 2 machine which can distinguish between these 2 cases then I am done.

So I will just give some names to these sets so let so given an x let S x be the set of all random strings such that m accepts x with the help of that random string. So what I want to distinguish is? Is S of x greater than this number or smaller than 2 to the power m – n. So how do we go about this? So before we proceed with the proof let us take a D tour and look at the some definitions.

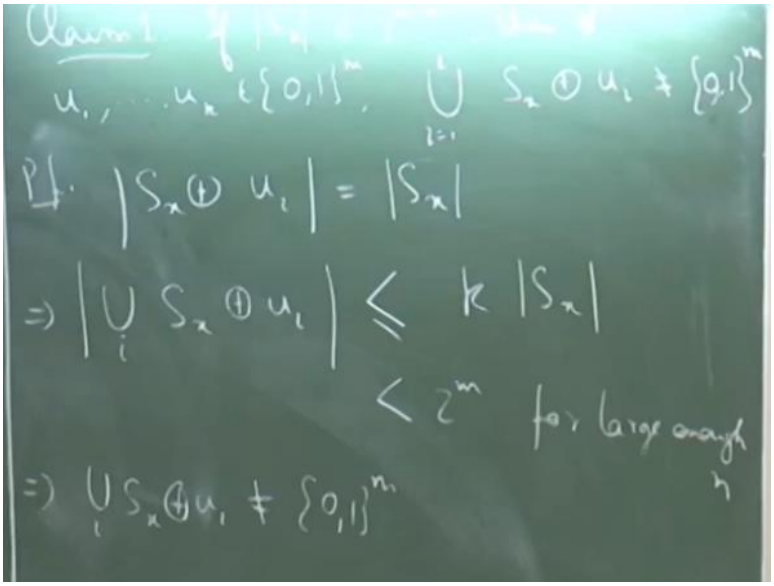
(Refer Slide Time: 08:18)



So given 2 strings let us say u, v in some universe 0, 1 to the power n let u + v denote the bit wise XOR of u and v. Suppose you have 2 m bit strings you can consider the bit wise one that is the bits are 0 or 1 you make it 0 or otherwise it is 1. And similarly given a sets and some u in 0, 1 to the power n I can extend this definition to the set +u as it has the natural definition it is the set of all w such that w is u XOR v for v belonging to S.

So you maybe I should have written this I mean it does not matter because this thing is commutative but just to be consistent I should write this as v XOR u. And this operation is commutative because it does not matter whether you look at u XOR v or v XOR u is the same thing.

(Refer Slide Time: 10:31)



So the first claim and let us fix a constant so it also defines this constant K should be (()) (10:53) of m by n+ 1. We will come to this later on why we have this particular definition but let us just define it this way for the time being. So the first claim is that if S of x is smaller than 2 to the power m – n then for all vectors u 1 up to u k belonging to 0, 1 to the power n. If I look at the union of all these sets S of x XOR with u i going from 1 to k. So this is not equal to the entire universal.

So in other words what this claim is saying is that? So this is my S of x suppose S of x is small then no matter and if I take k vectors no matter what k vectors I take and I consider the XOR of this set with each of these vectors. And I take union of all those elements that will never match up with entire universe. So there will be some element here which will not belong to that union. And the reason for that is that because this set is small so this proof is actually quite easy I mean it just follows from simple counting.

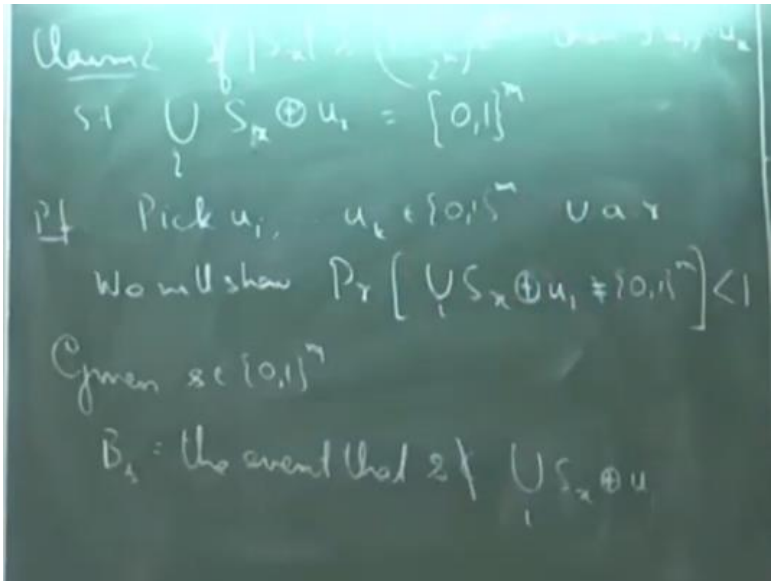
Because what is the cardinality of S of x XOR with any u i can it be strictly less than S of X it is actually equally it cannot be actually less than S of x also because look at XOR of any element here in fact with any other vector any 2 different vectors it will always give 2 different values. So in other words for the same vector it will not give for different vectors it will give 2 different values.

So this cardinality is equal to S of x what this implies is union of cardinality of union of S of x with u i is less than or equal to. Of course in this union can have common elements this is less than or equal to k times S of x and S of x is smaller than 2 to the power m – n. So this is strictly smaller than 2 to the power n for large enough m because k is just a m by n so it cannot cancel off 2 to the power n.

So which implies that union of S of x (()) (15:20) not equal to 0, 1 to, n any questions? So what is m? You can do that then there will be at least so then S of x is a small set which is fine. But then we are not done we have to look at the other side also so the claim 2 is that if you have such an S of x. And if you look at the XOR which these u i's for a large enough S of x it would basically fill up this entire space.

So if you take the error to be too small then you might have a problem it should not matter actually I mean you can have m = n. Because you can have a randomized machine where the number; of random choices is equal to the length of the input. So that can happen.

(Refer Slide Time: 17:09)

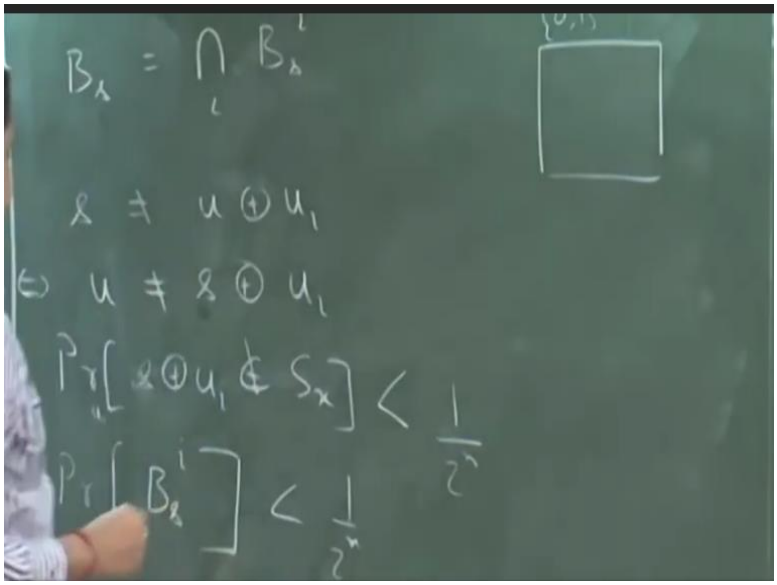


So the second claim is what I just that if S of x is bigger than 1 – 1 over 2 to the power n times 2 to the power n then their exist vectors you want to u k such that union of S of x with u i = 0, 1 to the power n. This is a slightly more complicated than the other one but these 2; game together will show the actual result. So how do we prove this so the proof is again a probabilistic so we will prove that the probability that suppose we randomly pick these vectors u 1 through u k.

Suppose these are picked randomly then the probability that the union of all these sets will not span the entire universe is non-zero in other words there exist some vectors u 1 through u k for which it will span the entire universe. So let me just write that down so repeat u 1 to u k which are m vectors uniformly at random. And then what we claim is we will show that the probability over all this vectors.

So the probability that we do not spend the entire space is not one in other words there exist a set of vectors for which it does that any questions? So far so let us define this event given a string S 0 1 to the power n we define the event b of s as equal to given that s does not belong to. So all we have to do is we have to calculate what is the probability of the union of all these events taken over all strings s.

(Refer Slide Time: 21:48)



We will define b s of 5 as the event that S does not belong to S of x XOR u i. So then basically this implies that B s is the intersection of all these events so the probability that it does not belong to the union is the probability that it does not belong to each and every set S x + u of i. So how do we compute this probability? So each B s i is an independent event exactly. So once we can compute this we will get this but how will you conclude this exactly so the point is that so what we want to check is that.

So S does not belong to so S is not equal to some u XOR u of i is the same as writing this as u is not equal to s XOR u of i because of way we have defined this operation. So now we have fixed n x and u i is chosen uniformly at random from this entire universe so the probability that the vector u is not equal to s XOR u of i. So the probability that vector u is not equal to s XOR u of i is the probability that u does not belong to this set.

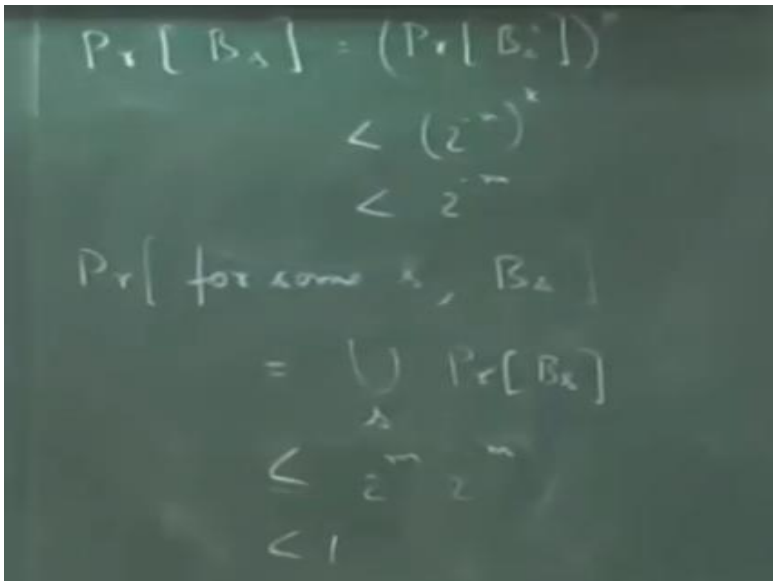
And that we know is smaller than u does not belong to this set so this is smaller than 2 to the power m – n. So the probability that B s of i happens is (()) (25:24) no so u is the vector in 0, 1 to the power. So that is not equal to vector of the form, s XOR u of i equal to should yes you can check the other one instead of (()) (26:47). I am just no this probability no but so this is for a fixed s and randomly chosen u of i.

Probability is over the choice of u of i so you have fixed x and a fixed u no but when I am talking of one particular event s n you are fixed as a, vary actually this is not the way. So I was getting confused because so this is actually does not make sense here because so what is s? If I am picking s from the set s of x then what we have written here is correct. But for any string s so the probability that this is not equal to this is just depends how we are picking this 2.

So these 2 are picked randomly then it will just be one over 2 to the power m or something like that the basically the size of the same. But actually what I want to claim here is that so the probability that s XOR u of I does not belong to s of x. So this is 1 over 2 to the power of n because all these elements so they are uniformly distributed in this entire universe. So, therefore the probability that any one element does not belong to this set is smaller than 1 over 2 to the power of n.

Therefore the probability that B s of i is true is s does not belong to s of x so it is less than 1 over 2 to the power n. So now we have a upper bound on this probability.

(Refer Slide Time: 30:53)

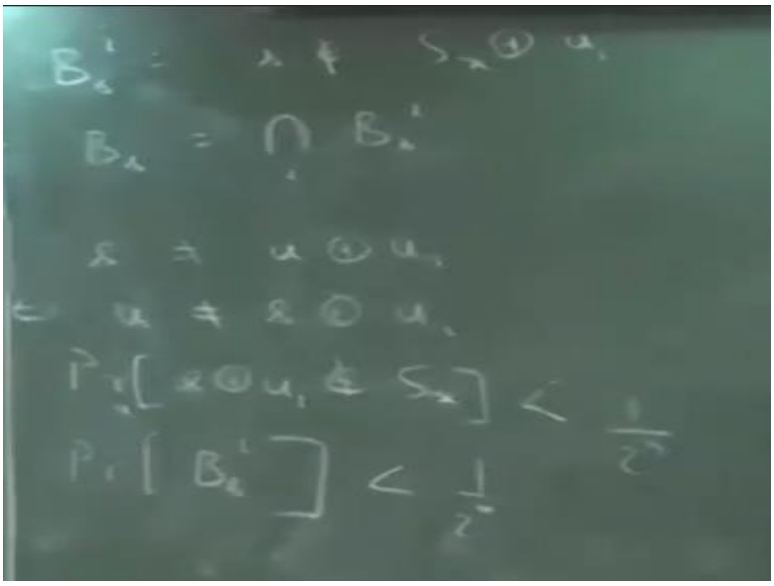


So now we can again bound the probability of b of s so what is the probability of this? If this is the probability of B s of i whole raised to k. Because all these events are independent event the probability that s does not belong to a certain s of x union XOR with u of i is independent with the even does not belong to s of x XOR with some other u j. So therefore this is less than 2 to the power minus of n whole raised to k.

And what is the definition of k that we had? Was m by n so therefore this is less than 2 to the power -m strictly less than 2 to the power –m. So now we can write this as probability that for some s that when b of s happens is less than or equal to the union of all these events. So union over all s probability of b of s and by the union bound we have this as exactly this is equal. But this is less than or equal to so how many s is can we have? 2 to the power n this is less than 2 to the power n times 2 to the power –m strictly less than.

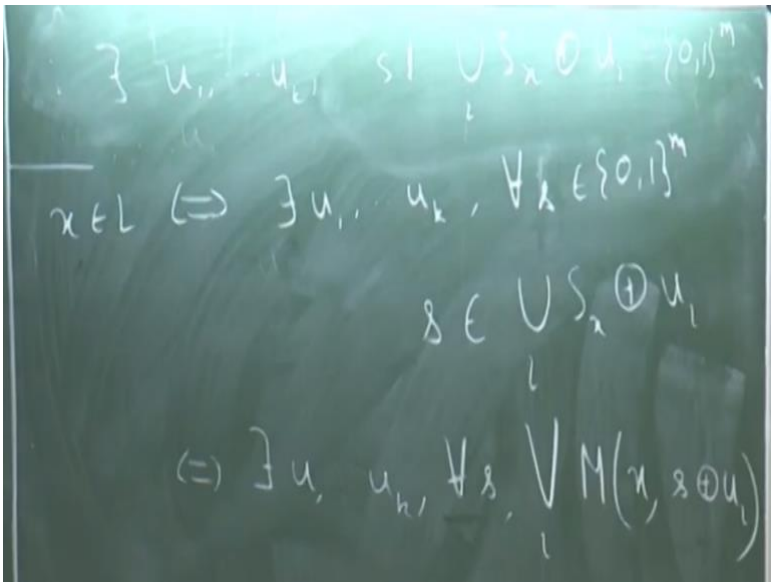
And this is less than 1 so are we done here? So the probability that for some s does not belong to this union so we are showing not to be less than 1 and that is what we actually wanted here that is the probability that there is some s in this set it does not belong to the union.

(Refer Slide Time: 33:50)



So therefore there exist some set of vectors you want to u k except the union of s of x and u of i is the entire universal. So the crucial thing here is that yeah got, myself a little bit confused here. But the property that this thing is commutative this operation the way it is defined. So the we can write that x does not belong to u XOR or u of i or s not equal to u XOR u of i or s not equal to u XOR u of i. So this is equivalent to saying that u is not equal to s XOR u of i so now let us complete the proof.

(Refer Slide Time: 35:08)



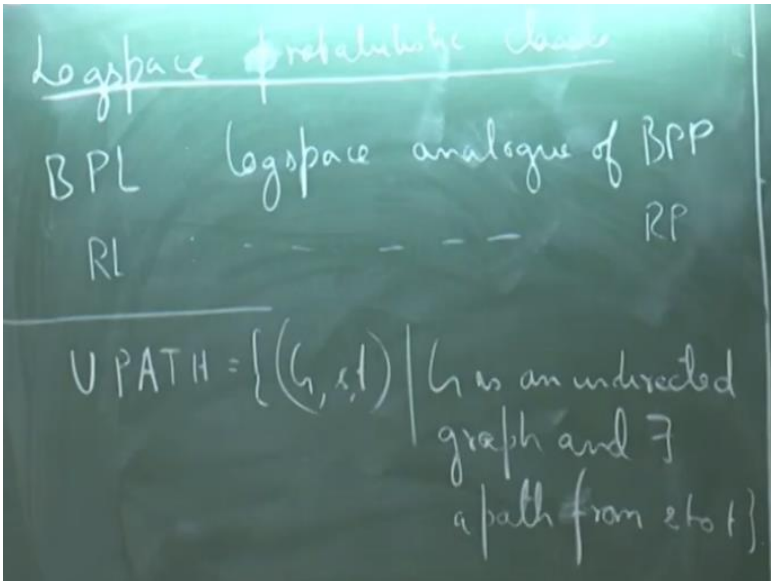
So what do we know that x is in l if and only if this set as a large size so when does the set have a large size there exists u 1 to u k such that for all strings r. Or let me just put it as s since I am using here of all strings s belongs to the union of s of x XOR u of i. So this is what we have and

now the thing is that we have to convert it this into a sigma 2 p definition. So this can equivalently be written as there exists vector c 1 through u k such that for all s.

If I look at this machine m that is given the string x and another string which is basically the random string that it simulates s XOR u of i. This machine should accept and we just do not make one run of this machine we make several runs of this machine. And if just in one of those runs the machine accepts then we accept. Because let us just pause here and see what this statement is claiming so this is claiming that for all strings x all strings s in the universe.

If this machine is provided with s and some u of i then it should accept x and that is what we know from our initial definition that if x belongs to the language then there is some u of i for which all strings s we will belong to this u m. Then length of the random string that is true. So let us look at couple of more things so similar to b p we can also define the log space analogs of these class this class is BPP and RPP.

(Refer Slide Time: 38:34)

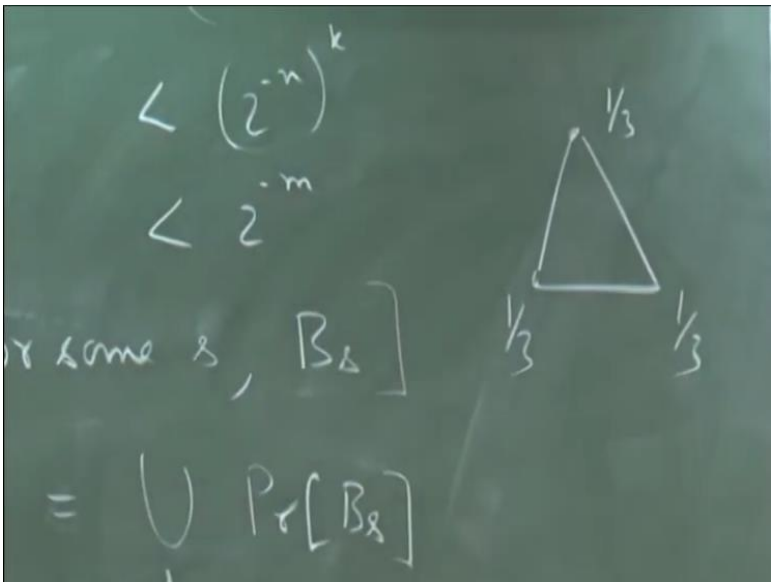


So log space probabilistic classes so we have the class BPL so I would not go ahead and formally defined it. But this is just a log space analog of BPP so there is a probabilistic machine which uses login amount of space and it makes an error of at most 2 by m. And we also have that class RL which is the log space analog of RT. So there is one very interesting problem which is contained in this class RL and that is the problem of undirected reachability in graph.

So recall the path problem that we defined the so similarly we can define the problem u path. So this is given in graph g and 2 vertices s and t where g is an undirected graphs and their exist path from s to t so what did you say? Because this was the first non-trivial result that was shown about the undirected path problem but then there are subsequently many improvement over it. And only in 2005 it was shown that this is in it.

So we just quickly mention some of those things but it is very important to understand this RL algorithm.

(Refer Slide Time: 41:32)



So this was shown in 79 I like 79 or 78 that few path is in RL and this algorithm is quite simple. So what this algorithm says is that u yes it is the NL of course because you can always guess a path and take the correct answer. But it has been shown that it is in l as well so there is the whole sequence of results. So first it was shown in RL in 79 then I think in late eighties it was shown that this problem is in l to the power 3 by 2.

So what that means is? D space log to the power 3 by 2 n then I think in 2000 or 99 sometimes in the late nineties. It was shown that this problem is in D space log to the power 4 third of n and then finally it was shown by Rhine Gold in 2005 that this is in l that kind of catches the complexity. This are the very important line of work because the, I mean as you all know that the direct version of this problem is for hard for n.

So any improvement on directed version would immediately give a collapse on class NL. So for the directed version for example we do not even know whether it is in RL or whether it is in 3 by 2. The best we know is by Savage's theorem which is log square because again if you can show it is in that RL that would immediately imply that NL is contained in RN which is not known which one?

I mean all these are the randomization results actually all these results kind of look at this and is some kind of de-randomization. But this is more complicated and it uses this theory of expanded graphs. So, expanded graphs are undirected graphs which have very low degree but very high connectivity. So when I say high connectivity it means that the, diameter of these graphs are very small. So from one vertex in the graph you can go to any other vertex let us say log in number of steps.

But these are certain things that you expect only for graphs which are very high degree so expanders are graphs which simultaneously have both these properties. So what Rhine Gold did was he took an arbitrary undirected graph and then he converted into an expander. Because once you have an expander once you have a graph which as diameter of let us say log n you can brute force DFS or something and you can solve it in l.

Because suppose the graph as constant degree let us say it has degree 2 and it has diameter log in. Then you can always keep a log in bit vector which will help you to do a DFS and whenever you are going left store 1 and whenever you are going you store 0. And you can explore all possible parts from S to any vertex in that component. But the crucial thing about this thing is that how do you convert it general undirected graph into an expander.

So that was the heart of his book so what I was mentioning here is that the first result in this line of work was this thing which showed that undirected path is in RL and the algorithm is quite simple what the algorithm does is? It starts at s and it just performs a random work of some polynomial length I think it does a random of some 8 times 10 to the power 4 where n is the number of vertices. And it claims that if t is reachable from s.

So in a undirected graph reachability is the same as checking in 2 vertices belong to the same component. So if s and t belong to the same component then what they argued is that with

probability greater than half you will reach t and of course if t does not belong if s and t do not belong to the same component no matter how long you are you will never reach. So the other direction is quite easy.

So one direction is a little bit non trivial but it is also not that complicated so what I will do is I will not proves this result in class but I will just post a link to a proof of this theorem. So those of you who are interested you can just go back and read it. So it is not very difficult it is about 1 or 2 page proof but it is a nice argument. So I will so it uses the undirected property because it checks for the fact that 2 vertices are same component.

Because in directed graphs there can be 2 vertices which are in the same component but they might not be path between them. But here they are using that so what they showed here is that I mean again let me get the idea is that if you have a stationery distribution on your graph the stationary distribution is again a. If want informally define it is basically assigning probabilities to every vertex of your graph such that if you take one step from any vertex.

The next probability distribution that you get is the same as your original distribution. Just to give a quick example suppose you have a triangle. So this is; the graph and let us say you are with equal probability of one third likely to be in any of the 3 vertices. And then suppose if you take a step from a vertex so let us say this is a, b and c so suppose if you go for vertex a to b.

So let us say the probability of going from a, to b is half the probability of going from a, to c is also half. So what is the probability that you were at k and in the next step you go to b 1 by 6. The other way also that you can come to b is from c what is the probability that you were at c in your previous step and you come to b. That is also 1 by 6 the probability that after 1, 6 you are at a vertex 3 is again 1.

So in this graph no matter how long a walk you take if you start with this distribution you will always remain at the same distribution. So this property is not true for directed graphs. In directed graph this notion of stationary distribution is not guaranteed. In an undirected graph you can show that any undirected graph you there is always a stationary distribution for that graph. But that is a crucial difference but anyway I mean I do not want to go into the details of that just post that link and (()) (49:52).