

## **NP-COMPLETENESS II**

THURSDAY NOVEMBER 3

Definition: Language B is NP-complete if:

1.  $B \in \text{NP}$
2. Every A in NP is poly-time reducible to B (i.e. B is NP-hard)

Theorem (Cook-Levin): SAT is NP-complete  
Corollary:  $\text{SAT} \in \text{P}$  if and only if  $\text{P} = \text{NP}$

Today we will show that there are many NP-complete languages

3SAT = {  $\phi$  |  $\phi$  is a satisfiable 3cnf-formula }

A 3cnf-formula is of the form:

$$(x_1 \vee \neg x_2 \vee x_3) \wedge (x_4 \vee x_2 \vee x_5) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$$

clauses

YES  $(x_1 \vee \neg x_2 \vee x_1)$

NO  $(x_3 \vee x_1) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$

NO  $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_4 \vee x_2 \vee x_1) \vee (x_3 \vee x_1 \vee \neg x_1)$

NO  $(x_1 \vee \neg x_2 \vee x_3) \wedge (x_3 \wedge \neg x_2 \wedge \neg x_1)$

Theorem: 3SAT is NP-complete

Proof:

(1) 3SAT  $\in$  NP

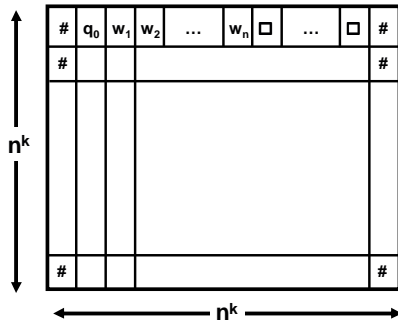
(2) Every language A in NP is polynomial time Reducible to 3SAT

We build a poly-time reduction from A to 3SAT

The reduction turns a string w into a 3-cnf formula  $\phi$  that simulates the NP machine for A on w

Let N be a non-deterministic TM that decides A in time  $n^k$  How do we know N exists?

A tableau for N on w is an  $n^k \times n^k$  table whose rows are the configurations of a branch of the computation of N on input w



A tableau is accepting if any row of the tableau is an accepting configuration

Determining whether N accepts w is equivalent to determining whether an accepting tableau for N on w exists

## VARIABLES

Let  $C = Q \cup \Gamma \cup \{ \# \}$

Each of the  $(n^k)^2$  entries of a tableau is a cell

The cell in row  $i$  and column  $j$  is called  $\text{cell}[i,j]$

For each  $i$  and  $j$  ( $1 \leq i, j \leq n^k$ ) and for each  $s \in C$  we have a variable  $x_{i,j,s}$

These are the variables of  $\phi$  and represent the contents of the cells

If  $x_{i,j,s}$  takes on the value 1, it means that  $\text{cell}[i,j]$  contains  $s$

$$x_{i,j,s} = 1$$

means

$$\text{cell}[i, j] = s$$

We now design  $\phi$  so that a satisfying assignment to the variables corresponds to an accepting tableau for  $N$  on  $w$

The formula  $\phi$  will be the AND of four parts:

$$\phi = \phi_{\text{cell}} \wedge \phi_{\text{start}} \wedge \phi_{\text{accept}} \wedge \phi_{\text{move}}$$

$\phi_{\text{cell}}$  ensures that for each  $i,j$  exactly one  $x_{i,j,s}$  is true

$\phi_{\text{start}}$  ensures that the first row of the table is the starting configuration of  $N$  on  $w$

$\phi_{\text{accept}}$  ensures that an accepting configuration occurs in the table

$\phi_{\text{move}}$  ensures that every row is a configuration that legally follows from the previous

$$\phi_{\text{cell}} = \bigwedge_{1 \leq i, j \leq n^k} \left[ \left( \bigvee_{s \in C} x_{i,j,s} \right) \wedge \left( \bigwedge_{\substack{s,t \in C \\ s \neq t}} (\neg x_{i,j,s} \vee \neg x_{i,j,t}) \right) \right]$$

This formula is the AND of a bunch of clauses  
Except the clauses may not have 3 literals each

If a clause has less than three variables:

$$(a \vee b) = (a \vee b \vee b)$$

If a clause has more than three variables:

$$(a \vee b \vee c \vee d) = (a \vee b \vee z) \wedge (\neg z \vee c \vee d)$$

$$\begin{aligned}
\phi_{\text{start}} &= X_{1,1,\#} \wedge X_{1,2,q_0} \wedge \\
& X_{1,3,w_1} \wedge X_{1,4,w_2} \wedge \dots \wedge X_{1,n+2,w_n} \wedge \\
& X_{1,n+3,\square} \wedge \dots \wedge X_{1,n^k-1,\square} \wedge X_{1,n^k,\#} \\
&= (X_{1,1,\#} \vee X_{1,1,\#} \vee X_{1,1,\#}) \wedge \\
& (X_{1,2,q_0} \vee X_{1,2,q_0} \vee X_{1,2,q_0}) \\
& \wedge \dots \wedge \\
& (X_{1,n^k,\#} \vee X_{1,n^k,\#} \vee X_{1,n^k,\#})
\end{aligned}$$

$$\phi_{\text{accept}} = \bigvee_{1 \leq i, j \leq n^k} X_{i,j,q_{\text{accept}}}$$

$$\begin{aligned}
&(a_1 \vee a_2 \vee \dots \vee a_t) = \\
&(a_1 \vee a_2 \vee z_1) \wedge (\neg z_1 \vee a_3 \vee z_2) \wedge (\neg z_2 \vee a_4 \vee z_3) \dots
\end{aligned}$$

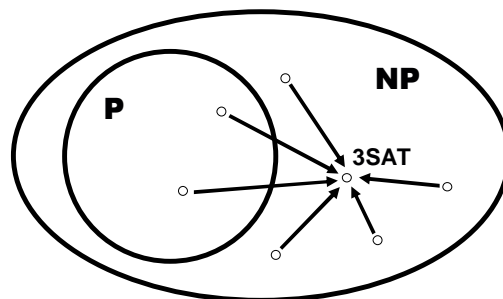
$$\phi_{\text{move}} = \bigwedge_{1 \leq i, j \leq n^k} (\text{the } (i, j) \text{ window is legal})$$

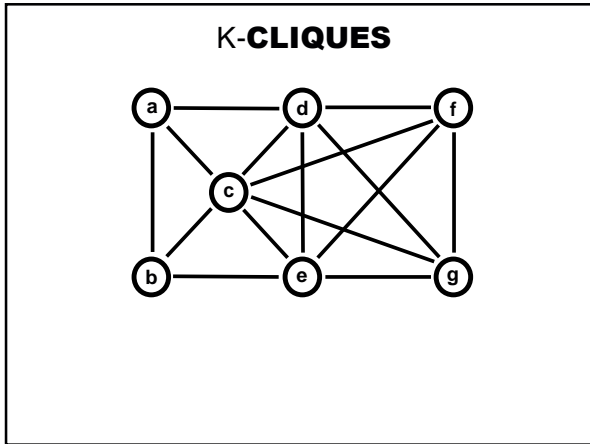
the (i, j) window is legal =

$$\bigvee_{a_1, \dots, a_6} (X_{i-1,j,a_1} \wedge X_{i,j,a_2} \wedge X_{i+1,j,a_3} \wedge X_{i-1,j+1,a_4} \wedge X_{i,j+1,a_5} \wedge X_{i+1,j+1,a_6})$$

is a legal window

### 3SAT is NP-Complete



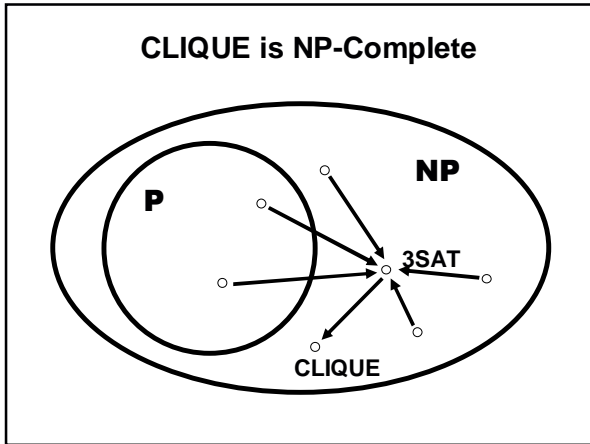


**CLIQUE** = { (G,k) | G is an undirected graph with a k-clique }

**Theorem: CLIQUE is NP-Complete**

(1) **CLIQUE**  $\in$  NP

(2) **3SAT**  $\leq_p$  **CLIQUE**

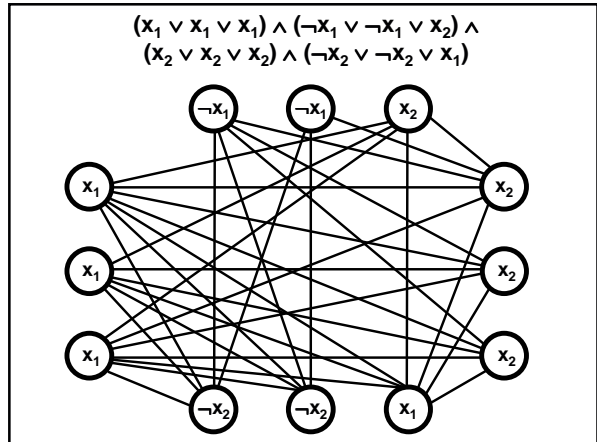
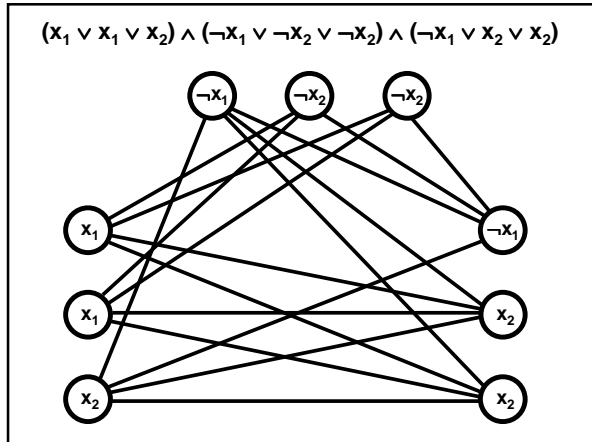


**3SAT**  $\leq_p$  **CLIQUE**

We transform a 3-cnf formula  $\phi$  into (G,k) such that

$\phi \in 3SAT \Leftrightarrow (G,k) \in CLIQUE$

The transformation can be done in time polynomial in the length of  $\phi$



**3SAT  $\leq_p$  CLIQUE**

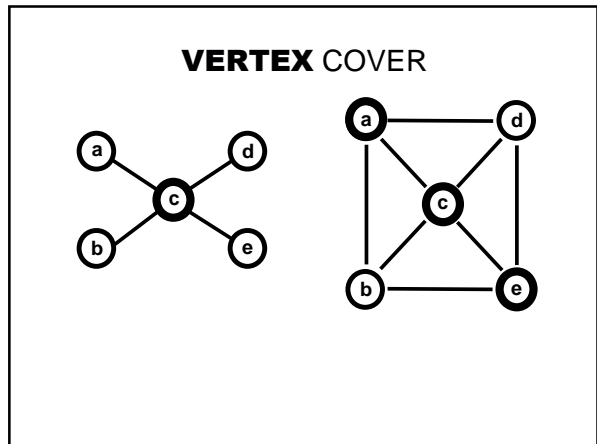
We transform a 3-cnf formula  $\phi$  into  $(G,k)$  such that

$\phi \in 3SAT \Leftrightarrow (G,k) \in CLIQUE$

If  $\phi$  has  $k$  clauses, we create a graph with  $k$  clusters of 3 nodes each. Each cluster corresponds to a clause. Each node in a cluster is labeled with a variable from the clause.

We do not connect any nodes in the same cluster

We connect nodes in different clusters whenever they are not contradictory



VERTEX-COVER = { (G,k) | G is an undirected graph with a k-node vertex cover }

Theorem: VERTEX-COVER is NP-Complete

(1) VERTEX-COVER ∈ NP

(2) 3SAT ≤<sub>p</sub> VERTEX-COVER

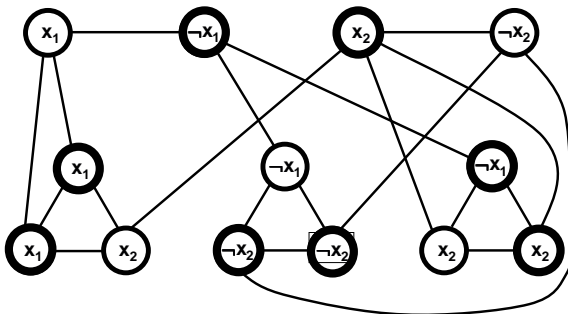
**3SAT** ≤<sub>p</sub> VERTEX-COVER

We transform a 3-cnf formula  $\phi$  into (G,k) such that

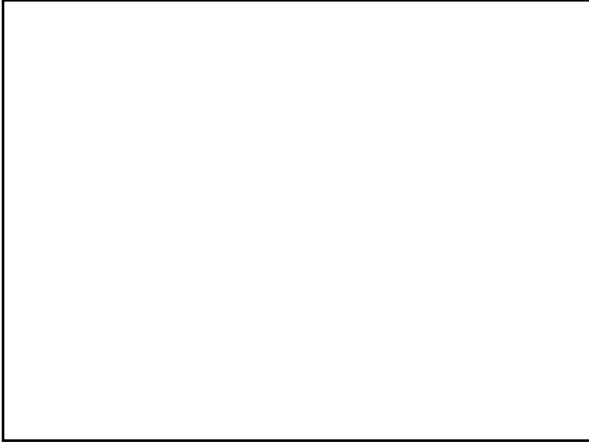
$\phi \in 3SAT \Leftrightarrow (G,k) \in VERTEX-COVER$

The transformation can be done in time polynomial in the length of  $\phi$

$(x_1 \vee x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_2)$



$(x_1 \vee x_1 \vee x_1) \wedge (\neg x_1 \vee \neg x_1 \vee x_2) \wedge (x_2 \vee x_2 \vee x_2) \wedge (\neg x_2 \vee \neg x_2 \vee x_1)$



**WWW.FLAC.WS**