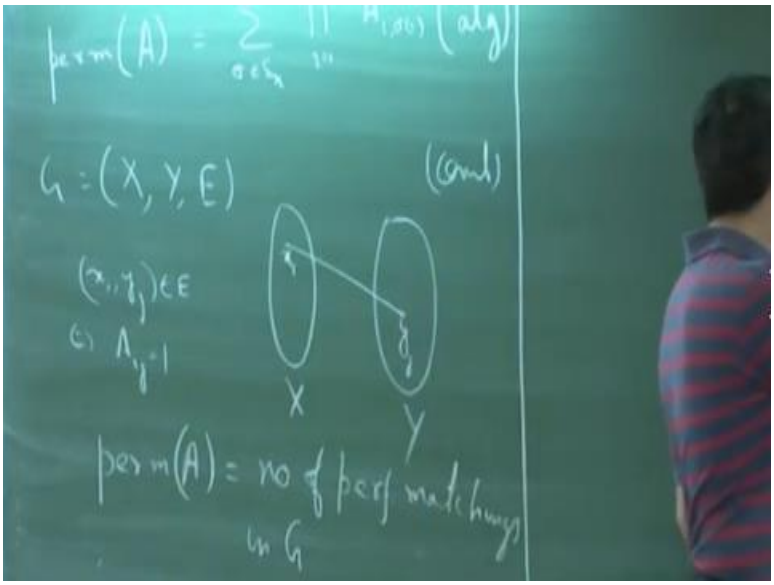
Computational Complexity Theory
Prof. Raghunath Tewari
Department of Computer Science and Engineering
Indian Institute of Technology, Kanpur

Lecture -29
Introduction

So, let us get started, so what we will see today is the permanent function and why it is sharp P hard, actually it is sharp P complete so we will see both the directions. So, let us recall the definition of permanent.

(Refer Slide Time: 00:40)



So, suppose A is a n cross n matrix and just for the time being we will assume that A is a 0, 1 matrix. So, the permanent of A is the sum over all permutations on n variables of the product of A i sigma i's over all i's. So, this is an algebraic definition of the permanent but there is also a nice combinatorial definition or I mean I should not say it is a definition but combinatorial representation of what the permanent is.

So, in general the matrix A can be any matrix, I mean it can have values that are greater than 1 and it can also have negative values, if you look at the definition this does not exclude any such so this does not have any such restrictions. But suppose we have a matrix with only Boolean

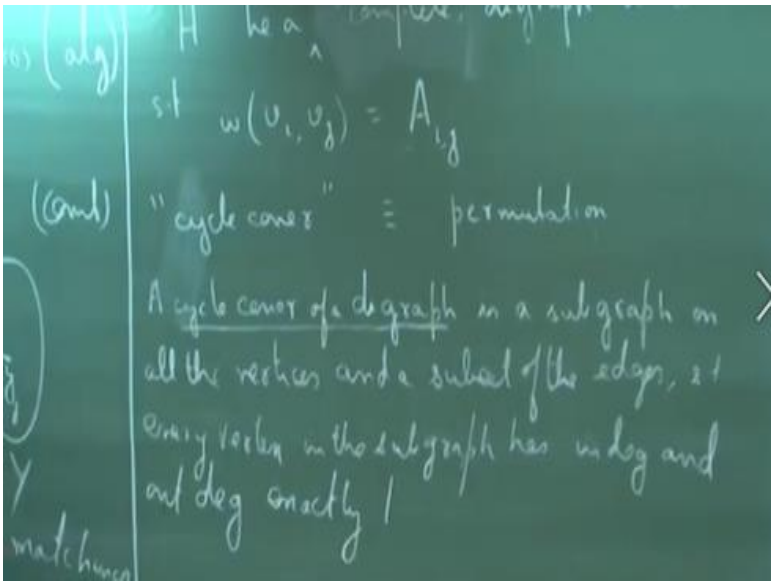
entries that is 0, 1 we can look at the permanent as, so consider the following bipartite graph, so G is a bipartite graph with bipartitions X, Y and a set of edges E such that.

So, let me draw this so we put an edge from the vertex x i to a vertex let us say y j if and only if the i, jth entry of A is 1. So, x i, y j belongs to E if and only if A ij is 1. So, this is the construction that we make, so what does the permanent translate to in this graph? So, what is one permutation? So, let us say we fix a permutation what does a permutation correspond to in this graph.

So, a permutation is basically a function which maps it is a 1 to 1 and onto function which maps every element i to an element j within the same set that is bit it is from 1 to n. So, what does a permutation correspond to in this graph but graph theoretically what is it? So, what is the bijection between these two sets? It is a perfect matching I mean it is a set of I mean it is a subset of the edges such that it covers every vertex in X as well as Y.

So, the what the permanent translate two is in this graph is the permanent of A is the number of perfect matching's in G because we are taking the sum over all permutations. So, this is one combinatorial way to look at what the permanent is there is. Also, another way I mean another combinatorial view so suppose A is a general matrix that is it can have values which are not zero or one.

(Refer Slide Time: 05:31)

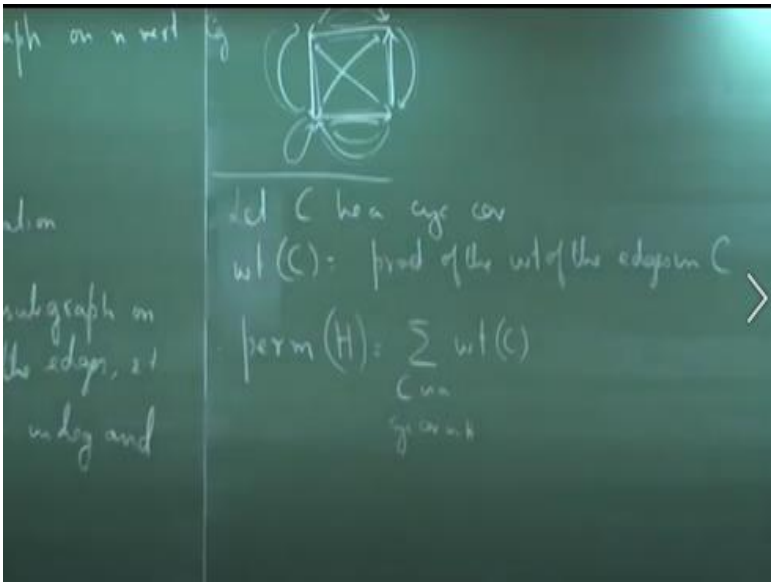


So, then consider the following graph so let H be a complete directed graph on n vertices so it is also weighted, so we weighted complete directed graph on n vertices such that weight of let us say the edge v i, v j so suppose these are the vertices of H so this is equal to the entry A i j put comma here. So, suppose we construct a digraph from the matrix in this manner, so then what does a permutation correspond to in this graph?

So, what a permutation so let me give the answer so what a permutation corresponds to in this graph is what is known as a cycle cover. So, there is basically a 1 to 1 mapping between a permutation in S n to a cycle cover in this graph H. And so, what is the cycle cover? So, cycle cover of a graph is a sub graph on all the vertices and a subset of the edges, such that so again when we talk about cycle covers, we talk about directed graphs.

So, let me include that in the definition so cycle cover of a directed graph is a sub graph on all the vertices and a subset of the edges such that every vertex in the sub graph has in degree and out degree exactly 1.

(Refer Slide Time: 09:43)



So, let us look at a very small example so suppose we have a square and these are the set of edges. So, what is the cycle cover in this graph? So, this is a cycle cover if I include, I mean if I consider these four edges because all the vertices will have in degree and out degree exactly one in this sub graph. Similarly, this is a cycle cover, so there are many other cycle covers for example this together with this is a cycle cover and as you can see there are several others.

So, it is basically a collection of cycles I mean the way it is defined I mean if you have a sub graph where every vertex has in degree and out degree exactly one what it will translate to is basically a collection of cycles and since I am required that the sub graph is on the set of all vertices on all the vertices of the graph it is basically a set of cycles a set of disjoint cycles that covers all the vertices of the graph.

So, the question is that why is this true I mean why does a permutation correspond to a cycle cover? Again, it is quite simple I mean if you just look at the permutation I mean if you just look at any permutation if you just follow the indices of that permutation. For example, if 1 gets mapped to 5 you go from vertex 1 to 5 let us say then 5 get maps to 3 from 5 you go to 3 so at some point you will come back to 1.

So, since it is a so since n is a finite number so that will basically give you one cycle, so it might happen that all the elements between 1 and n are not covered in this cycle. So, then you choose

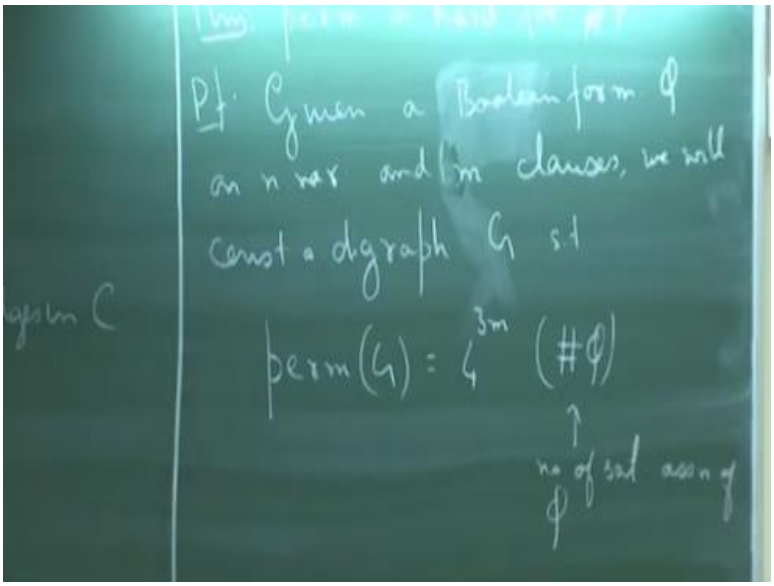
the first vertex or the first number that is not covered and again you construct another cycle. So, again since n is finite if you look at finitely many such cycles it will cover all the elements between 1 and n.

So, that is the reason why if you take any permutation you will get a cycle cover, any cycle cover that will give you a corresponding permutation. It can happen, so I can have a cycle I mean I can have the graph where I have self-loops, I mean I can have something like this as well, this kind of an edge is permitted. So, here although I did mention that it is a complete digraph but let us say the edges which have weight 0, we just do not draw them explicitly.

So, when I say complete it means that between any pair of vertices there exists an edge irrespective of whether it is the same vertex. So, now that is what a cycle cover is. So, the weight of a cycle cover is defined as, so let C be a cycle cover. So, weight of C is the product of the weight of the edges in C. So, therefore the permanent so now I can talk about the permanent of let us say H is basically so it is the sum over all cycle covers C of the weight of C.

So, C is a, so for all the cycle covers I just take the sum. So, it is just basically translating this definition nothing more.

(Refer Slide Time: 15:27)



So, now let us see why this theorem is true? So, what we will show is that actually what we will basically show today is that permanent is hard for sharp P. So, the way the construction will go is that so given a so sharp SAT is a complete problem for sharp P. So, basically counting the number of satisfying assignments of a Boolean formula, so given a Boolean formula we will construct a graph.

Such that there is so there will be a way to compute the number of cycle covers of that graph given the number of satisfying assignments of the Boolean formula and the graph that we will construct will have arbitrary weights on its edges I mean arbitrary in the sense that it will not be restricted to 0, 1 weights but again that can be reduced to other graph where it has only 0, 1 weights.

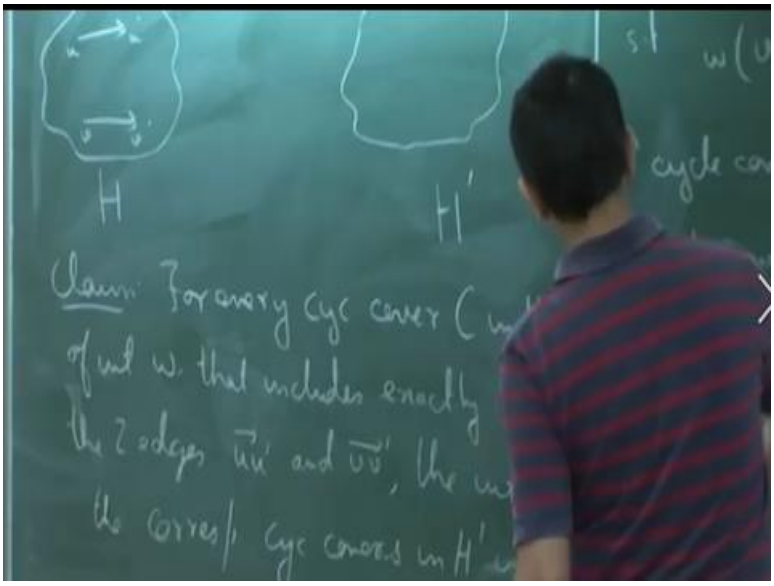
So, in a sense what is known as the permanent with only 0, 1 entries is hard for sharp P but we will look at a slightly weaker version of that theorem. So, again I will not give the entire proof I will just give the main ideas, so the idea is that given a Boolean formula phi on n variables and having m clauses we will construct a graph G, digraph G, such that permanent of G will be 4 to the power 3 m times the number of satisfying assignments of phi.

So, this called, is basically this denotes the number of satisfying assignments of the formula phi. So, again I mean once we get a digraph it is very easy to get the corresponding matrix so that trivial. Any questions so far? How do we get that number 4 to the 3m? So, that will come from the construction so we will see that construction, so before I proceed with one question, so can someone tell me why permanent is in sharp P?

So, let us say we are looking at permanent over 0, 1 matrices which basically means that how many perfect matching say bipartite graph has, so why is this a sharp P function? Exactly, so because computing a perfect matching in a bipartite graph is known to be in polynomial time what you can do is you just guess a set of edges and you just can verify whether this sequence of edges give you a perfect matching.

So, this is an NP computation and the number of accepting paths of this NP computation is exactly the number of perfect matching's in the graph. So, as I said we will just look at an overview of the proof so what we will do is so this is a very nice proof in the sense that it involves the construction of some nice gadgets and how these gadgets are interconnected to give this nice property.

(Refer Slide Time: 21:06)



So, the first gadget that we will see is what is known as the XOR gadget so before I go ahead so let me state so this was proved by Valiant and this was shown in 1979 in a very important paper by Valiant, because prior to this I mean because of the similarity between the two algebraic definitions of permanent and determinant. So, this was something which was not quite believed to be true.

So, it was believed that probably there is also some polynomial time algorithm to compute permanent and it is just more difficult than determinant that is all. But because of this result what would imply that if you can get a polynomial time algorithm for permanent it would immediately show that P is equal to NP and that is the reason why it is not believed that permanent has a polynomial time algorithm.

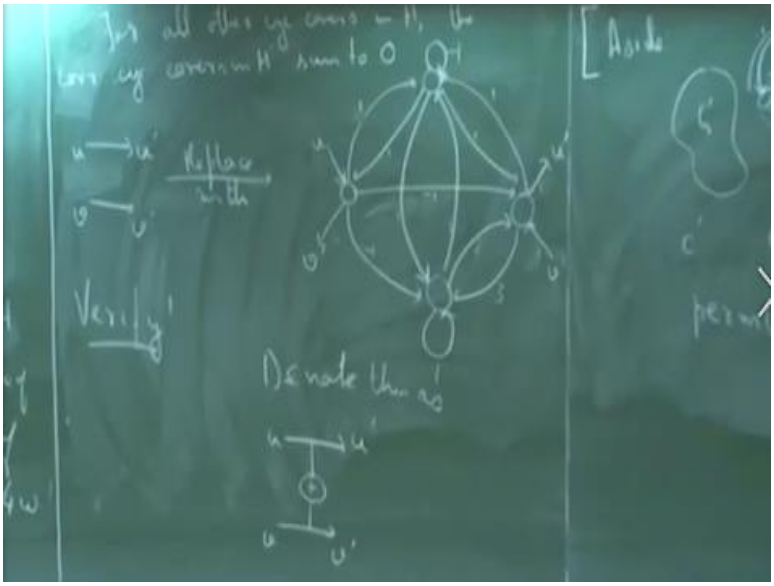
And that is the reason why these two functions are very different in complexity. That is why this was a very important result at that point of time. So, coming back to the construction so this

XOR gadget, what it does is. So, suppose H is a graph that has a sum two edges u, u prime and v, v prime and what I want to do is from H, I want to construct another graph H prime with the property that every cycle cover in H that includes I mean for every cycle cover in H that includes exactly one of these two edges.

I will have a set of cycle covers in H prime with a certain weight. So, if w is the weight of a cycle cover which includes exactly one of these two edges the set of cycle covers in H prime will basically have weight 4w and for every other cycle covers that is cycle covers let us say which does not include any of the two edges or let us say which include both the edges the set of cycle covers in H prime that I will have weight 0.

So, that is the property that I want to have in H prime. So, let me just state that before I construct the gadget. So, for every cycle cover C in H of weight w that includes exactly one of the two edges u, u prime and v, v prime the weight of the corresponding cycle covers in H prime is 4w.

(Refer Slide Time: 25:45)



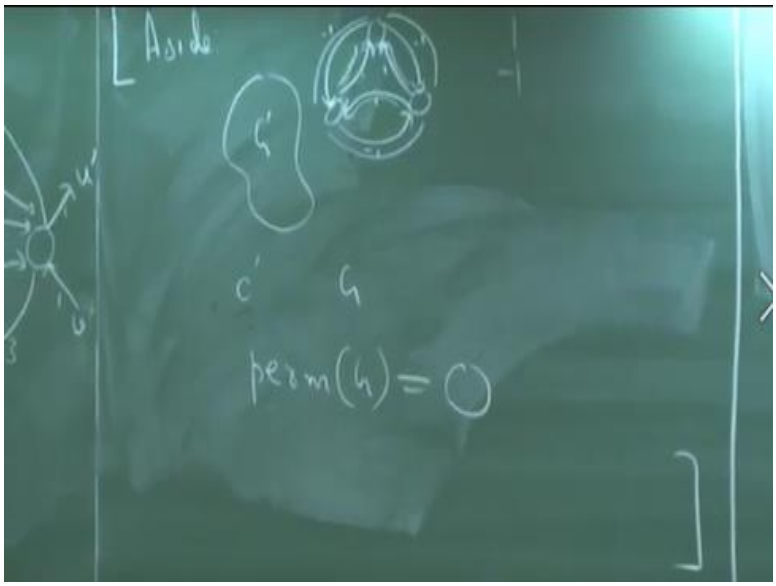
And moreover, so for all other cycle covers in H the corresponding cycle covers in H prime will sum to 0. So, what we do is we construct this very fancy gadget which looks like this so we replace these two edges u, u prime and v, v prime with the following gadget. So, we had u, u prime and v, v prime, so this gadget has four vertices. So, this is so these are the two vertices corresponding to u, u prime and these two vertices correspond to v, v prime.

So, this is v and this is v prime and so I have to so this edge has weight - 1. So, both these edges have been let me use a different color. So, those are all the edges so these four edges have weight 1 each this has again -1 this has -1 this has 1 and these two edges have weight 2 and 3 respectively and these 2 again has 1 each and these four also has 1. So, I will just leave the verification to you.

So, you verify that suppose if you take a cycle cover which includes only u, u prime or only v, v prime then the sum of the weights in this gadget will be equal to 4w in other words what whatever was w that will be multiplied by 4 and for any other cycle covers that is if there is a cycle cover which includes both u, u prime and v, v prime or neither of them basically whatever other cycle covers that occur here they will cancel of each other.

So, I will give you a very small example, so I will not go through the various cases because that will take a long time so this is not very difficult is just looking at the various cases whatever are the possible cycle covers and what are their respective weights and how they cancel out each other, so that is all there is to it. But let me give you a small example to motivate how things cancel out. So, this is aside this is not part of the proof.

(Refer Slide Time: 30:50)



So, again this is from your text so suppose you have a graph let us say G prime you have a graph G which has some component G prime and another component on three vertices that looks like this and let us say we give - 1 to the outer edges and 1 to the inner edges, how many cycle covers does this small graph have? So, let us just try to explicitly draw them, so this is one cycle cover so I take this edge, this edge and this edge, any other cycle covers?

I can take this, this and this. This will also give me another cycle cover and in fact these are the only two cycle covers in this graph. So, what are their weights so weight of the purple cycle cover is - 1 time - 1 times 1 so that is 1 and weight of the green cycle cover is 1 times 1 times - 1 which is - 1. So, now let us take any cycle cover in G prime so let us say that C prime is some cycle cover in G prime.

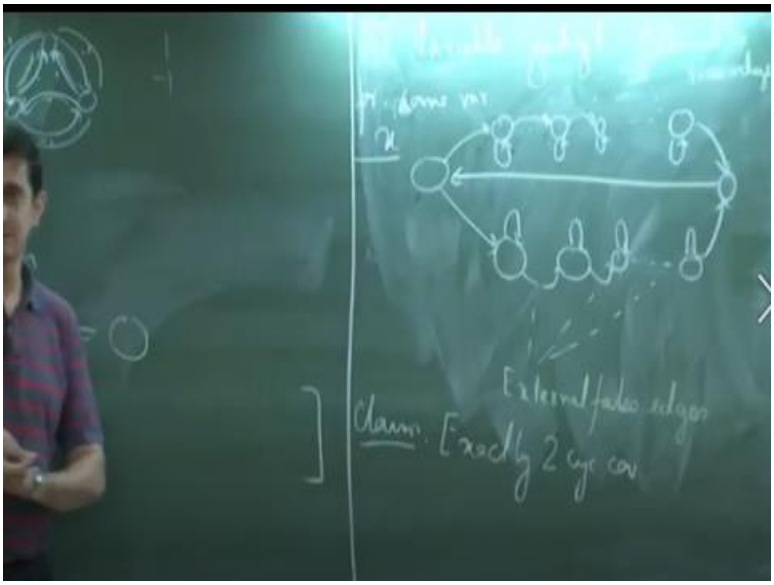
If I look at a cycle cover in the entire graph it will basically be C prime together with either of these two cycles covers and because their weights are negative of each other they will basically cancel out. So, every cycle cover in the entire graph will correspond to basically some cycle cover in G prime and some cycle cover here but there will be again some other cycle cover which will be the same cycle cover in G prime.

But the other cycle covered in this smaller sub graph and these two will cancel out. So, basically what this tells us is so let G be this entire graph, what would the permanent of G be? It will be 0, because all the cycle covers will basically cancel off with their counterparts. So, this also kind of proceeds in a similar fashion well not completely but all the cycle covers which include both these edges or none of these edges will cancel off in a similar fashion.

So, I will leave the verification to you but let us get back to the proof. So, I should not have said one here actually the way the thing will go is that they will have a weight one always but us just keep this blank. So, if they have some weight which is not equal to 1 will plug those weights in these 4 vertices. Well I can give one of those 2 vertices 1 and the other I will give whatever are the weights of those two.

Because it will anyway be a product, but it will not be the case I mean when you see the entire construction, they will have weight one basically. In fact, in the remaining part of the graph all the other edges will have weight 1. It is only in this gadget that you have weights other than 1. For every gadget you are introducing these four new vertices. And let me denote this as u, u prime this. So, I will just use this shorthand notation to denote the gadget. So, now how do we use these XOR gadgets so let us look at some more.

(Refer Slide Time: 36:11)



So, the second type of gadget that we will have is a gadget corresponding to every variable. So, for every variable x we will have a gadget that looks as follows and they also have a self-loop setting at all these vertices. So, we will call these edges the external edges so these ones, these will be called the external true edges and these ones will be called the external false edges. So, we will see later that how many of these external true and false edges will be required.

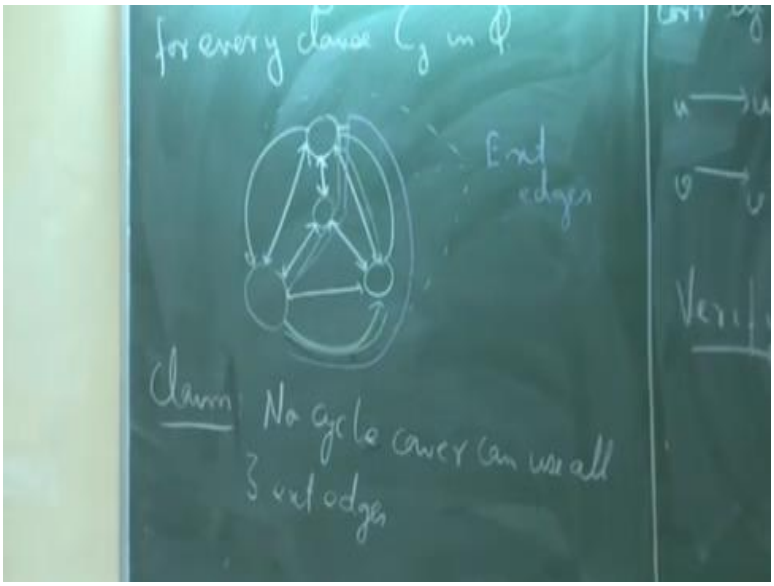
But for the, time being I just leave it as some arbitrary number will fix that later. So, the point is in this gadget how many cycle covers does it have. So, let us start at this vertex maybe so where all can we go from this vertex up or down so let us go up so if I go up, I have two choices I can either take this but that will not give me a cycle cover because then I will be repeating this vertex.

So, the only choice that I have to go to the next vertex and so on I come to this vertex and come down to this vertex and then I have only one choice to come back here, so that will give me one cycle which will cover all these vertices but how do I cover the bottom vertices the only way to cover the bottom vertices is by using the self-loop there is no other way. I cannot use these external false edges to cover them.

So, that is one cycle cover and I will have another cycle cover where I take this edge and take all the external false edges and come back. But then the only way to cover these top vertices is to use this self-loop. So, there are exactly two cycle covers in this graph, one corresponding to the true edges and the other corresponding to the false edges. So, what we will do is so the reason why we call these external is that.

So, later on we will be connecting these external edges to other gadgets using these XOR gates so these XOR gates are basically connectors. So, they connect sum to edges in your graph using this kind of a mechanism. This whole gadget is for a single variable so for all the variables we have a separate gadget, so this is for some variable x. So, you have a separate gadget for all the others, let me say x i may be that is better way. So, that will fix later so I will just come to that how many vertices we need.

(Refer Slide Time: 40:56)



So, the last type of gadget is what is known as the clause gadget. So, the clause gadget has only four vertices, so again this corresponds to every clause. So, for every clause let us say C j in phi. So, this has four vertices. So, for the clause gadget we will call these edges as the three external edges and the rest are internal edges. So, basically these three edges will correspond to the three literals in a clause.

So, when we are looking at a 5, we are looking at a 3 CNF Boolean formula. So, these internal edges are both sided edges, so there is basically an edge in both directions. So, there is one edge from here to here and one is from here to here. So, that is true so there are two edges going in the same direction but that is fine I mean we can I mean replace them with a suitable edge later so after the entire graph is constructed we can basically replace that edge with one single edge with a certain weight that will take care of both those edges.

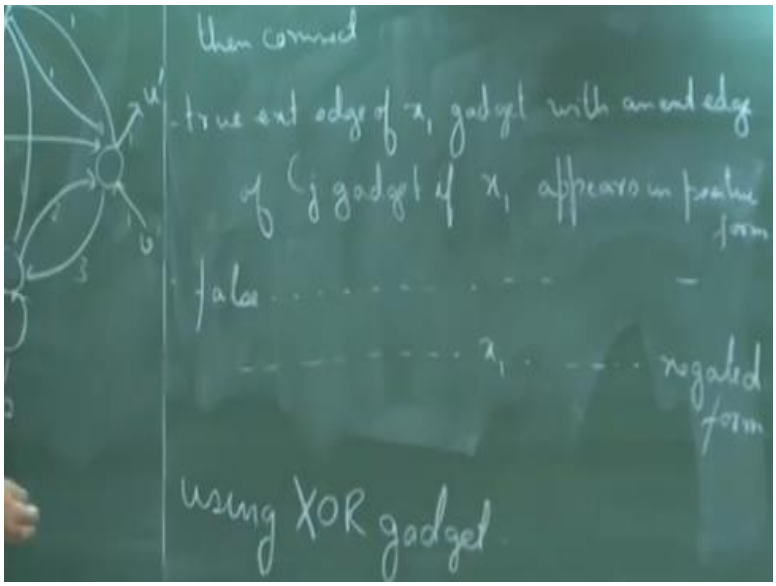
So, those are not difficult, so you are right. So, we have basically two edges going in the same direction but let us just have it for the time being. So, the property that this clause gadget has is that no cycle cover can use all three external edges. So, if there is some cycle cover which is using all the three external edges then this middle vertex will remain uncovered. So, the only cycle covers of the clause gadget are those that omit at least one of the three external.

In fact for every omission I mean if you look at every proper subset of these three external edges here is a cycle cover corresponding to it. So, let us look at some examples so suppose if I want to consider only this edge if I want to consider only this edge, I have a cycle cover which goes like this, this way this way and then comes here. If I want to consider two of them let us say I want to consider this and this.

I can have a cycle cover which goes like this, this and again so come down and completely. And similarly, if you want a cycle cover which does not use any of the three external vertices again that can be achieved easily. One way to do that is to let say take these two edges and these two edges. So, you can basically fix one cycle cover for each proper subset of these external edges. So, now the final claim is that let me just leave this here.

so to complete the construction how do we connect this. So, the basically the variable gadgets get connected to the clause gadgets in this manner.

(Refer Slide Time: 46:10)

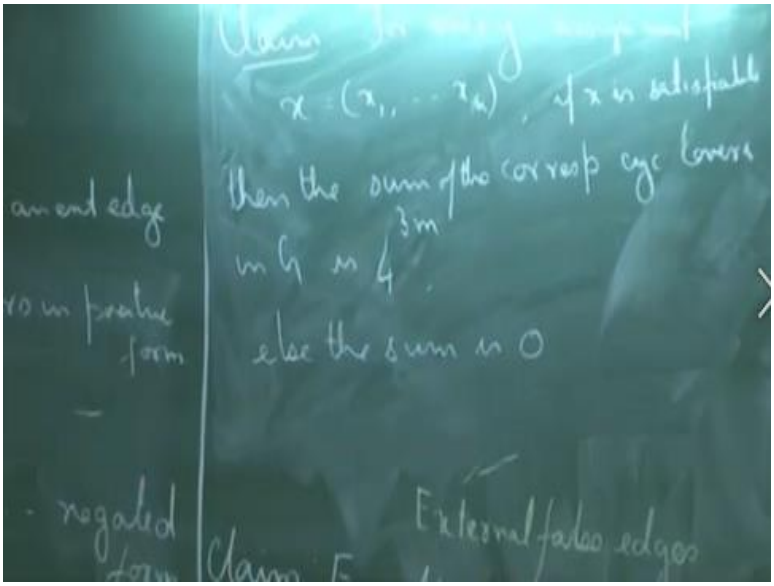


So, let us x i be a literal or let us x i be a variable in class C j. So, then connected true external edge of x i gadget with an external edge of C j gadget if x i appears in the non-negated form so if x i appears in positive form. Otherwise this is one the second is so else, connect a false external edge of x i with an external edge of C j gadget if x i appears in the negated form. So, if you look at a clause C j, so a variable can appear in two ways, it can appear either as x i or it can appear as x i bar.

So, if the variable appears as x i, so then you connect an external edge of that variable to an external edge of the clause in which it appears. Otherwise you connect so if it appears in it is negative form you have connect the false external edges and this connection you do using XOR gadgets, so using the XOR gadget. If you have a variable which lets say appear in it is positive form in three clauses then you have three external edges.

And let us see if it appears in five clauses in it is negated form you have five external edges. So, it is basically at most m each of them because they are at most m clauses.

(Refer Slide Time: 49:48)



So, the claim is that to complete the proof here is this, for every assignment x so x is an assignment of values to the variables if x is satisfiable then there exist or then the sum of the corresponding cycle covers in G is 3 to the 4 m else the sum is 0. So, let us stop here we will sorry 4 to the power 3 m. So, we will see why this is true next time. But once we get this the final theorem is obvious.

Because for every satisfying assignment you are adding so much and for every non satisfying assignment you are adding 0 so therefore the total is this time the number of satisfying assignments. So, this is also not very difficult to see once we I mean have these properties analyzed. So, just one point before we go so, I think that there might be a problem with this gadget.

So, what so the book basically has this gadget I mean if you look at chapter 17, they have these three gadgets and what they claim is that for every proper subset of the three external edges there is a unique cycle cover in this gadget which covers the four vertices, but that is not quite true. So, in fact if you look at just this edge so there is one cycle cover which takes this green edge, this edge, this edge and comes here.

There is also another which takes this green edge then takes this edge, this edge and then comes back here. So, that is not quite true what the books is but I could not figure it out completely but

again I am sure that an alternate gadget will be easy to construct which has this property with a unique cycle cover corresponding to all proper subsets. But anyway, so I will look at that once more and talk about this next class.