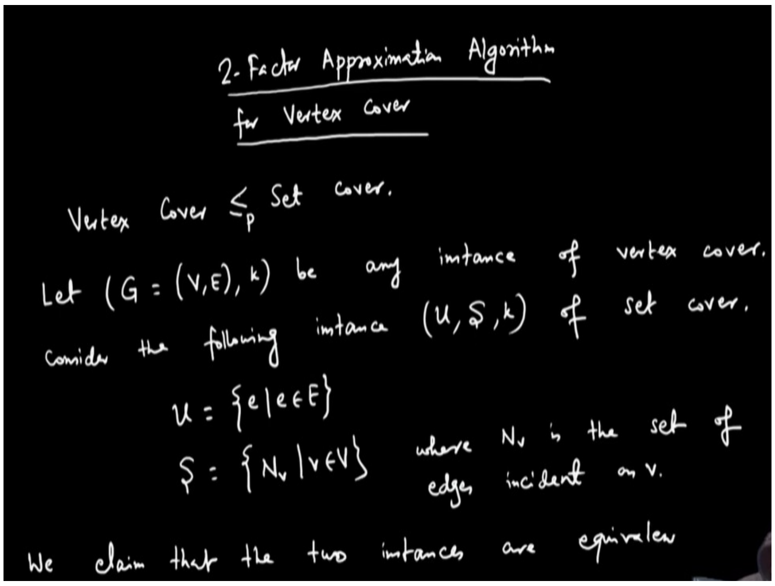
Selected Topics in Algorithms
Prof. Palash Dey
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Module No # 11
Lecture No # 53
Vertex Cover Using Reduction to Set Cover

Thank you welcome so in the last class we have seen an f factor approximation algorithm for set cover and we have also seen the approximation preserving reduction from vertex cover to set cover. So that proof we did not cover in the last plus so let us continue that.

(Refer Slide Time: 00:49)



2-Factor Approximation Algorithm for Vertex Cover

Vertex Cover $\leq_p$ Set Cover.

Let $(G=(V,E),k)$ be any instance of vertex cover.
Consider the following instance (U, S, k) of set cover.

$U = \{e \mid e \in E\}$

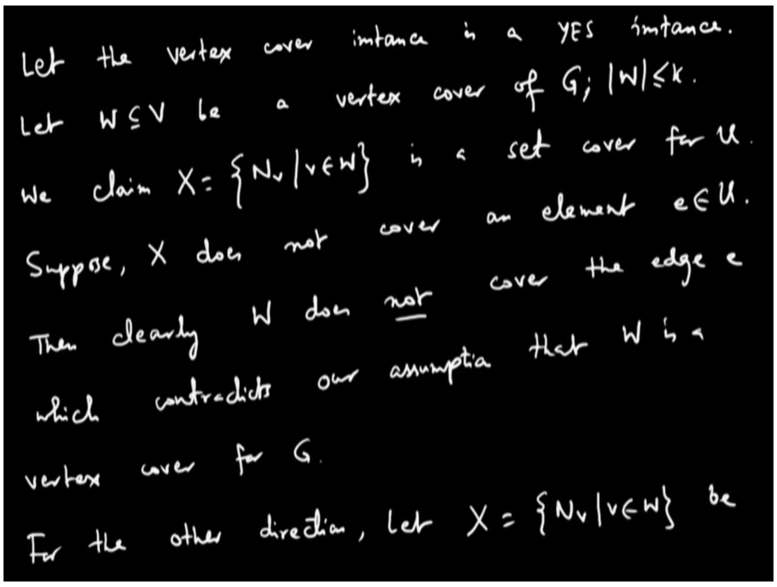
$S = \{N_v \mid v \in V\}$ where N_v is the set of edges incident on v.

We claim that the two instances are equivalen

Today's topic is a 2 factor approximation algorithm for vertex cover so for that we reduced from vertex covered to set cover and that, is a polynomial time approximation preserving reduction this is p set cover so what is the reduction? So let $(G=(V,E),k)$ be any instance of vertex covered. Considered the following instance U, S and k of set cover what is the instance? U is the set of edges for every age I have an element in the universe and s is for every vertex the neighbourhood of that vertex where $N(v)$ is the, set of edges incident on v.

We claim that the 2 instances are equivalent how do you show that?

(Refer Slide Time: 03:54)

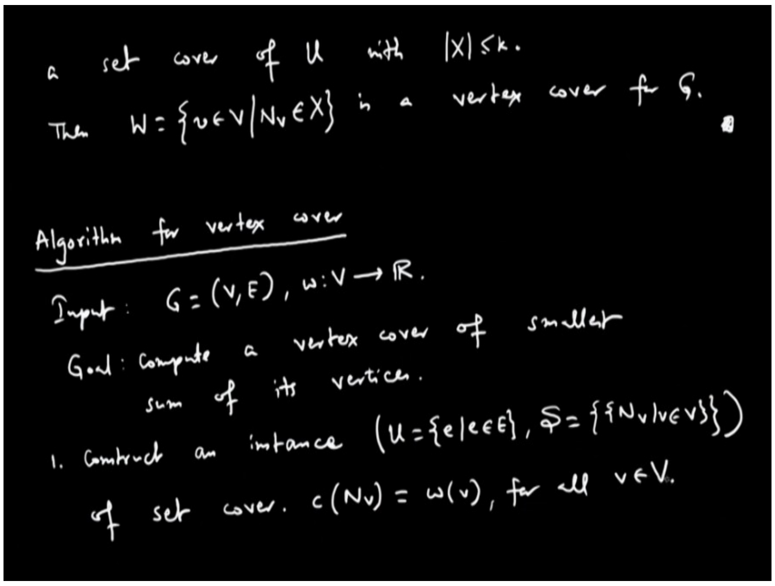


Let the vertex cover instance is a YES instance.
Let $W \subseteq V$ be a vertex cover of G; $|W| \leq k$.
We claim $X = \{N_v \mid v \in W\}$ is a set cover for U.
Suppose, X does not cover an element $e \in U$.
Then clearly W does not cover the edge e
which contradicts our assumption that W is a
vertex cover for G.
For the other direction, let $X = \{N_v \mid v \in W\}$ be

So let the vertex cover instance is a YES instance let W subset of V be a text covered of g. We claim that let us call this set X equal to neighbourhood of v where v varies in W is a set cover for U true by contradiction. So suppose not suppose x does not cover an element $e \in U$ then clearly W does not, cover the edge e which contradicts. Our assumption that W is a vertex covered for g for the other direction it is exactly similar.

But it is important to do it for equivalence it is an if and only statement one instance is true if and only the other instance is true. For the other direction let $X = N(v)$ here also we have cardinality w is less than equal to k then we have cardinality x is, also less than equal to k that is what a yes instance of vertex cover means.

(Refer Slide Time: 07:25)



a set cover of U with $|X| \leq k$.
Then $W = \{v \in V \mid N_v \in X\}$ is a vertex cover for G.

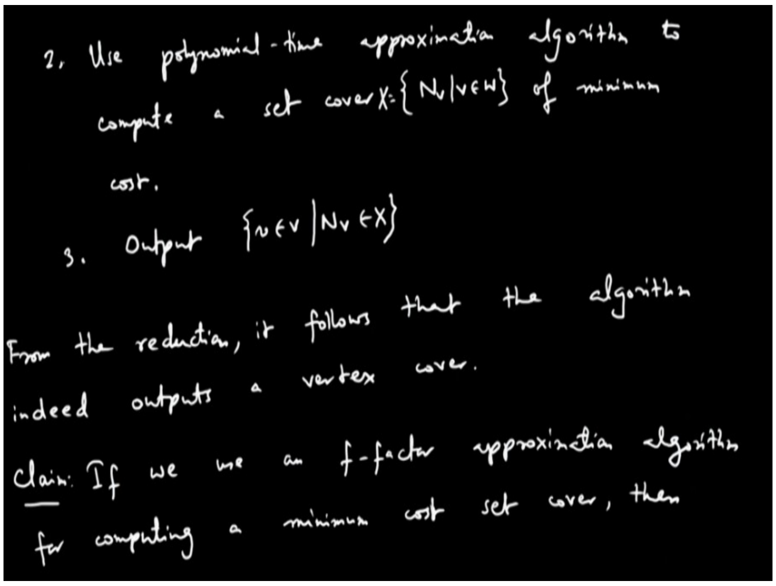
Algorithm for vertex cover

Input: $G = (V, E)$, $w : V \longrightarrow \mathbb{R}$.
Goal: Compute a vertex cover of smaller sum of its vertices.
1. Construct an instance $(U = \{e \mid e \in E\}, \mathcal{S} = \{\{N_v \mid v \in V\}\})$ of set cover. $c(N_v) = w(v)$, for all $v \in V$.

So v in w we are set cover of u with cardinality X is less than equal to k then W = $v \in V$ such that $N_v \in X$ is a vertex cover for G. And again if it is not a vertex cover then some h is missed some it is not covered by W but then that edge is also that the corresponding element is also not covered by X. So and, now we observe so this proves the equivalence now the 2 factor algorithm for vertex covered so input is some this graph $G=(V,E)$ and we need to compute the compute the smallest vertex cover also there could be weight so it could be weighted.

So W maps from V to real numbers the goal is to compute a vertex cover of smallest sum of its vertices. So step 1 let U equal to or not let, construct and instance u = e in E and script S = N_v in V construct an instance this of set cover define the cost of each set c of s c of N_v is set is look like N v c of n v is w of v for all v in V.

(Refer Slide Time: 11:35)



2. Use polynomial-time approximation algorithm to compute a set cover $X = \{N_v \mid v \in W\}$ of minimum cost.

3. Output $\{v \in V \mid N_v \in X\}$

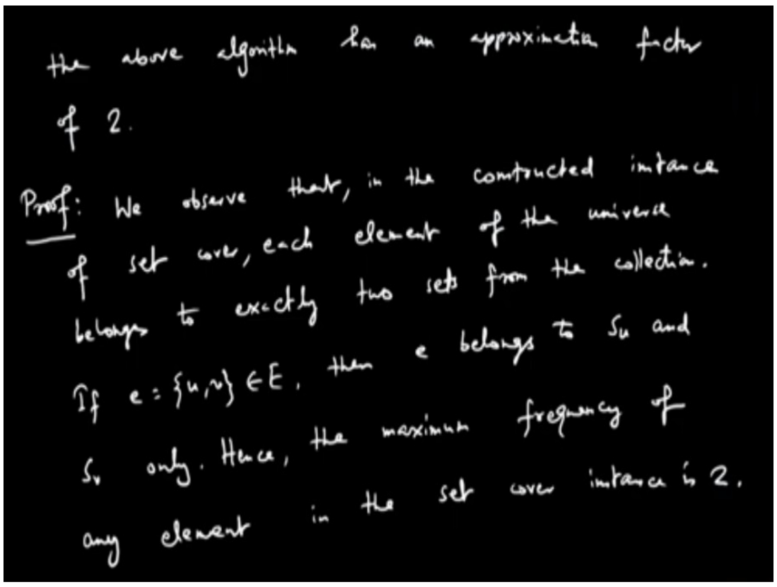
From the reduction, it follows that the algorithm indeed outputs a vertex cover.

Claim: If we use an f-factor approximation algorithm for computing a minimum cost set cover, then

Second step use approximation algorithm use polynomial time approximation algorithm to compute set covered let us call it $N_v \in W$ to compute a set, cover of minimum cost. Then simply output those set of vertices where in v let us call this set cover X, $N_v \in X$ output this as the vertex covered. So from equivalence from the reduction it follows that the algorithm indeed outputs vertex cover.

Now here if we use and if factor where f; is the frequency of any element at maximum frequency of any element if we use an; if factor approximation algorithm for computing minimum cost set cover.

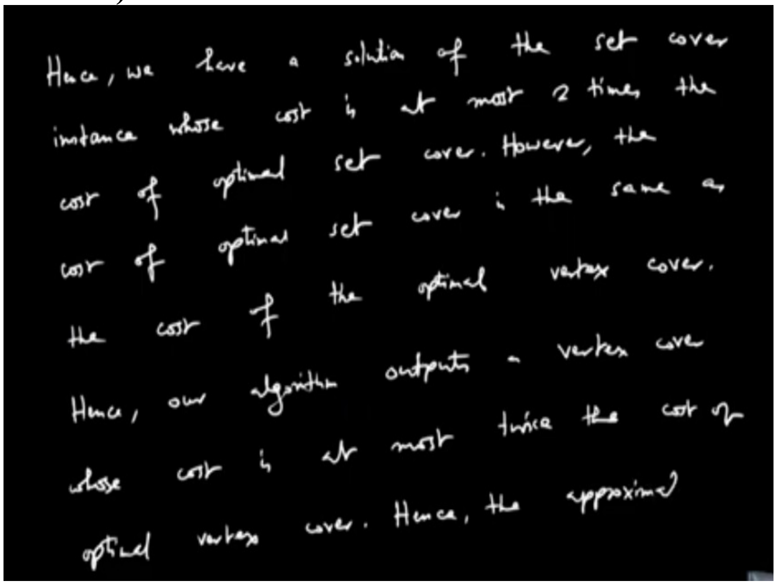
(Refer Slide Time: 15:18)


the above algorithm has an approximation factor
of 2.

Proof: We observe that, in the constructed instance
of set cover, each element of the universe
belongs to exactly two sets from the collection.
If $e = \{u, v\} \in E$, then e belongs to S_u and
S_v only. Hence, the maximum frequency of
any element in the set cover instance is 2.

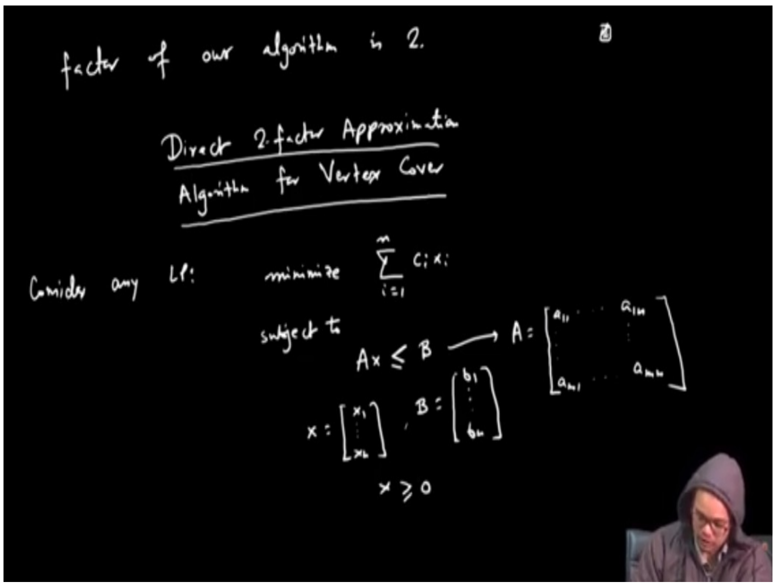
Then the above algorithm has an approximation factor of 2. That is the clip proof so what is the value of f? First we observe that in the constructed instance of set cover each element of the universe belongs to exactly 2 sets from, the collection that is if e equal to say u, v is an edge in g then e belongs to S_u and S_v only. Hence the maximum frequency of any element in the set cover instance is 2.

(Refer Slide Time: 18:34)


Hence, we have a solution of the set cover
instance whose cost is at most 2 times the
cost of optimal set cover. However, the
cost of optimal set cover is the same as
the cost of the optimal vertex cover.
Hence, our algorithm outputs a vertex cover
whose cost is at most twice the cost of
optimal vertex cover. Hence, the approximation

Now because the approximation factor if is f of our set cover approximation algorithm hence we have a solution of the set cover instance whose cost is at most 2 times the, cost of optimal set cover. But we have observed that however the cost of optimal set cover is the same as the cost of the optimal vertex covered. Hence our algorithm outputs are vertex covered whose cost is at most twice the cost of optimal vertex cover. Hence the approximation factor of our algorithm is 2.

(Refer Slide Time: 22:08)



factor of our algorithm is 2.

Direct 2-factor Approximation Algorithm for Vertex Cover

Consider any LP: minimize $\sum_{i=1}^{m} c_i x_i$

subject to

$Ax \leq B \longrightarrow A = \begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mm} \end{bmatrix}$

$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad B = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$

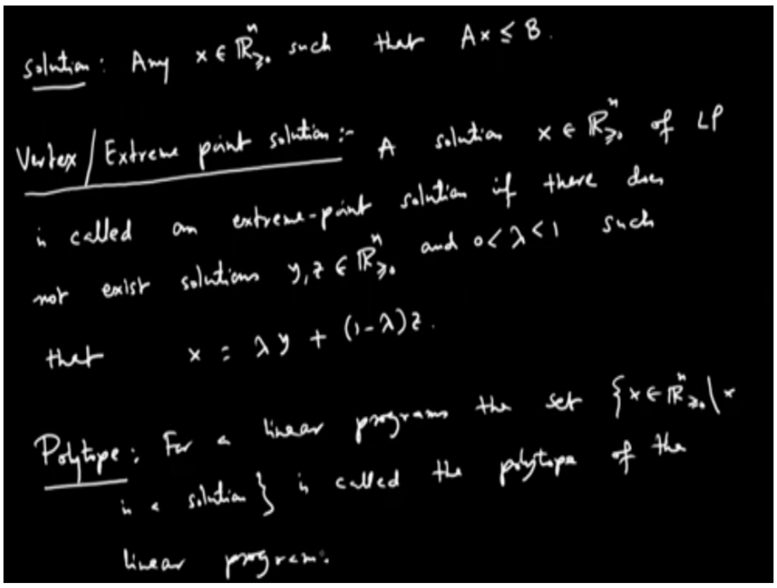
$x \geq 0$

So next what we, will show is another direct rounding approach also very simple now direct rounding approach using LP rounding for 2 factor approximation algorithm for vertex cover this is also simple but here we need we are we are reducing it to set cover. So next is a direct and there we will see some more extra structure of the fractional solution of vertex cover. So direct 2 factor approximation algorithm for vertex covered for that let me use this as an excuse to introduce couple of couple of notions.

So let consider any arbitrary LP considers any linear program how does it look like? Minimize in the standard form minimize $\sum_{i=1}^{n} c_i x_i$ subject to some linear constraints. So what are the linear constraints if you recall it can be written as Ax is less than equal to b how? This, is basically in the matrix notation where A is this matrix, x and b are column vector.

So $x=(x_1 \ldots x_n)$ and $b=(b_1 \ldots b_n)$ and this less than equal to means component wise it should be less each component it should be less. And of course we have the constraints that x is greater than equal to 0.

(Refer Slide Time: 25:49)



<u>Solution</u>: Any $x \in \mathbb{R}^n_{\geq 0}$ such that $Ax \leq B$.

<u>Vertex / Extreme point solution</u> :- A solution $x \in \mathbb{R}^n_{\geq 0}$ of LP is called an extreme-point solution if there does not exist solutions $y, z \in \mathbb{R}^n_{\geq 0}$ and $0 < \lambda < 1$ such that $x = \lambda y + (1-\lambda) z$.

<u>Polytope</u>: For a linear program the set $\{x \in \mathbb{R}^n_{\geq 0} \mid x$ is a solution$\}$ is called the polytope of the linear program.

So some notations of solution is any $x \in \mathbb{R}^n$ such that is greater than equal to 0 because it satisfies it needs to satisfy these constraints and x is less than equal to b. So that is called a solution the second one is called polytope for it is called a vertex or vertex it has couple of names vertex or extreme point solution. A solution x in solution x of LP is called an extreme point solution if x cannot be written as a convex combination, of 2 other solutions.

If they are does not exist solutions y and z in $\mathbb{R}^n$ and lambda in between 0 and 1 such that x = lambda times $y+(1-\lambda)z$. So this means cannot be written in such a solution and there is another concept called polytope. So for a linear program the set of solutions the set x in such that x is solution to the linear program is called the polytope of the, linear program. So the next class we will see some interesting properties of the polytope of vertex cover LP and using that we will design a 2 factor approximation algorithm for vertex cover thank you.