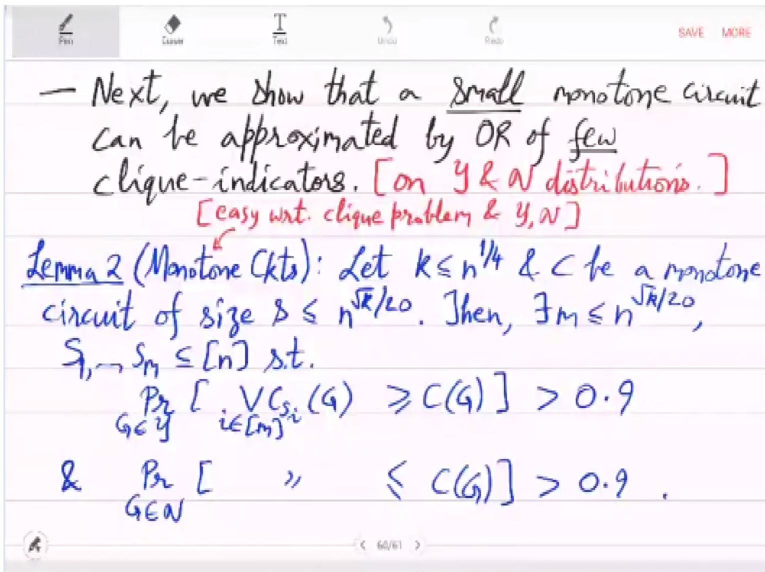
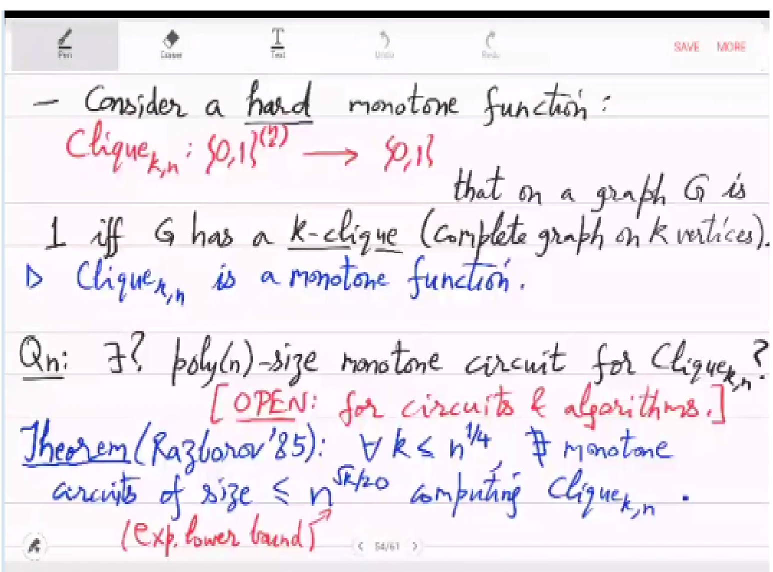
Randomized Methods in Complexity
Prof. Nitin Saxena
Department of Computer Science and Engineering
Indian Institute of Technology - Kanpur

Lecture - 08
Monotone Circuit Lower Bound and Sunflower Lemma

(Refer Slide Time: 00:13)

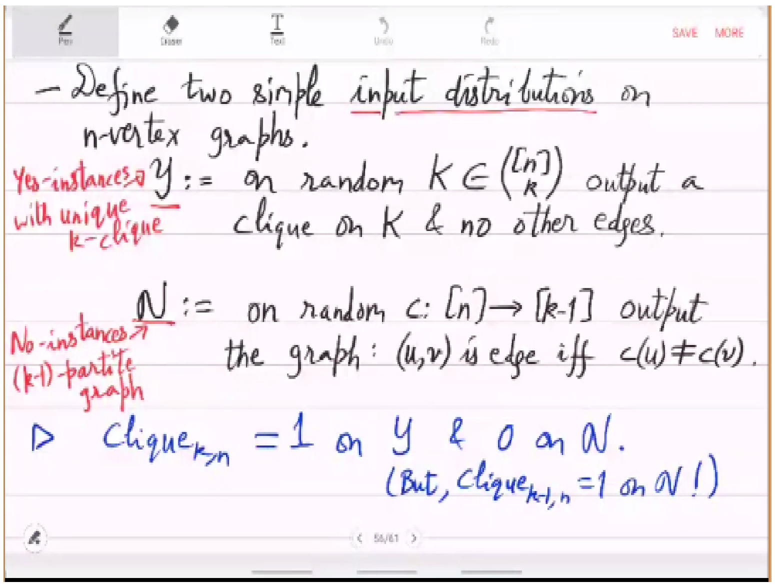


(Refer Slide Time: 00:16)



Yesterday, we saw 2 lemmas. So, the theorem that we are proving is for monotone circuits. So, this theorem in blue, you want to show Razborov's theorem that the k-clique Boolean function on n vertex graphs. If any monotone circuit solves it, then the size has to be $n^{\sqrt{k}}$. So, if you think of k to be, let us say, n/10, then this $n^{\sqrt{k}}$is like $n^{\sqrt{n}}$. So, this is a very large exponential sized monotone circuit. So, to prove this theorem we have to work hard. So, we have to prove 2 lemmas.

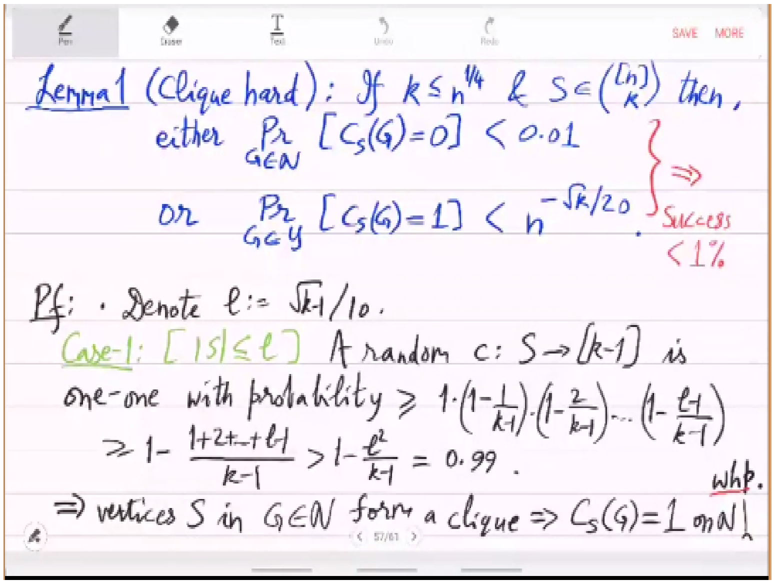
(Refer Slide Time: 00:58)



The first lemma that we showed, so, recall these input distributions. So, we will be working with these kind of graphs, first is the yes distribution. The yes graphs why? Which is basically just randomly pick subset of size k and the set is called K and on this draw clique, join every possible pair of vertices. The no graph is basically these, which have k - 1 partite connections. So, you basically just label randomly the vertices, labels 1 to k - 1.

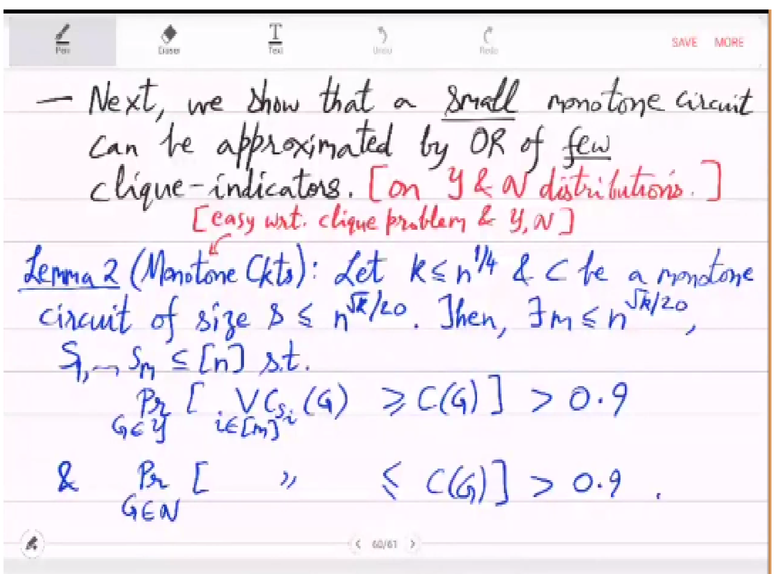
So, there will be k - 1 cliques but not k cliques. These are the clear no instances. So, only on these special yes and no graphs, we will work. So, lemma 1 is that which we have already shown that clique is a hard function on this distribution assuming that you use clique indicators.

(Refer Slide Time: 02:04)



So, if you are only going to do an OR of clique indicators, then the success probability for a single clique indicator is so low, it is less than 1 percent that you will actually need larger than $n^{\sqrt{k}}$ these many clique indicators. In few clique indicators, you cannot achieve success of 90%. That was the first lemma.

(Refer Slide Time: 02:38)

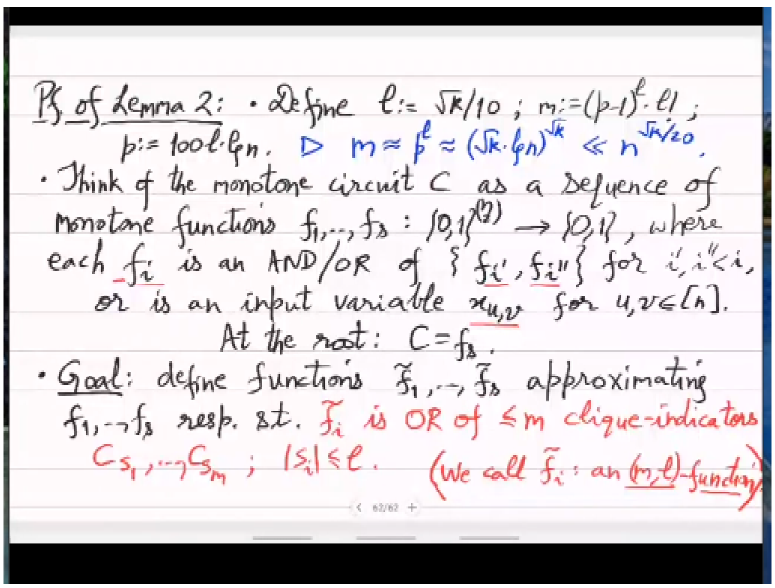


Second lemma that we have stated again in this blue is that if you look at the output of a monotone circuit solving clique on these yes and no distributions, input distributions and the size is smaller than $n^{\sqrt{k}}$, then actually few clique indicators is enough to achieve 90% success. So, these 2 lemmas taken together give you the proof of Razborov's theorem. This also we finished. So, the only thing remaining is lemma 2.

So, we will prove this now. We will show that a monotone circuit has few clique indicators. So, the size of the monotone circuit is S $\leq n^{\sqrt{k}}$ and the number of clique indicators is $m \leq n^{\sqrt{k}/20}$ some constant is there and we want to show that the success probability is 90% on the yes and no distributions. So, remember that we are only looking at special graphs.

So, let us prove this. This is going to be a very tricky proof with interesting combinatorial techniques.

(Refer Slide Time: 03:58)



So, like we had clique indicator size before $l := \sqrt{k}/10$ and the number of clique indicators that we will ultimately get that will be m:=(p-1)$l.l$! where=100l.logn. So, p is like l, p is like $\sqrt{k}$, so, your m is $\sqrt{k}$ kind of that. So, m≈pl ≈($\sqrt{k}$.logn)$\sqrt{k}$ So, this thing in all is, we have taken k to be $n^{1/4}$.

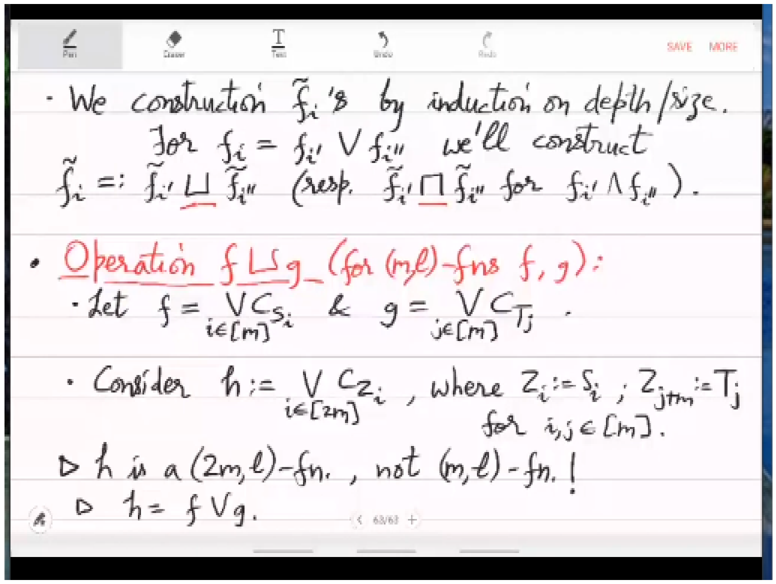
So, overall this is much smaller than n$\sqrt{k}/20$. So, this is the m that hereafter. We will reduce to these many clique indicators, such that the OR of these clique indicators is correlated well with the monotone circuit output on the y and n distributions. So, think of the monotone circuit as a sequence of monotone functions $f_1\ldots f_s$: input is a graph, so, input $\frac{n}{2}$.

Output is a bit, where each f_i, so, monotone circuit is computing, is first it applies some AND/OR gates on the input, which is basically your sequence of edges, $\frac{n}{2}$ many bits and on this output, another layer of AND/OR gates acts and so on. This is how you can see the computation in the monotone circuit happening, leaves to the root. So, the functions which you are getting these are all monotone functions and this is what $f_1\ldots f_s$represent.

So, each f_i is an AND/OR of 2 previous functions, $f_{i'}$ and $f_{i''}$ or is an input variable $x_{u,v}$. So,f_i is, we are assuming this to be, all functions to be binary, AND OR are our binary gates. So,f_i will have 2 inputs. They are $f_{i'}$, $f_{i''}$or it is a variable.f_i is a variable at the leaf. That is one way to model the monotone circuit and then finally or at the root, what happens is, you get circuit equal to the circuit value C = f_s that is the Boolean function in the end.

So, our goal is to, express f_i as clique indicators or of clique indicators, define functions $\overline{f_i}\ldots\overline{f_s}$, such that $\overline{f_i}$ is an OR of ≤m clique indicators. Write m clique indicators, $C_{s_1},\ldots,C_{s_m}$; $|s_i| \leq l$. So, m clique indicators each of size l and we call this function and (m,l)function. So, each intermediate function f_i will represent it as an (m,l) function, which means that we will actually devise m clique indicators each of size l . So, that the OR is well correlated with f_i value.

(Refer Slide Time: 11:24)

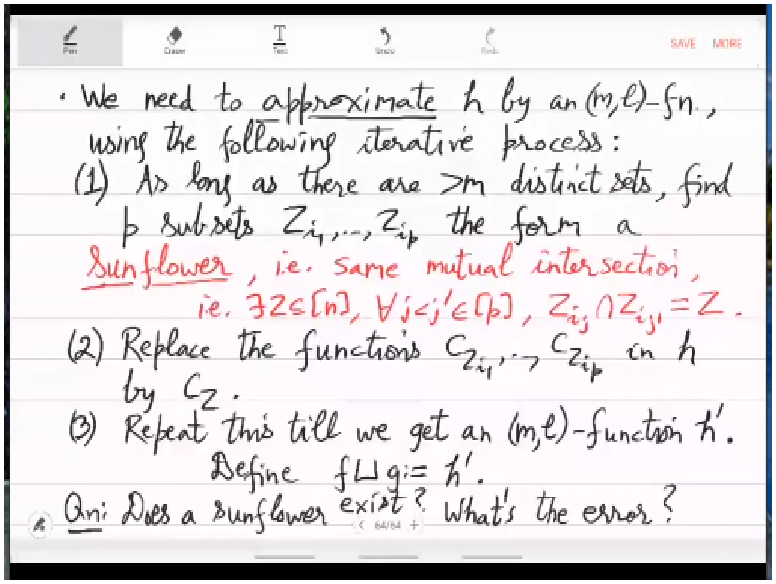


So, we construct a $\overline{f}_i$ by induction, again induction on depth and size. So, for the OR gate in the monotone circuit, we will construct $\overline{f}_i := \overline{f}_i' \sqcup \overline{f}_i''$, some operation this square cup operation on the 2 input (m,l)functions. So, $\overline{f}_i'$ is an (m,l) function and same thing with $\overline{f}_i''$, using those two (m,l) functions square cap operation, will define. This (m,l)function for $\overline{f}_i$ and respectively, $\overline{f}_i' \sqcap \overline{f}_i''$for $f_i' \wedge f_i''$.

So, these are the 2 operations that we have to now define. So, what is this operation? So, both f and g are given to you as(m,l) functions. Let us also write that from this these (2m,l) functions, we want to produce a single one. So, say f is these clique indicators $f = \vee_{i\epsilon[m]} C_{s_i}$ and $g = \vee_{j\epsilon[m]} C_{T_j}$, and remember that you want to compute OR of f and g. So, what should you do?

You should just takeS_i and T_j's together, but then you will get 2 m clique indicators, $h = \vee_{i\epsilon[2m]} C_{Z_i}$ where $Z_i = S_i$ and $Z_{j+m} = T_j$; for all i, j$\epsilon[m]$. So, we are just placing $S_1 \ldots S_m$and then $T_1 \ldots T_m$ calling them Z_i in order $Z_1 \ldots Z_{2m}$. And this h is clearly an OR of f and g, but it is a (2m,l) function. So, we have to drop or rewrite the clique indicators to make it m comma 1 form (2m,l), but this h is equal to f OR g that is correct, but it is not (m,l). So, let us now do a procedure to reduce the clique indicators.

(Refer Slide Time: 16:20)



So, we need to approximate h by an (m,ℓ) function. So, what we will do is using the following iterations. So, how do we approximate ideally? We want to drop some of the clique indicators. So, what we can do is, we can actually merge clique indicators. So, for example, if Z_1 and Z_3 clique indicators, they have something in common then we replace the 2 by their intersection.

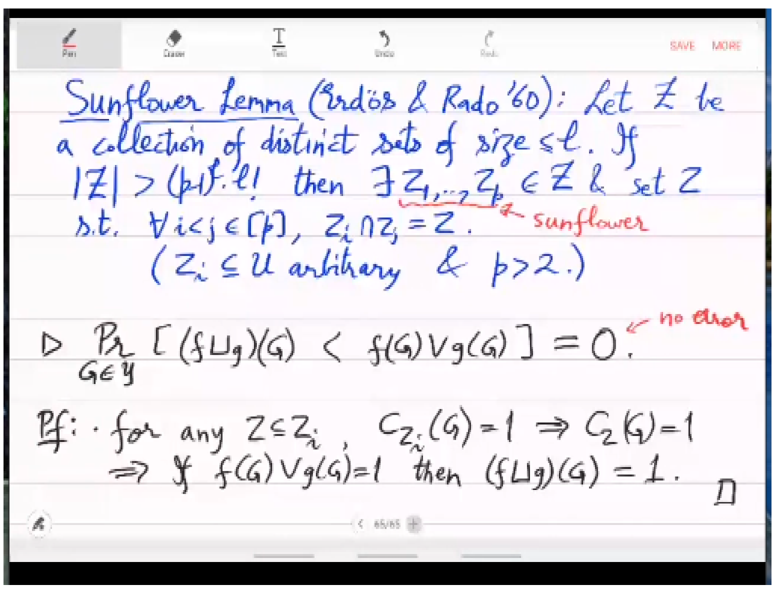
So, in set theoretic terms instead of checking, whether subset Z_1 is a clique or Z_2 is a clique, you check whether $Z_1 \cap Z_2$ is a clique. So, you may be making mistake, but at least you have reduced the number of clique indicators. So, that is what we want to do. So, we will do it systematically. So, as long as, there are >m distinct sets, find p subsets. So, what was p? p was $\sqrt{k}\ logn$.

So, find p subsets, $Z_{i_1} \ldots\ldots Z_{i_p}$ that form a sunflower. So, what is a sunflower? So, in this set theoretic or combinatorial terms, sunflower means that mutual intersection of $Z_{i_1} \ldots\ldots Z_{i_p}$ is the same. That is, there exists a subset of $Z \subseteq [n]$, $\forall\, j < j' \epsilon [p]$, $Z_{i_j} \ \cap\ Z_{i_{j'}}$. So, find p subsets that form a sunflower, which means their mutual intersection is the same and then replace them by this intersection.

So, that will shrink your clique indicators that is the trick. So, replace $C_{Z_{i_1}}, \ldots, C_{Z_{i_p}}$ clique indicators in h by C_Z. And third is, maybe still, you have more than m clique indicators, because originally you had 2 m, m is much bigger than p. So, in 1-round, you are only reducing by p, – p, so, you might have to repeat this. So, keep repeating, till you go below m or till you reach m. So, when you stop, then you call each prime to be $f \sqcup g := h'$.

So, there are these rounds of identifying in each round, you are identifying a sunflower and then replacing the p clique indicators with the intersection. So, the question is, will a sunflower always exist? That is the first question. If we keep finding sunflowers and we keep reducing this way, what is the error that we are introducing? So, these are the two things, we have to understand now, the process is clear. This is the process, but why does it proceed? And second is, what is the result? How good is the result?

(Refer Slide Time: 22:56)



So, first question, we will answer by stating sunflower lemma. So, Erdos and Rado prove this. So, let this $\mathcal{Z}$ be a collection of distinct sets of size $\leq l$. So, like **(( )) (23:58)** 2 m sets each of size l that is the set of subsets is called fancy Z. Now, if it is a lot of subsets, so, if $|Z| > ((p-1)^l \cdot l!$, then there exists a sunflower basically, then there exists $Z_1 \ldots\ldots Z_p \in \mathcal{Z}$ basically.

And a set Z such that $\forall$ i<j ϵ[p], $Z_i \cap Z_j = Z$ when you take the intersection, it is exactly Z, so, that is a sunflower. So, what this is saying is that if your collection is large, relative to the set size, which is relative to the subset sizes, which is l. So, if your collection is exponential in that, then there is always a suitably large sunflower in the collection. So, in this the universe is arbitrary.

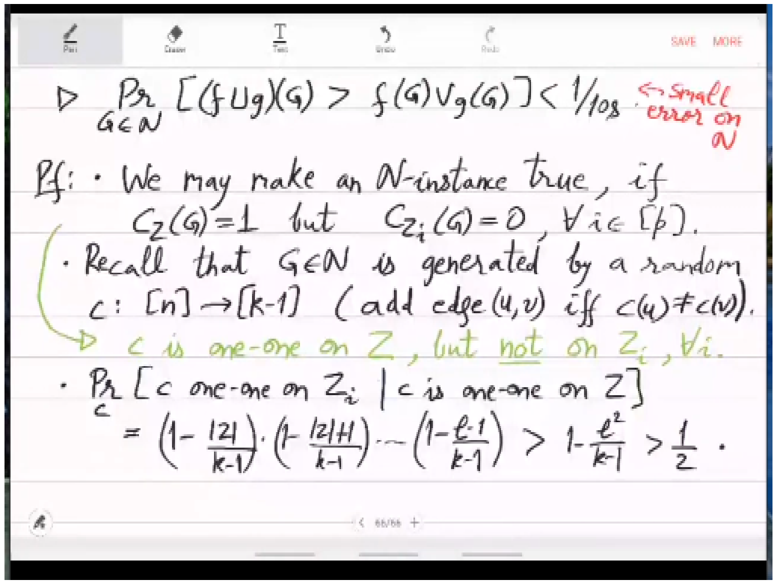
So, Z_i are subsets of an arbitrary universe, there is no size information there and p >2. This is also what we are assuming. So, at least you will get in the sunflower at least 3 subsets Z_1, Z_2, Z_3 such as the mutual, there are 3 mutual intersections, they are all equal. So, this we will prove later, the proof is again by induction, it is by induction on the collection size. Assuming this, let us continue with the error analysis.

So, first we will look at the yes instances and show that $Pr[(f \sqcup g)(G) < f(G) \vee g(G)] = 0$. This is what we will show. This is kind of obvious because, in this h which was the all the clique indicators, 2 m clique indicators, we are just replacing some of these clique indicators by the intersection.

So, if the original value was 1, then after the removal or after the replacement also it remains 1. So, 1 cannot become 0. So, this inequality can never hold. So, the reason is $C_{Z_1}(G) = 1 \Rightarrow C_2(G) = 1$. Because Z_i , you have already checked that on Z_i there is a clique in the graph g. So, on Z_i there is a clique, there is also a clique, obviously there is a clique on a subset.

So, if OR was 1 then in this approximation, the value will be 1. There is no error introduced. Second is on the no instances.

(Refer Slide Time: 28:55)



So, probability on the no instances, what may happen on the no instances is that there was no clique but since now, the clique indicator is smaller, it says that there is a clique. So, what is the probability then? $Pr\,[(f \sqcup g)(G) \ > f(G) \vee g(G)] \ < 1/10S$The probability is 1 / 10 s that is what we will show. So, let us analyze this. This will be a probability analysis. Remember no instances are k - 1 partite labeling. So, a random labeling is chosen.

So, we will use that to estimate the error. We may make an no instance true if $C_Z(G)=1$ but $C_{Z_i}(G)=0$, $\forall$ $i\epsilon[p]$. So, originally the OR of clique indicators was 0 but when you replace it by the intersection it is 1. It is a smaller clique indicator. So, it is making a mistake. So, recall that G is generated by a random vertex labeling into k - 1 parts and what are the edges? So, add edge if and only if different labels.

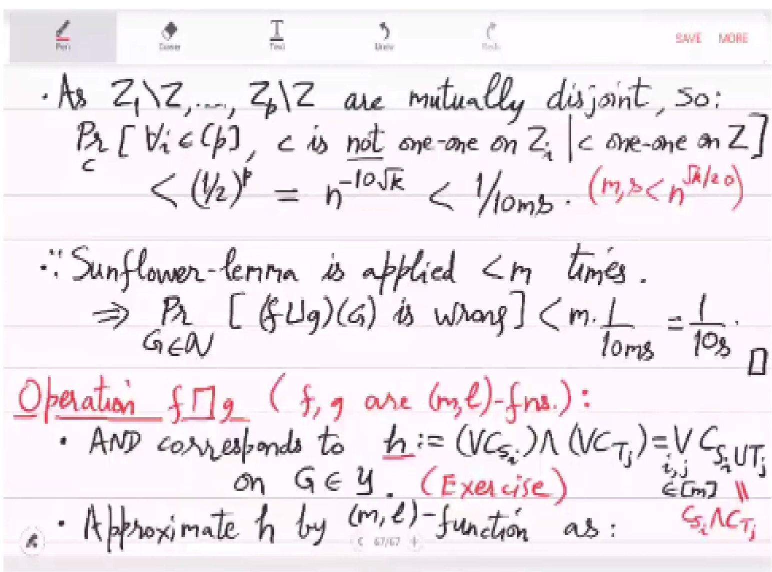
So, what is happening is that C actually was not one to one on Z_i but it is one to one on Z, on Z_i's. This is what has happened. So, the random C that you labeling that you picked, it may it was actually not one on one to one on Z_i but the places where it was not one to one, they are lost because you replace them by the smaller Z. So, what is the probability? Let us say, what is the probability C is one to one

on Z_i given C is one to one on Z.

Let us instead calculate this probability. So, if you have checked that C is one to one on this smaller subset Z, what is the chance? That it is one to one on bigger subset Z_i that is containing Z. So, that is basically you just have to make choices for $Z_i - Z$. So, this probability is equal to, $Pr\ [c\ one\ -\ one\ on\ Z_i\,|c\ is\ one\ -\ one\ one\ Z] = (1 - \frac{|Z|}{k-1})\ \ (1 - \frac{|Z|+1}{k-1})\ldots\ldots.(1 - \frac{l-1}{k-1})$so, remove the Z values which you have picked out of k - 1 and then Z + 1 and finally Z_i is at most l large if you assume exactly$l$ this is removing $l - 1$.

And this fraction, we have seen before, so, this comes out to be at least$1 - \frac{l^2}{k-1}$, which is more than ½ in this setting, $\frac{l^2}{k-1}$ was small. So, this is at least 50 percent. So, once you have checked on Z, then on Z_i, the result will be the same usually. So, C is most probably one to one on Z_i. Now, but there were Z_1, Z_2 and Z_p, there were p things chosen. So, what is the chance that C is not one to one on any of those.

(Refer Slide Time: 35:00)



So, $Z_1\backslash Z\ldots\ldots\ Z_p\ \backslash Z$ they are disjoint because of the sunflower property. They are mutually disjoint. So, we also get pi independence. So, this looking at C, across Z_i's is actually independent. These are independent p events, so, we can multiply the probability. So, now $Pr_C[\forall\ i\epsilon[p], c\ is\ not\ one\ -\ one\ on\ Z_i|c\ one\ -\ one\ on\ Z]$. So, previously, we showed that this probability for Z_i is less than 1/2.

So, when you go over all these Z_i's, it is less than $1/2^p$, 2^p is the way we chose it, this was a $10\sqrt{k}$.logn. So, you exactly get $n^{-10\sqrt{k}}$. So, if you look at m, what was m? m and S both of them we had picked $n^{\sqrt{k}/20}$. So, we can claim that this is smaller than 1/10ms. This is much smaller. So, that is the probability that you make a mistake on when you replace the sunflower by a single clique indicator.

And sunflower lemma will be applied. So, this replacement of a sunflower will happen how many times. You are going throughout. You are covering every gate in the monotone or you are covering in this case all the OR gates in the monotone circuit, which are at most as many you wanted to reduce 2 m to m. So, every time you reduce a sunflower, you are reducing the number of clique indicators by at least 1 obviously.

And you want to reduce them maximum by m. So, this is done m times at most. So, sunflower lemma is applied less than m times, which means that the probability over the no instance, when you are doing this single square cup definition. This is less than m. 1 / 10 ms just by the union bound, which is 1 /10 s that is what we wanted to show. So, the error that we introduce is at most 1 / 10 s per OR gate but over 10 s is good because there will be at most S OR gates.

So, by union bound again, the error introduced overall is just 1 by 10, just 10 percent error 90 percent success. Let us now look at the second construction which is operation f ⊓ g for AND. When you see an AND gate, what do you do? So, AND corresponds to,

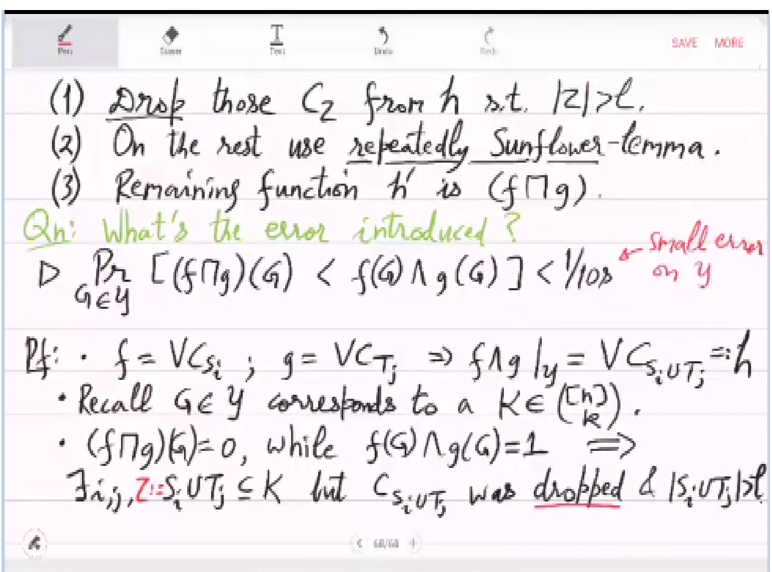
$h := (\vee\, CS_i) \wedge (\vee\, CT_j) = \vee\, C_{S_i \cup T_j}$ So, you distribute this AND on OR's and then you want both S_i indicator and T_j indicator to be true, which means that S_i is a clique; T_j is a clique.

Now, will that mean that union is a clique. So, actually on the yes and no instances. It will yes and no instances. So, if you on the yes instance basically the only case when S_i and T_j both are cliques, is when both of them are in the subset k. So, the union will be at clique. In the no instance, if S_i is a clique and T_j is a clique then again, so, this has to be seen. Let me just say, yes at least this much is true.

So, let us leave this as an exercise. AND will actually correspond to this function h. So, we will focus on this r of $S_i \cup T_j$ for all i, j. This is our h in the AND case and $f \sqcap g$. So, we have these m square. We have these m square indicators. We want to bring it down to m again using the sunflower trick. So, approximate each by (m, l) function as so, what do you do?

So, you had this $C_{S_i \cup T_j}$ many m square, many indicators, so, again, we want to drop some of them in every round we want to drop them. So, that they become m or less. So, we will again be looking for a sunflower and removing it but some of them may actually have now length also size also more than l because S_i was at most l; T j was at most l; maybe the union is more than l. Those we will just drop more. So, between l to l will drop and l and below, we use sunflower lemma.

(Refer Slide Time: 44:07)



On the rest, you use the sunflower lemma. And then what remains is called h'. h' this is the definition of f. This is what we call . So, why will this process proceed? Well. That is because you will keep getting sunflower lemma as you got in the first case square cup case. So, square cap case also will have the same reason why you will keep getting sunflowers till you reach m clique indicators.

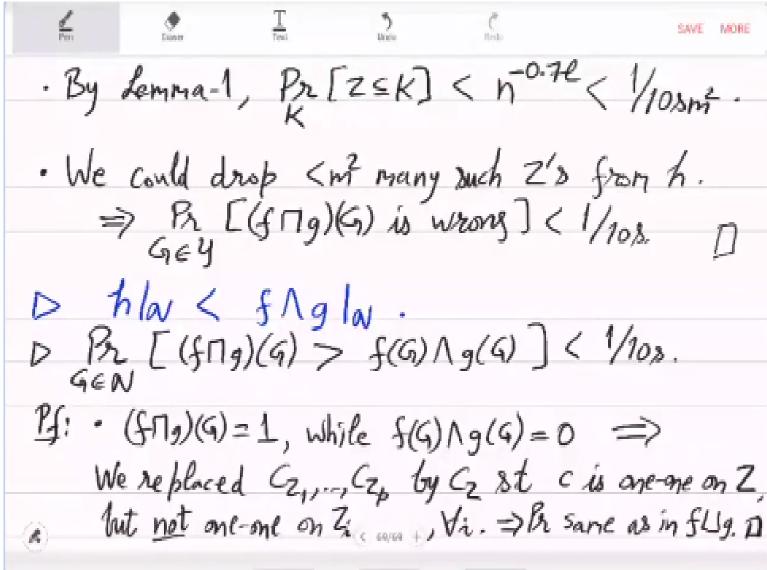
So, only question here is how much is the error introduced. So, this we have to work out. So, what is the error introduced in this procedure when you want to get AND of f and g. So, let us first see the, again the yes instance, no instance. So, first is on yes instances, $Pr[(f \sqcap g)(G) < f(G) \wedge g(G)$. It can be smaller, but we will show that the probability is also small$Pr[(f \sqcap g)(G) < f(G) \wedge g(G)$<1/10s. So, small error on.

So, $f \vee C_{S_i}$, $g = \vee C_{T_j} \Rightarrow f \wedge g|_y = \vee C_{S_i \cup T_i} = h$and recall that $G \epsilon y$ yes distribution graph corresponds to subset K $\epsilon \frac{[n]}{k}$ on which there is a clique and there is no other edge in the graph. So, if this (f ⊓ g)(G)vanishes, then what does it mean? If this vanishes while the $f(G) \wedge g(G)$= 1, then what does it mean?

So, AND was 1 means that there was a clique on some $S_i \cup T_j$. So, $S_i \cup T_j$ was in contained in k but that was dropped. So, that clique indicator was dropped. This is the only thing that can happen. If it was not dropped, then there was no chance of getting this f ⊓ g to be 0. So, there existed i, j such that $S_i \cup T_j \subseteq$ K but $C_{S_i \cup T_i}$this was dropped.

So, the size $|S_i \cup T_i|$was large. It was bigger than$l$ which is why it was dropped in the process in step 1 that is the bad case here, which we have to analyze. So, let us call this Z and compute the probability.

(Refer Slide Time: 50:31)



So, by lemma 1, the probability over random K says that this $Z \subseteq K$. So, remember Z is larger than l. And k was, so, what is the chance that you pick a random key and it contains this large subset. So, that probability will be small $Pr[Z \subseteq K] < n^{-0.7l}$, which is smaller than if you recall S n m expression, it is smaller than their product in fact, smaller than $1/10ms^2$.

Just recall$l$ S and m, then you can see this. So, the process how long will the process continue; how many steps; how many rounds will the process have for 1 square cup or square cap that will be less than m^2because you had ing at most m^2. So, we could drop less than m square many Z's from h, which means that by union bound$Pr[(f \sqcap g)(G)\ is\ wrong]$. This is smaller than 1/10 s.

So, on the yes instances, the probability of or the error we have introduced is less that is what we have shown. Let us now go to the no instances. So, first observe that on the no instances, h is less than f and g. So, this $S_i \cup T_j$ indicators their OR is indeed equal to the AND of f and g; f is for S_i's and g is r of T_j's. So, on the no instances also, they match. So, this you can see by the way no instances were defined. They were defined by this k - 1 partite method.

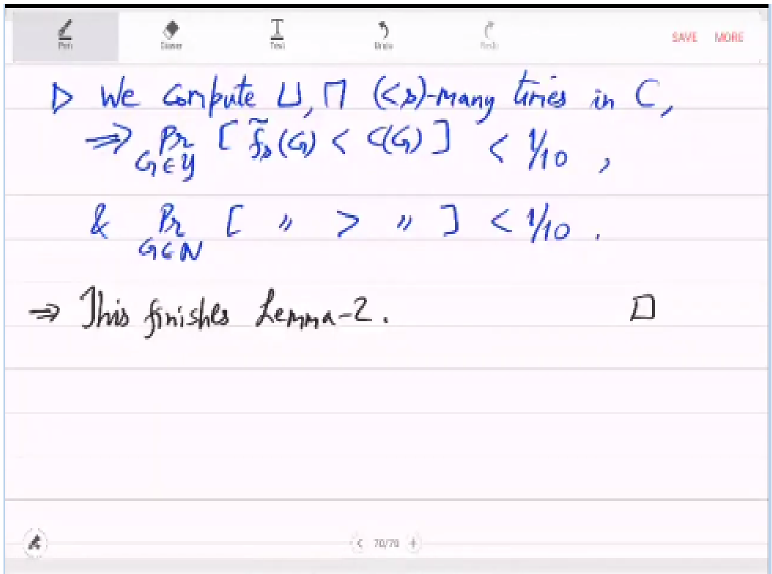
So, f and g being, I mean, you have to look at 2 cases 1 or 0. If it is 1, then it means that there is an S_i and there is a T_j, which are both k - 1 cliques. If f and g is 0 on some in no instance if it is 0, then basically there is no S_i and no T_j simultaneously cliques. So, there is no chance of their union being a clique. So, we only have to look at the other case, which is probability on the no instances.

So, we have to look at this square cap g. Can it be greater than? So, what is the chance that when we are doing this process to get square cap f square cap g in that process either we drop a clique indicator or we drop a sunflower and because of that a false value becomes true of the OR. So, what is the probability of that happening? So, this will again be by the sunflower analysis like we did in case 1 in the square cup case. It is 1/10 s.

If $(f \sqcap g)(G)$= 1, while the AND was not, $f(G) \wedge g(G)$= 0. So, how did this error happen? So, we do 2 things. In step 1, we drop a clique indicator and in step 2, we drop a sunflower. We replace a sunflower by some clique other clique indicator by the intersection basically. So, the first step cannot turn 0 to 1 because first step is just dropping. So, if it was 0 before, then it will remain 0.

However, in the second step, when you are replacing a sunflower by something some smaller clique indicator that may become 1. So, we replaced $C_{Z_1} \ldots\ldots C_{Z_P}$ by C_Z such that this labeling is one to one on Z but not one to one on Z_i for all the i's. So, then this probability follows by what we did for $f \sqcap g$. It is the same probability analysis like this we have analyzed before and that will give you 1/10 s. So, we have done both the operations square cup and square cap.

(Refer Slide Time: 58:17)



So, what we learn combining the two is that we compute square cup and square cap less than S many times in the circuit C. So, this means that the probability on yes instances of this final (m, *l*) function f S tilde, this being wrong and on the no instances, this being wrong. Both these probabilities are smaller than one tenth. Because for single operation we had shown 1/10 s.

So, by union bound on S many, you get 1/10. So, this finishes lemma 2. So, this means that we have rewritten monotone circuit at least for yes and no instances in these input distributions as an OR of few clique indicators. So, the only thing remaining is proof of the sunflower lemma that will finish next time.