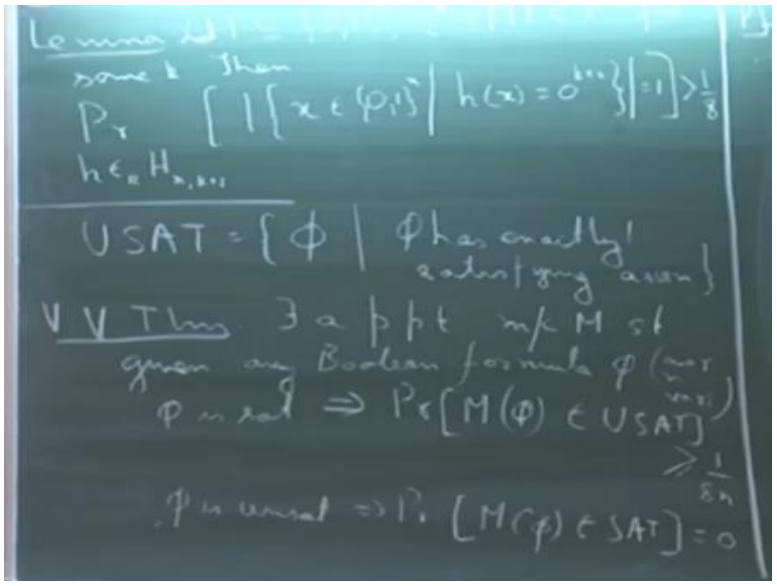
Computational Complexity Theory

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

Module No # 05
Lecture No # 24
Valiant-Vazhirani Theorem - I

So then so what we so the idea behind showing this lemma was that we wanted to prove Valiant Vazirani's theorem. So let me state that theorem once again but before stating that let me just define this language so that it will be little be easier to state that theorem.

(Refer Slide Time: 00:35)



Lemma
some k then

$$\Pr_{h \in_R H_{n,k+2}} \left[\left| \{ x \in \{0,1\}^n \mid h(x) = 0^{k+2} \} \right| = 1 \right] > \frac{1}{8}$$

$USAT = \{ \phi \mid \phi$ has exactly 1 satisfying assgn $\}$

V V Thm $\exists$ a ppt m/c M st
given any Boolean formula ϕ (over n vars)

ϕ is sat $\Rightarrow \Pr[M(\phi) \in USAT] \geq \frac{1}{8n}$

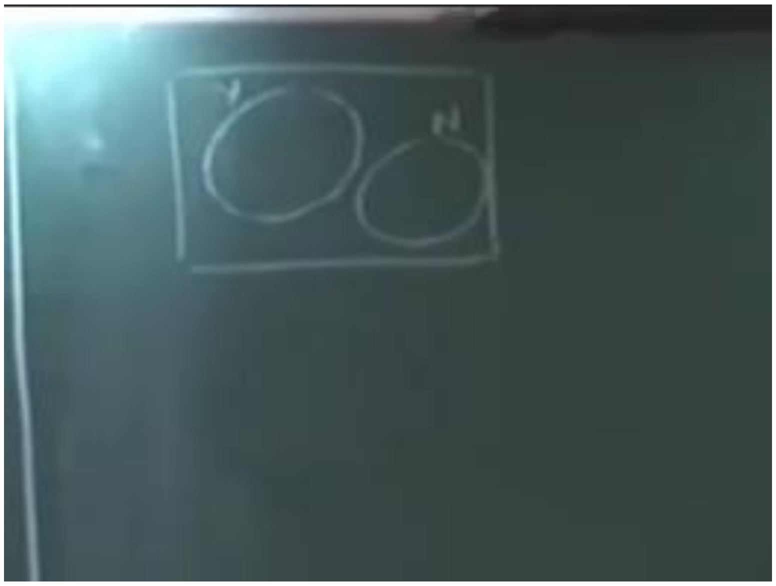
ϕ is unsat $\Rightarrow \Pr[M(\phi) \in SAT] = 0$

So you define this language U SAT to be set of all Boolean formulas let us say in CNF form or Boolean formulas Phi such that is as exactly one satisfying assignment. So now we can restate Valiant Vazirani's theorem in this context that there exist a probabilistic polynomial time machine M such that given any Boolean formula Phi. Let us say over n variables if Phi is satisfiable then the probability that M given Phi belongs to U SAT is greater than or equal to 1 over 8 n.

And if Phi is unsatisfiable and the probability that M given Phi shall make this a little bit more, stronger belongs to SAT is 0 so it is not even satisfiable the new formula that we construct. So this is just a rephrasing of the theorem that we stated at the beginning of last class. So what we said was that if Phi unsatisfiable the new formula that we construct is not satisfiable. And if Phi were to be satisfiable then the new formula has a unique satisfying assignment with at least so much probability.

So how does this follow from the lemma that we have so basically the way we are defining this language you said. So the way in which complexity theorists look at this language U SAT is they think of this as a promise problem. So it is not exactly a language or it is not exactly a problem in the traditional sense. In the sense that every string which belongs to the language as this property and every string which does not have the properties outside this language. It is not exactly that it is basically you have this set let us say so I will just refer this proof.

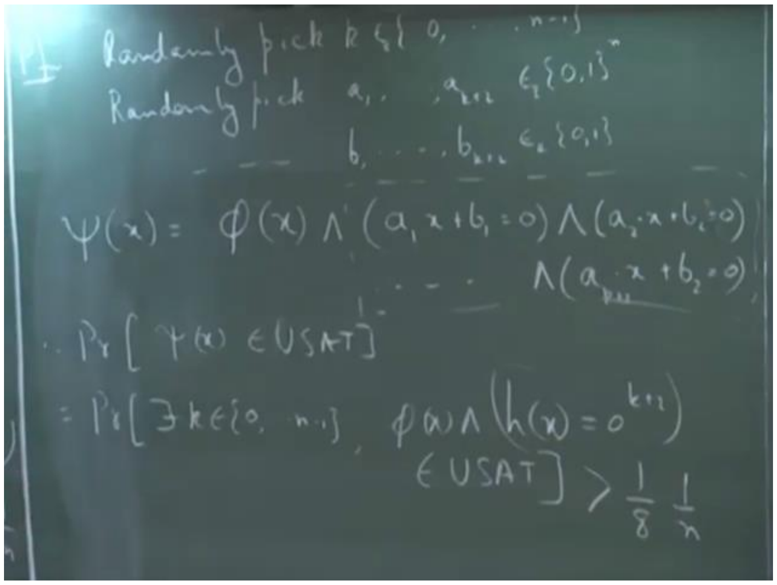
(Refer Slide Time: 05:03)



So we have this set let us say 0, 1 star or 0, 1 to the power n and you divide this into 2 sets. Set of all sum y and there is another disjoint set which I call n. So now we say that an instance belongs to U SAT if it belongs to the Y partition and it does not belong to U SAT if it belongs to the no partition. In this case our no will be set of unsatisfiable assignments and Y is the set of assignments with exactly 1 Y is the set of Boolean formulas with exactly one satisfying assignment.

Because exactly so here we have the promise that the input belongs to only by union of these 2 things but clearly there can be other formulas which lie outside which have let us say 2 satisfying assignments or things like that. So that is why I stated this as belonging to SAT instead of U SAT.

(Refer Slide Time: 06:31)



Pf Randomly pick $k \in_R \{0, \ldots, n-1\}$

Randomly pick $a_1, \ldots, a_{k+2} \in_R \{0,1\}^n$

$b_1, \ldots, b_{k+2} \in_R \{0,1\}$

$$\psi(x) = \phi(x) \wedge (a_1 \cdot x + b_1 = 0) \wedge (a_2 \cdot x + b_2 = 0) \ldots \wedge (a_{k+2} \cdot x + b_2 = 0)$$

$$\therefore \Pr[\psi(x) \in USAT]$$

$$= \Pr[\exists k \in \{0, \ldots n-1\}, \phi(x) \wedge (h(x) = 0^{k+2}) \in USAT] > \frac{1}{8}\frac{1}{n}$$

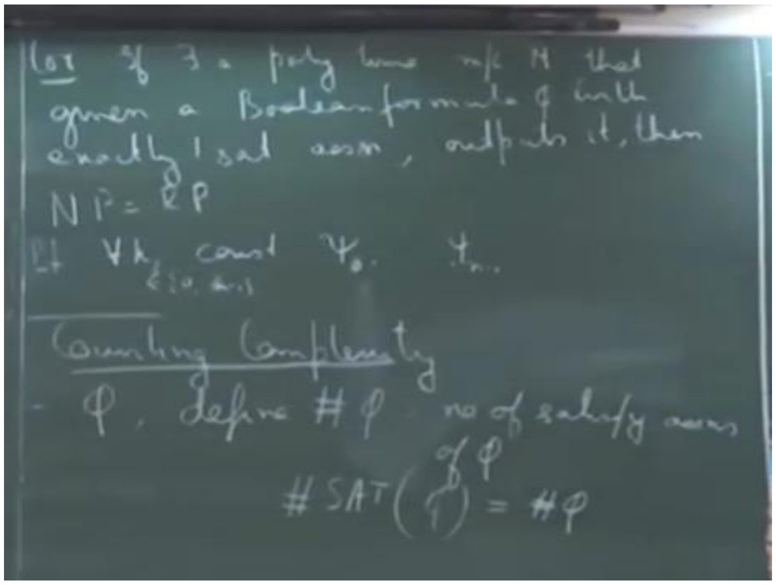
So these are known as promise problems in complexity but so how do we prove this? So the proof is not difficult basically what you do is? You construct a formulas Psi as follows. So you first randomly pick k from the following set 0 up to n – 1 and then randomly take a 1 so how many do we need? k + 2 a 1 through a k + 2 in 0, 1 to the power n in case such vectors. And also you pick these bits b 1 through b k + 2 and construct the formula let us call it Psi as follows.

So Psi of x is defined as it is the conjunction of the input formula Phi X with the following clauses. So the first clause is a 1 dot x + b 1 = 0. So it is basically the same way as this hash function works. If you just look at this entire thing that we have here this is nothing but saying that h of x = 0 to the power k + 2. You have so I am just splitting that into these k + 2 clauses and you can always come up with Boolean expressions with and or and not Boolean expressions with and or n not gets which will be equivalent to these whatever these clause states.

And then we can just apply the earlier lemma so the earlier lemma says that for some k such that the cardinality of t lies between 2 to the power k and 2 to the power k + 1. The probability that the number of x is that gets hashed to the all 0 string is 1 is at least 1 over 8. So therefore the probability that Psi of x is in U SAT is equal to the probability that so I do not want to write this entire thing.

But basically it is equal to that there exists a k such that this formula Phi of x conjunction with this h of x = 0 to the power k + 2. So this belongs to U SAT and if you just apply the union bound here this is greater than 1 over 8 times 1 over n. Because the probability that the correct k is chosen 1 over n.

(Refer Slide Time: 11:33)



Cor If $\exists$ a poly time m/c M that given a Boolean formula ϕ with exactly 1 sat assn, outputs it, then $NP = RP$

Pf $\forall k \in \{0, \ldots, n-1\}$ const $\Psi_0 \ldots \Psi_{n-1}$

Counting Complexity

- ϕ. define $\#\phi$ no of satisfy assns of ϕ

$\#SAT(\phi) = \#\phi$

So one immediate corollary of this result is that if there exist a polynomial time machine M that given a Boolean formula Phi with exactly one satisfying assignment output set. Then N P = R P so we know R P is contained in N P but suppose if you have a deterministic polynomial time machine which does the following. So suppose if the machine is given a formula Phi which does have this property then we do not assume any we do not assume anything about the machine.

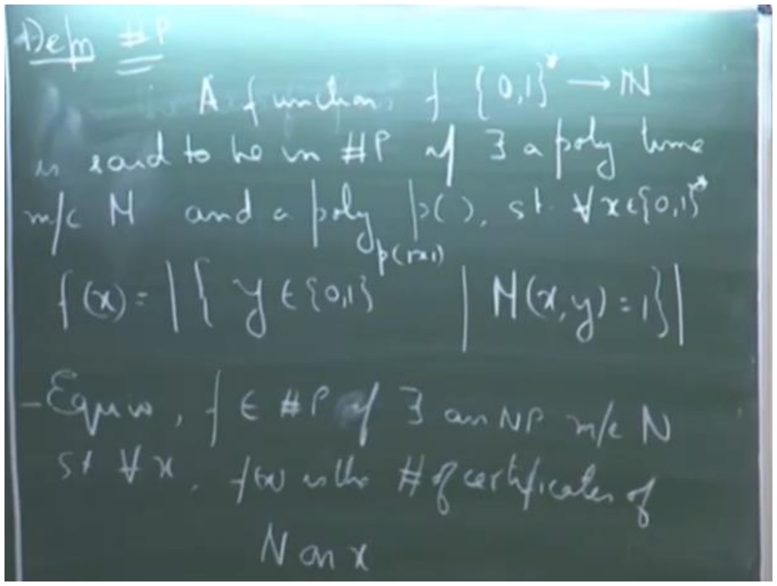
So it can do anything but if it is given a formula which has this property that it has only one satisfying assignment then the machine will output it. So the proof is again simple so suppose if you have such a machine what you do is given a formula Phi what you do is first you generate all, these different Psi. So you or so the idea is that for all k construct Psi 1 up to or Psi 0. So we are taking k in 0 up to n – 1 so you construct these formulas Psi 0 to Psi n – 1 and then you feed it to this polynomial time machine.

So we know from the earlier thing that there is one which has I mean one of these formulas have a unique satisfying assignment with high probability. If the given formula Phi where to be satisfiable with so the polynomial time machine outputs that formula and that will help us to check whether Phi satisfiable or not. So the, you can check the details but so any questions no but then you are strengthening the hypothesis you are assuming a more powerful machine.

I am only saying that if there exist, even a polynomial time machine which has less power. See R P machine has more power than a polynomial time machine. If it is an R P machine then it implies that it is so if we have this condition then it implies that M is also an R P machine. Basically which has 0 error you can see that also but we have a R P machine it will work. So any other questions so let us look at some definitions so we will look at some counting classes.

So in more generally we will look at what is known as counting complexity? And we will try to come up with an analogous theorem of Valiant Vanirani’s theorem. So if Phi is a Boolean formula define sharp Phi to be equal to the number of satisfying assignments of Phi. And in the same way you can define this function sharp set that given a formula Phi it outputs the number of satisfying assignment of that formula. So this is just a number it is not outputting all the assignments but just the cardinality of that set.

(Refer Slide Time: 18:24)



Defn #P

A function $f\ \{0,1\}^* \rightarrow \mathbb{N}$ is said to be in #P if $\exists$ a poly time m/c M and a poly $p(\,)$, st $\forall x \in \{0,1\}^*$

$$f(x) = \left|\left\{ y \in \{0,1\}^{p(|x|)} \;\middle|\; M(x,y) = 1 \right\}\right|$$

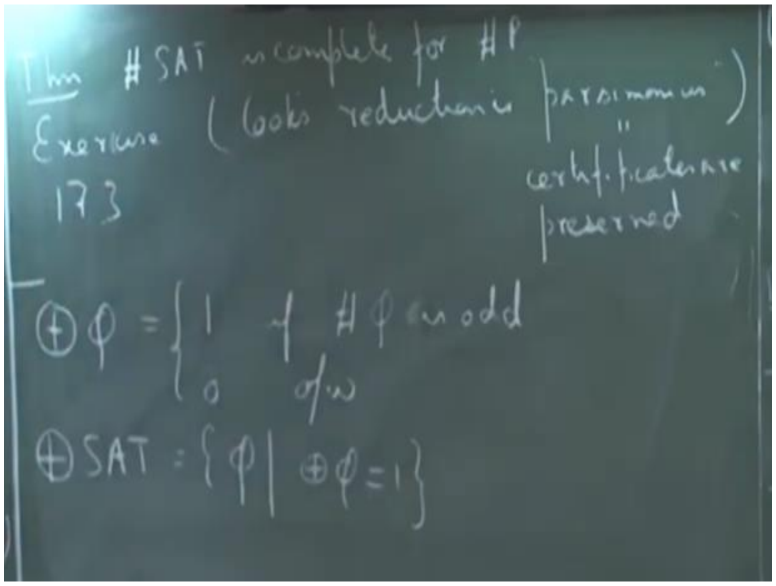
- Equiv, $f \in$ #P if $\exists$ an NP m/c N st $\forall x$, $f(x)$ is the # of certificates of N on x

So more generally we can define what is known as the class so you just state this as a definition sharp p. So this is not again a usual complexity class in the sense that this is not a class of languages per se but it is a class of functions. So sharp p is the class of functions f from 0, 1 star to natural numbers let me just rephrase this a little bit. So the function f is said to be in sharp p if there exist a polynomial time machine M.

And let us say a polynomial p such that for all x f of x is the cardinality of the set of all strings y having size p of the length of x. Such that m given x and y such that accepts so in other words what is this class of functions? It is in terms of the language N P it is the number of certificates. So an equivalent way of looking at this definition is so f belongs to sharp p if there exist and N P machine n such that for all x f of x is the number of certificates of n on x.

So if you simulate the machine n on x f of x is the number of certificates so it can be 0 also in particular when the string does not belong to the language of n number of certificates is 0. And that is the only time when it is 0 so let me just state this result.

(Refer Slide Time: 22:26)



Thm #SAT is complete for #P

Exercise 17.3 (Cook's reduction is parsimonious)

" certificates are preserved

$$\oplus\phi = \begin{cases} 1 & \text{if } \#\phi \text{ is odd} \\ 0 & \text{o/w} \end{cases}$$

$$\oplus SAT = \{\phi \mid \oplus\phi = 1\}$$

So this language or this function sharps at. So this is complete for sharp p and the proof of this basically observing that the reduction Cook's reduction that, we saw earlier in this course is basically a reduction which preserves certificates. So if you are given a language in N P and for every string the corresponding formula that you have constructed that; was basically a reduction which preserved certificates.

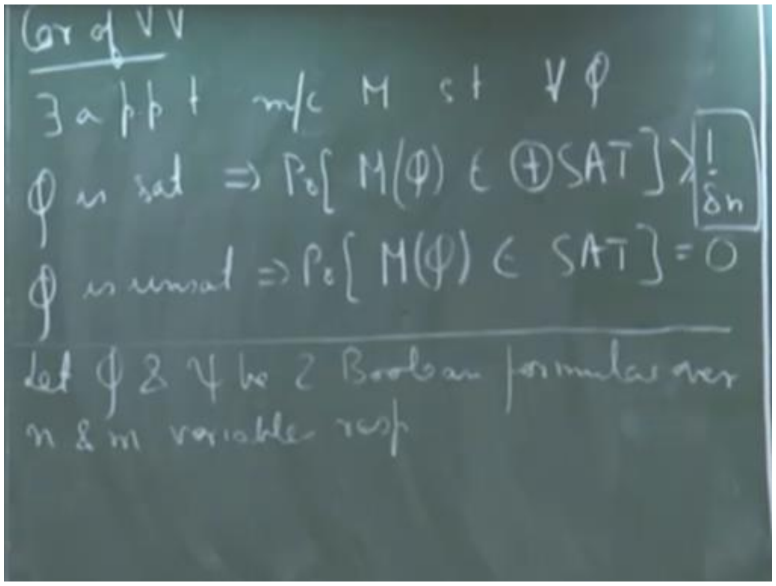
So whatever certificates that machine had for that string the corresponding Boolean formula will also have the same number of certificates. So I would not again going to the details of the proof but you can check it for yourself. So I will just leave this as an exercise with the comment that

Cook's reduction is. So the technical term for this property is that it is parsimonious. In other words certificates are preserved.

So I think even in the book if you look at chapter 17, section 3 there is a short proof of this result. I mean they do not give the proof I mean they just argue that why is Cook's reduction parsimonious? So you can look at the following section but so let me come back to where we are? So similar to that sharp quantifier we can also define an, other quantifier so this is known as parity.

So parity of Phi is equal to 1 if; the Boolean formula Phi as an odd number of satisfying assignments that this is odd and it is equal to 0 otherwise. And again using this quantifier we get the language parity SAT which is set of all formulas Phi such that parity Phi is equal to 1. All those Boolean formulas which; have an odd number of satisfying assignment.

(Refer Slide Time: 26:19)



Cor of VV

$\exists$ a ppt m/c M st $\forall \phi$

ϕ is sat $\Rightarrow \Pr[M(\phi) \in \oplus SAT] \geq \boxed{\frac{1}{8n}}$

ϕ is unsat $\Rightarrow \Pr[M(\phi) \in SAT] = 0$

Let ϕ & ψ be 2 Boolean formulas over n & m variables resp

So, now as an immediate corollary of Valiant Vazhirani's theorem we get the following result that their, exist a probabilistic polynomial time machine M such that for all Boolean formulas Phi. If Phi is satisfiable then the probability that M of Phi belongs to parity SAT is greater than 1 over 8 n and if Phi is not satisfiable. Then the probability that M of Phi is also not satisfiable has one probability I mean in other words if M of Phi satisfiable is has 0, probability.

And this is again obviously true because if M of Phi belongs to U SAT it means it has only one satisfying assignment which clearly means that it belongs to parity SAT as well because it is odd so 1 is also that is the reason why this is true. But so this is something which seems very obviously true; and a question is that why are we going in this direction. So the reason we are going in this direction is because we want to amplify this probability.

So let us come back to Valiant Vazhirani’s theorem and let us ask the same question here that is it possible for us to amplify this probability that is given by Valiant and Vazhirani. Can we do any better than e over 8 n? In other words suppose if a formula is satisfiable can I construct some formula Psi which belongs to U SAT with a value that is greater than 1 over 8 n. So again last class we saw this argument has to how the success probability of a R P machine can be amplified.

It is basically by taking some number of trails and you take the r of all those trails. I mean if one of those trials gives a positive answer you output a positive answer. And then the probability that you do get a negative answer is 1 minus the product of all these things which is very small. But what happens here I mean what happens if it try to simulate that same argument here. So let us say that this is the formula Psi that we get so when I run this machine once I get a formula Psi.

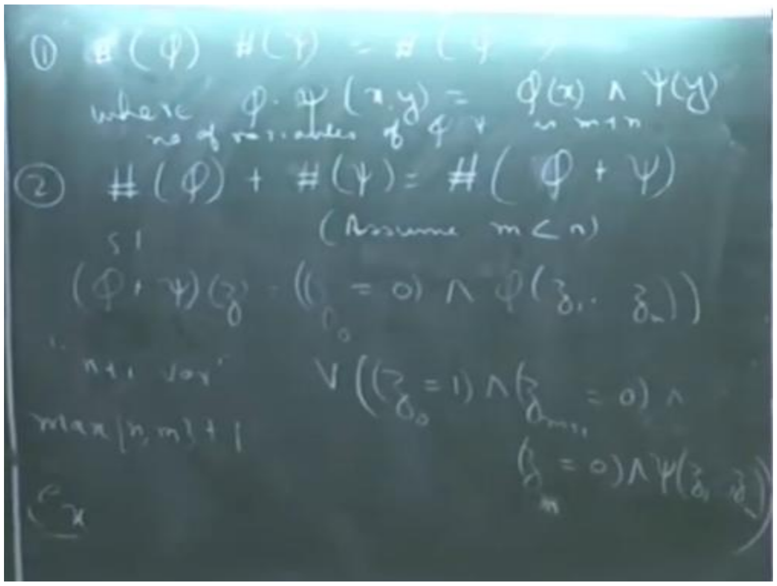
Suppose I want to amplify this probability and I run this machine a second time. And let us say I get a formula Psi 1 now suppose that one of those formula were to be true and the other is false can I clean that or can I construct a formula from Psi and Psi 1 which belongs to U SAT. So that is something which is not known to be true in other words if you let us say if you take some t trials of this machine M you run this some t times and you get some t formulas Psi 1 up to Psi t.

How can you combine these formulas such that if the original formula where satisfiable? And the combination of all these, t formulae give you only one satisfying assignment so that is something which is we not known. In fact it is an open problem whether any number that is better than 1 over 8 n can be achieved for this theorem. So that is an open question so can we do better than 1 over 8 n?

So this is open in the context of the U SAT version of valiant and Vazhirani. But as it turns out parity SAT is much more expressive parity SAT has much more power than U SAT and we can go ahead and amplify this probability to any number which is exponentially close to 1. So that is what we want to show so let us see why that is true so, is it clear? What we are trying to get here any questions?

So let us some properties of this quantifier parity so let Phi and Psi be 2 Boolean formulae over n and m variables respectively. Then what we can do is we can construct a Boolean formulae whose I mean which have the following property.

(Refer Slide Time: 33:35)



① #(φ) #(ψ) = #(φ·ψ)
where φ·ψ (x, y) = φ(x) ∧ ψ(y)
no of variables of φ·ψ is m+n
② #(φ) + #(ψ) = #(φ + ψ)
(Assume m < n)
(φ + ψ)(z) = ((z_0 = 0) ∧ φ(z_1 .. z_n))
n+1 var
∨ ((z_0 = 1) ∧ (z_{m+1} = 0) ∧ (z_n = 0) ∧ ψ(z_1 .. z_m))
max{n, m} + 1

So the first thing is that their exist let me set it this way if we look at the so the number of satisfying assignments of Phi taken as a product with the number of satisfying assignments of Psi is equal to the number of satisfying assignments of other Boolean formula Phi for Psi where we define this Phi dot Psi. Let us say over the set of variables x, y as Phi of x and Psi of y. So let us just see what we are saying here so suppose of you have 2 Boolean formulae Phi and Psi over the set of variables x and y.

So think of x and y as a vector of variables so I can just write that here where x has n variables and y has m variables. Then if it consider the formula Phi dot Psi over the set of variables which is a concatenation of all the variables in x and y defined as the conjunction of these 2 formulae. Then the total number of satisfying assignments of this new formula is a product of number of satisfying assignments of these 2 individual formula.

And so this is easy to see because you take any satisfying assignment of Phi and you take any satisfying assignment of Psi and you just concatenate those 2 that will give you a satisfying assignment of Phi dot Psi. And similarly you take satisfying assignment of Phi dot Psi and you just look at the appropriate partition. So the individual bits strings will be satisfying assignments of these 2 individual formulae.

So also we can do a similar thing if you want the some of the 2 so suppose I want to define a formula. Let me call it so such that number of satisfying assignments of Phi plus the number of satisfying assignments of Psi is equal to the number of satisfying assignments of this new formula which we denote as Phi + Psi. Such that so this is what we want so how do we define this? So will or work or will actually give much more?

Actually it will be so it will have some M times 2 to the power n plus n times 2 to the power n. So what you want is just this plus this you want much lesser or will not work. So the way you can do it is there is a little tricky way to define this so before we go into this plus thing how many variables does this new formula have in terms of Phi and Psi? So the number of variables of Phi dot Psi is m + n.

So let us come here so Phi + Psi we define a new set of variables z as equal to. So let us assume that m is less than n. So it is the formula z naught equal to 0 and Phi of z 1 up to z n or the clause z naught equal to 1 and z m + 1 is equal to 0. So 1 up to z n is equal to 0 and Psi of z 1 up to z m. So let us take a moment here and understand what is happening? So we construct this formula of Phi + Psi.

So in this case it is a formula over n + 1 variable so because we assume that n is greater than m. But in general basically the number of variables Phi + Psi would be the max of n, m + 1. So here since we are assuming n is the larger number we have z as a, over n variables. So z m + 1 and z m + 2 so 1 up to z m because m is smaller than, n that is why? So exactly what is happening here so let us understand the intuition behind this construction.

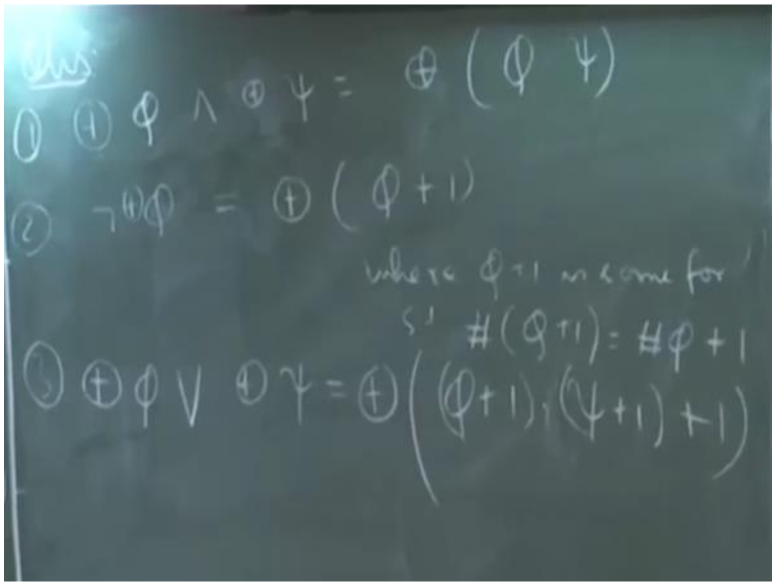
So what we want is that only if one of the so let us look at the 2 clauses that we have so this is one clause and this is the other clause. So one property of this entire construction is that both these clauses cannot be, prove at the same time. Because one contains z naught equal 0 and the other contains n naught equals 1. So only one of them can be true more over let us say the first clause is true then it implies that the Phi is also true it has a satisfying assignment.

And of the second clause is true then it implies that Psi is equal to 2 so if we take a satisfying assignment of Phi that we can translate into a satisfying assignment of Phi + Psi just by setting z naught = 0. And if we take a satisfying assignment of Psi we can translate that into a satisfying assignment of Phi + Psi again by setting z naught = 1 z m + 1 = 0. So 1 up to z n = 0 and whatever is the value that is therefore z 1 to z n.

So note that these variables are exclusive so z naught only appears once here and then we have z 1 to z n and here again z 1 to z m does not appear in any of these equalities. So that is one direction similarly we can also look at the other direction that if Phi + Psi as a satisfying assignment. As we argued earlier that both these clauses simultaneously cannot be true so if this as a satisfying assignments then either z naught = 0 or z naught = 1.

So if z naught = 0 then it means that there is a satisfying assignments of Phi as well and if z naught equals 1. It means that these clauses have I mean these variables z m + 1 through z n all have a value 0 and there is a satisfying assignment for Psi as well. So you can see both directions so that is the essential idea but I will just leave it as an exercise to formally to prove this. So just go back and write of the proof so is this construction clear?

(Refer Slide Time: 45:45)



Obs:

① $\oplus\varphi \wedge \oplus\psi = \oplus(\varphi\ \psi)$

② $\neg\oplus\varphi = \oplus(\varphi + 1)$

where $\varphi + 1$ is come for s.t. $\#(\varphi+1) = \#\varphi + 1$

③ $\oplus\varphi \vee \oplus\psi = \oplus\big((\varphi+1)\cdot(\psi+1)+1\big)$

So now let me just state one more property and then we will leave it at that point take this as an observation. So firstly if you look at parity of Phi and parity of Psi so this is equal to parity of the formula Phi dot Psi. So in other words if so when is, the left side true or when does the left side evaluate to 1. When both these are odd when both these have an odd number of satisfying assignments so if these have an odd number of satisfying assignment the product of 2 odd numbers is odd which means that this is also true.

And similarly the only way in which the product of 2 numbers can be odd is if both the numbers are odd which implies that this is also true and this is also true. So that is easy to see the second property is that so basically what we are showing here is that this parity operation is or this parity quantifier is closed under the Boolean and operation. Similarly it is also closed under the not operation.

So naught of parity of Phi is equal to parity of the following formula so I will call it Phi + 1 where Phi + 1 is some formula. Such that number of satisfying assignments of Phi + 1 is equal to number of satisfying assignments of Phi + 1. So in other words you can take this formula to be Psi where it is some trivial formula. Let us say x it has only one variable so it has only one variable then it has exactly one satisfying assignment which is 1.

So you can trivially construct a formula by applying this construction so now why is this true? Because if this where to be true then now this becomes odd or even which makes this false and vice versa. The third one is the little bit tricky but again not difficult to see is with respect to the, or operation. So parity Phi and parity Psi is equal to again parity of a single formula so what do you think it should be?

So we want this to be true if at least one of them is true so you can also use the naught also by using De Morgan’s law but you can give them a direct formula. So it is Phi + 1 product it is Phi + 1 dot Psi + 1 + 1. So if one of them is true so let us say this is true that means this has odd number so then this will have an even number. So an even number product with any number is always even.

So this entire thing becomes odd and the only way this will become even is when both these numbers are odd which is when both these have an even number of satisfying assignment which means that both these are false. So that is what I had so now actually you can see yourself that you can try to see how this will help you to boost this probability but anyway we will discuss it next time how these things can be used. So anyway we have over short will stop here.