CS 154

Cook-Levin Continued, More NP-Complete Problems

Theorem (Cook-Levin): 3SAT is NP-complete

Proof Idea:

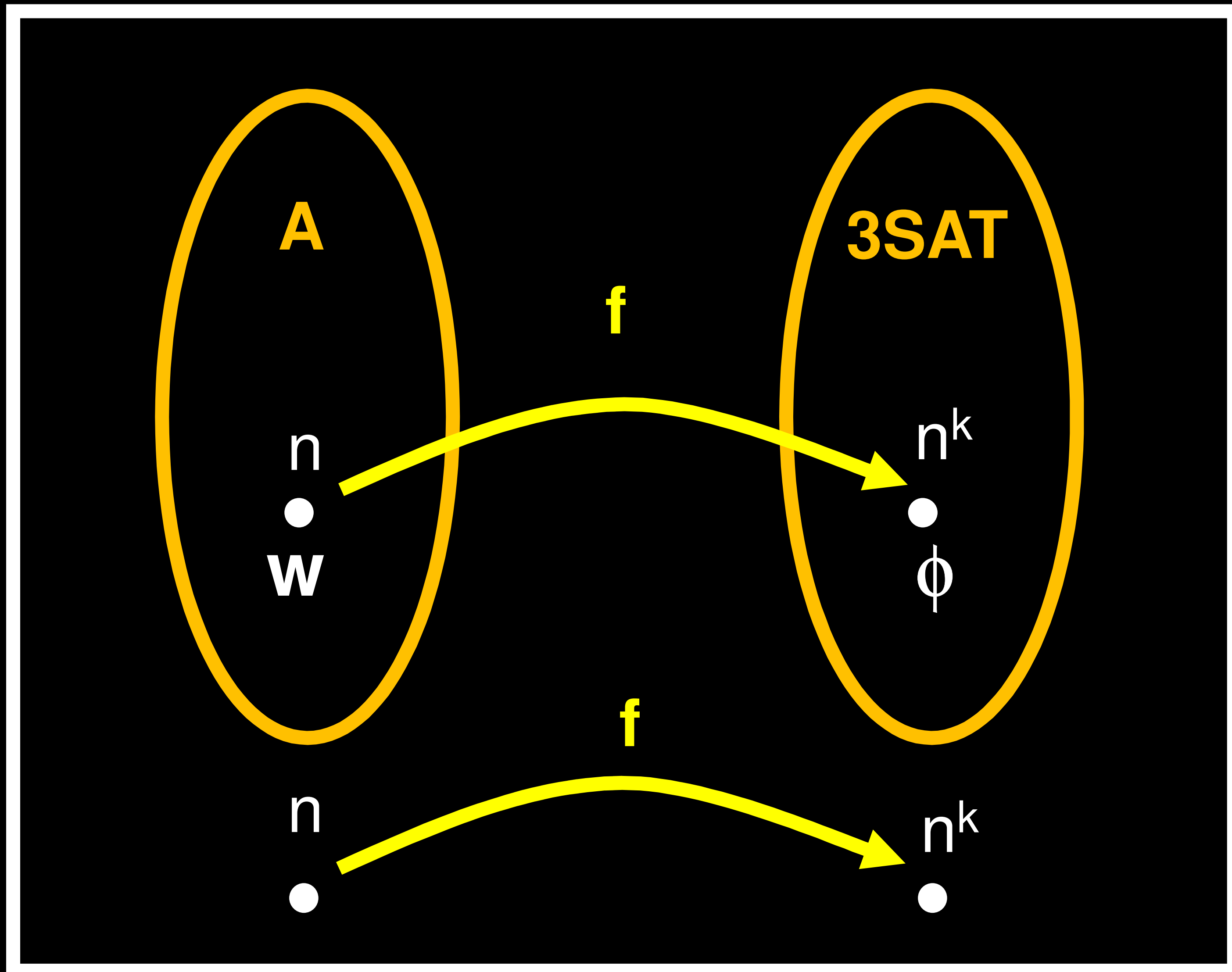
(1) 3SAT $\in$ NP (already done)

(2) Every language A in NP is polynomial time reducible to 3SAT (this is the challenge)

Poly-time reduction which converts a string w into a 3cnf formula ϕ such that $w \in A$ iff $\phi \in$ 3SAT

For any $A \in$ NP, let N be a nondeterministic TM deciding A in n^k time

ϕ will simulate N on w

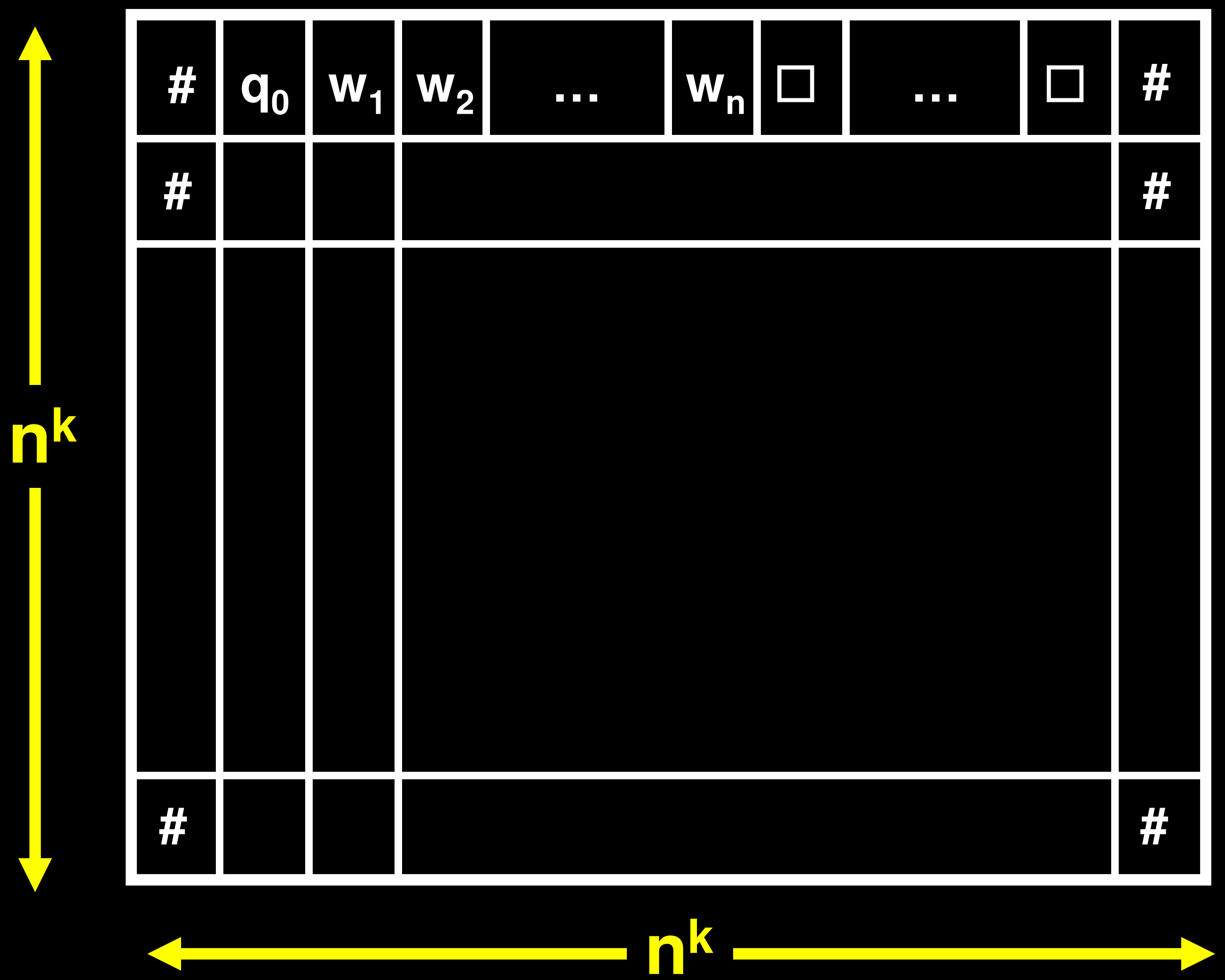


f turns any string w into a 3-cnf formula ϕ such that

$w \in A \Leftrightarrow \phi$ is satisfiable

ϕ will simulate an NP machine N on w, where A = L(N)

Let $L(N) \in$ NTIME(n^k). A **tableau for N on w** is an $n^k \times n^k$ table whose rows are the configurations of *some* possible computation history of N on w



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

N accepts w if and only if
there is an accepting tableau for N on w

Given w, we'll construct a 3cnf formula ϕ with $O(|w|^k)$ clauses, describing logical constraints that every accepting tableau for N on w must satisfy

The 3cnf formula ϕ will be satisfiable *if and only if* there is an accepting tableau for N on w

Variables of formula ϕ will *encode* a tableau

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

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

cell[i,j] = value of the cell at row i and column j
= the jth symbol in the ith configuration

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

Total number of variables = $|C|n^{2k}$, which is $O(n^{2k})$

These $x_{i,j,s}$ are the variables of ϕ and represent the contents of the cells

We will have: for all i,j,s, $x_{i,j,s} = 1 \Leftrightarrow$ cell[i,j] = s

Idea: Make ϕ so that every ***satisfying assignment*** to the variables $x_{i,j,s}$ corresponds to an ***accepting tableau*** for **N** on **w** (an assignment to all **cell[i,j]'s of the tableau**)

The formula ϕ will be the **AND** of four CNF formulas:

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

ϕ_{cell} : for all i, j, there is a unique $s \in C$ with $x_{i,j,s} = 1$

ϕ_{start} : the first row of the table equals the *start* configuration of **N** on **w**

ϕ_{accept} : the last row of the table has an accept state

ϕ_{move} : every row is a configuration that yields the configuration on the next row

ϕ_{cell} : for all i, j, there is a unique $s \in C$ with $x_{i,j,s} = 1$

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

for all i,j

at least one $x_{i,j,s}$ is set to 1

at most one $x_{i,j,s}$ is set to 1

ϕ_{start} : the first row of the table equals the *start* configuration of **N** on **w**

$$\phi_{start} = \quad x_{1,1,\#} \wedge x_{1,2,q_0} \wedge$$
$$x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge$$
$$x_{1,n+3,\square} \wedge \cdots \wedge x_{1,n^k-1,\square} \wedge x_{1,n^k,\#}$$

#	q_0	w_1	w_2	…	w_n	$\square$	…	$\square$	#
#									#

ϕ_{accept} **: the last row of the table has an accept state**

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

ϕ_{move} : every row is a configuration that yields the configuration on the next row

Key Question: If one row yields the next row, how many cells can be different between the two rows?

Answer: AT MOST THREE CELLS!

#	b	a	a	q_1	b	c	b	#
#	b	a	q_2	a	c	c	b	#

ϕ_{move} : every row is a configuration that yields the configuration on the next row

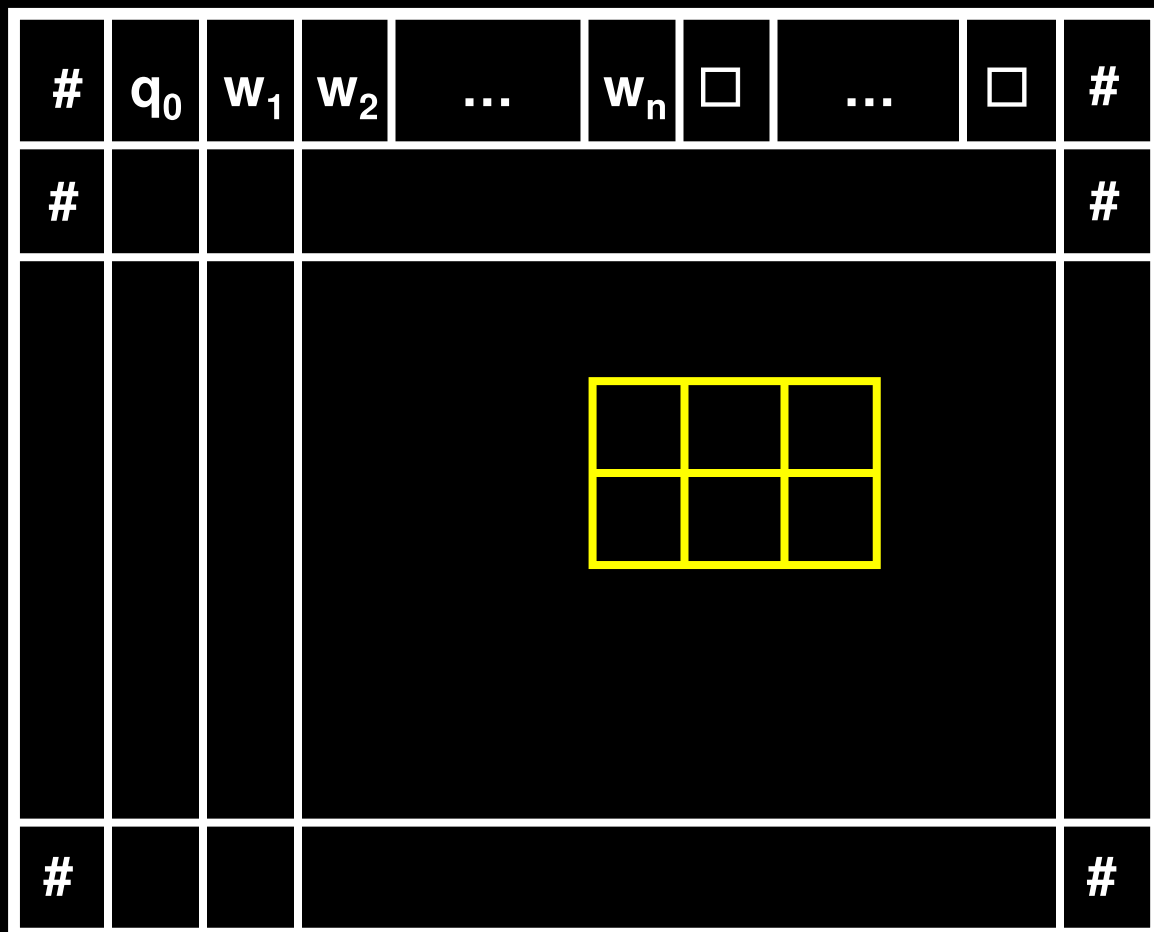
Key Question: If one row yields the next row, how many cells can be different between the two rows?

Answer: AT MOST THREE CELLS!

#	b	a	a	q_1	b	c	b	#
#	b	a	q_2	a	c	c	b	#

ϕ_{move} : every row is a configuration that yields the configuration on the next row

Idea: check that every 2×3 "window" of cells is legal (consistent with the transition function of N)



If $\delta(q_1,a) = \{(q_1,b,R)\}$ and $\delta(q_1,b) = \{(q_2,c,L), (q_2,a,R)\}$ which of the following windows are legal?

a	q_1	b
q_2	a	c

a	q_1	b
q_1	a	a

a	a	q_1
a	a	b

#	b	a
#	b	a

a	b	a
a	b	q_2

b	q_1	b
q_2	b	q_2

a	b	a
a	a	a

a	q_1	b
a	a	q_2

b	b	b
c	b	b

Key Lemma:

IF Every window of the tableau is legal, and
The 1st row is the start configuration of N on w

THEN for all $i = 1,\ldots,n^k - 1$, the ith row of the tableau is a configuration which yields the (i+1)th row.

Proof Sketch: (Strong) induction on i.

The 1st row is a configuration. If it *didn't* yield the 2nd row, there's a 2 x 3 "illegal" window on 1st and 2nd rows

Assume rows 1,…,L are all configurations which yield the next row, and assume every window is legal.

If row L+1 did *not* yield row L+2, then there's a 2 x 3 window along those two rows which is "illegal"

The (i, j) window of a tableau is the tuple $(a_1,\ldots, a_6) \in C^6$ such that

	col. j	col. j+1	col. j+2
row i	a_1	a_2	a_3
row i+1	a_4	a_5	a_6

ϕ_{move} : every row is a configuration that legally follows from the previous configuration

$$\phi_{move} = \bigwedge_{\substack{1 \leq i \leq n^k - 1 \\ 1 \leq j \leq n^k - 2}} (\text{ the } (i, j) \text{ window is legal })$$

$$(\text{the } (i, j) \text{ window is legal}) =$$

$$\bigvee_{\substack{(a_1, \ldots, a_6) \\ \text{is a legal window}}} (x_{i,j,a_1} \wedge x_{i,j+1,a_2} \wedge x_{i,j+2,a_3} \wedge x_{i+1,j,a_4} \wedge x_{i+1,j+1,a_5} \wedge x_{i+1,j+2,a_6})$$

$$\equiv \bigwedge_{\substack{(a_1, \ldots, a_6) \\ \text{is NOT a legal window}}} (\overline{x}_{i,j,a_1} \vee \overline{x}_{i,j+1,a_2} \vee \overline{x}_{i,j+2,a_3} \vee \overline{x}_{i+1,j,a_4} \vee \overline{x}_{i+1,j+1,a_5} \vee \overline{x}_{i+1,j+2,a_6})$$

How do we get 3SAT?

We got a CNF formula, but not a 3CNF…
how do we convert the CNF into a 3CNF formula?

A nice trick to “shorten” clauses:

$(a_1 \vee a_2 \vee \ldots \vee a_t)$ is equivalent to

$$(a_1 \vee a_2 \vee z_1) \wedge (\neg z_1 \vee a_3 \vee z_2) \wedge (\neg z_2 \vee a_4 \vee z_3) \ldots \wedge (\neg z_{t-3} \vee a_{t-1} \vee a_t)$$

where z_i are new variables

What's the total length of ϕ?

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

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

$O(n^{2k})$ clauses

$$\phi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\square} \wedge \cdots \wedge x_{1,n^k-1,\square} \wedge x_{1,n^k,\#}$$

$O(n^k)$ clauses

$$\phi_{accept} = \bigvee_{1 \le j \le n^k} x_{n^k, j, q_{accept}}$$

$(a_1 \vee a_2 \vee \ldots \vee a_t)$ is equivalent to

$(a_1 \vee a_2 \vee z_1) \wedge (\neg z_1 \vee a_3 \vee z_2) \wedge (\neg z_2 \vee a_4 \vee z_3) \ldots$
yields O(t) new 3cnf clauses

$O(n^k)$ clauses

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

the (i, j) window is legal =

$$\equiv \bigwedge_{\substack{(a_1, \ldots, a_6) \\ \text{ISN'T a legal window}}} (\overline{x}_{i,j,a_1} \lor \overline{x}_{i,j+1,a_2} \lor \overline{x}_{i,j+2,a_3} \lor \overline{x}_{i+1,j,a_4} \lor \overline{x}_{i+1,j+1,a_5} \lor \overline{x}_{i+1,j+2,a_6})$$

$O(n^{2k})$ clauses

Summary. We wanted to prove:
Every A in NP has a polynomial time reduction to 3SAT

For every A in NP, A is decided by some nondeterministic n^k time Turing machine N

We gave a generic way to reduce (N, w) to a 3CNF formula ϕ of $O(|w|^{2k})$ clauses such that

satisfying assignments to the variables of ϕ
directly correspond to
accepting computation histories of N on w

The formula ϕ is the AND of four 3CNF formulas:

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

Reading Assignment

Read Luca Trevisan's notes for an alternative proof of the Cook-Levin Theorem!

Sketch:

1. **Define CIRCUIT-SAT:** ***Given a logical circuit C(y), is there an input a such that C(a)=1?***
2. **Show that CIRCUIT-SAT is NP-hard:**
   The n^k x n^k tableau for N on w can be simulated using a logical circuit of $O(n^{2k})$ gates
3. **Reduce CIRCUIT-SAT to 3SAT in polytime**
4. **Conclude 3SAT is also NP-hard**

Theorem (Cook-Levin):

SAT and 3SAT are NP-complete

Corollary: **SAT $\in$ P if and only if P = NP**

Is 3SAT solvable in O(n) time on a multitape TM?

Are there logic circuits of size 6n for 3SAT?

If yes, then not only is P=NP,
but there would be a "dream machine" that could crank out short proofs of theorems,
quickly optimize all aspects of life...
recognizing quality work is all you need to produce

THESE ARE OPEN QUESTIONS!

There are thousands of
NP-complete problems

Your favorite topic certainly has an
NP-complete problem somewhere in it

Even the other sciences are not safe:
biology, chemistry, physics have
NP-complete problems too!

Theorem (Cook-Levin):
SAT and 3SAT are NP-complete

Corollary: SAT $\in$ P if and only if P = NP

Given a favorite problem $\Pi \in$ NP,
how can we prove it is NP-hard?

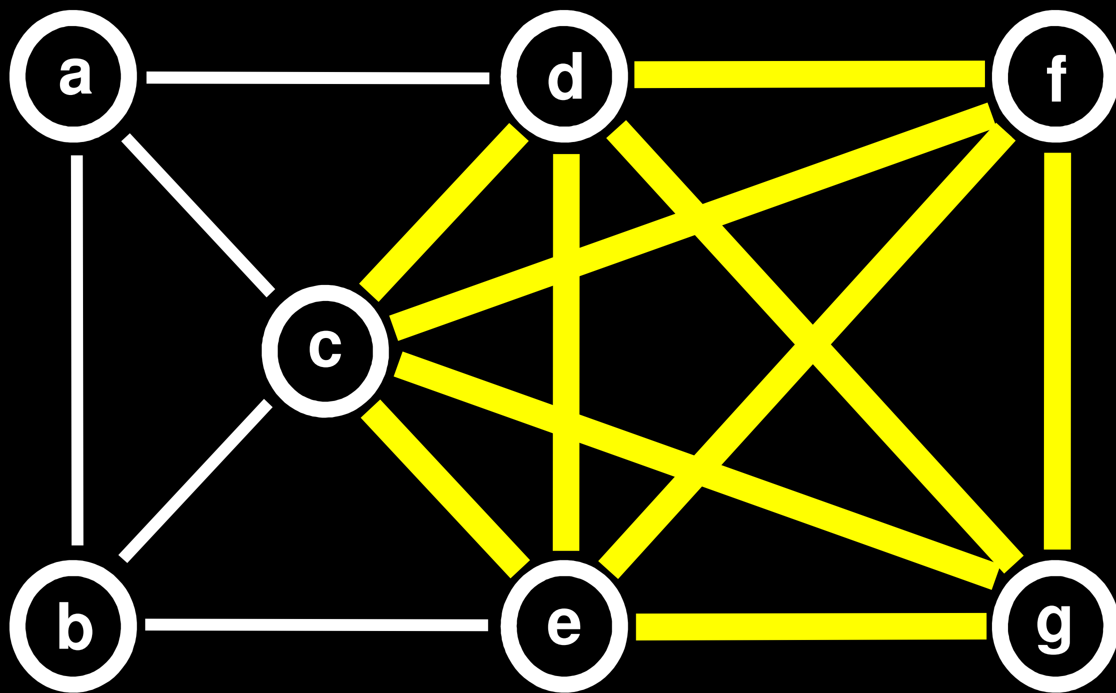
Recipe:

1. Take a problem Σ that you know to be NP-hard (3-SAT)
2. Prove that $\Sigma \leq_P \Pi$

Then for all A $\in$ NP, A $\leq_P \Sigma$ and $\Sigma \leq_P \Pi$

We conclude that A $\leq_P \Pi$, and Π is NP-hard

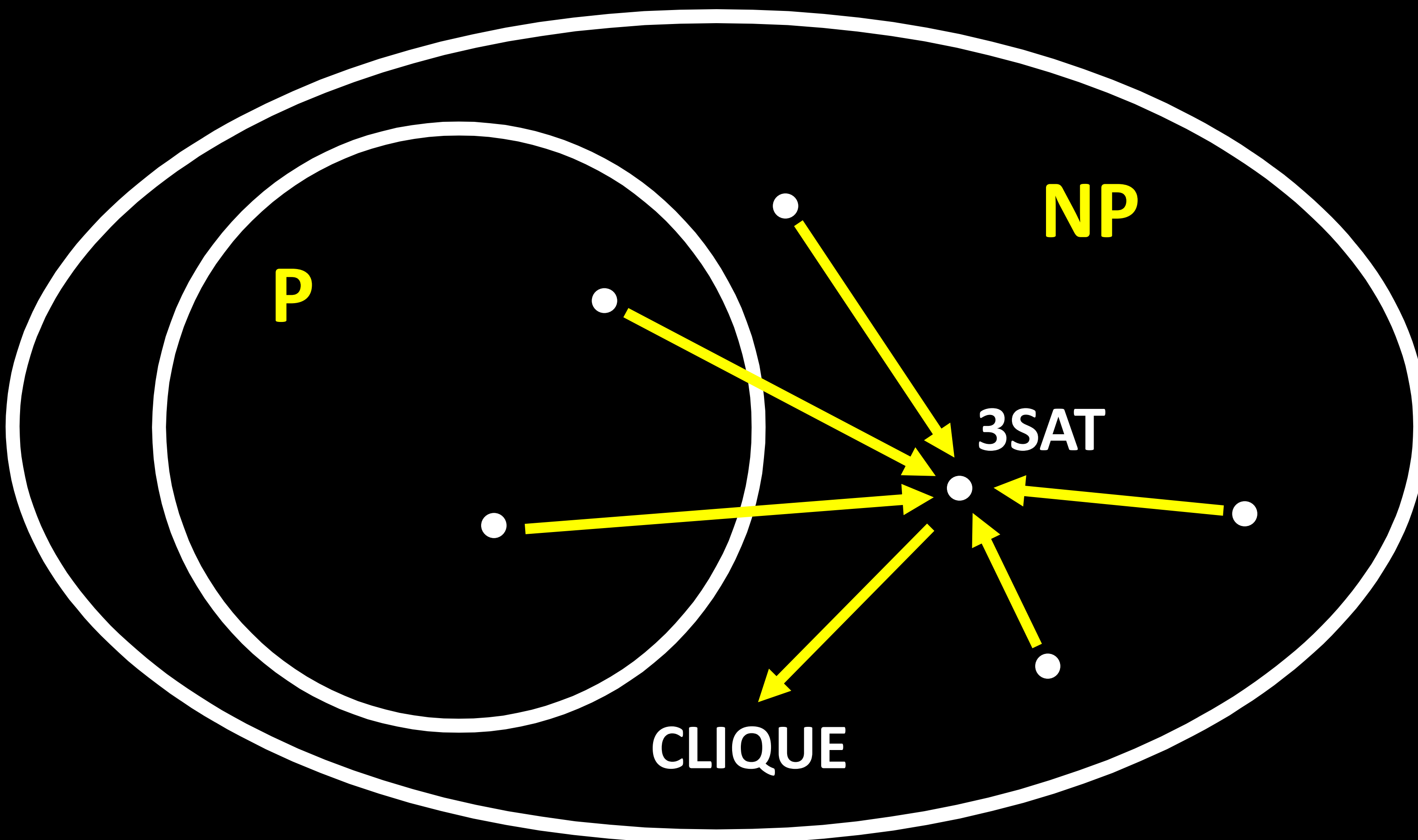
The Clique Problem



Given a graph G and positive k, does G contain a complete subgraph on k nodes?

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

Theorem: CLIQUE is NP-Complete



$3SAT \leq_P$ CLIQUE

Transform a 3-cnf formula ϕ into (G,k) such that

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

Want transformation that can be done in time that is **polynomial in the length of ϕ**

How can we encode
a *logic* problem as a *graph* problem?

$3SAT \leq_P CLIQUE$

We transform a 3-cnf formula ϕ into (G,k) such that

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

Let m be the number of clauses of ϕ. Set k=m.
Make a graph G with m *groups* of 3 nodes each.

Group *i* corresponds to a clause C_i of ϕ

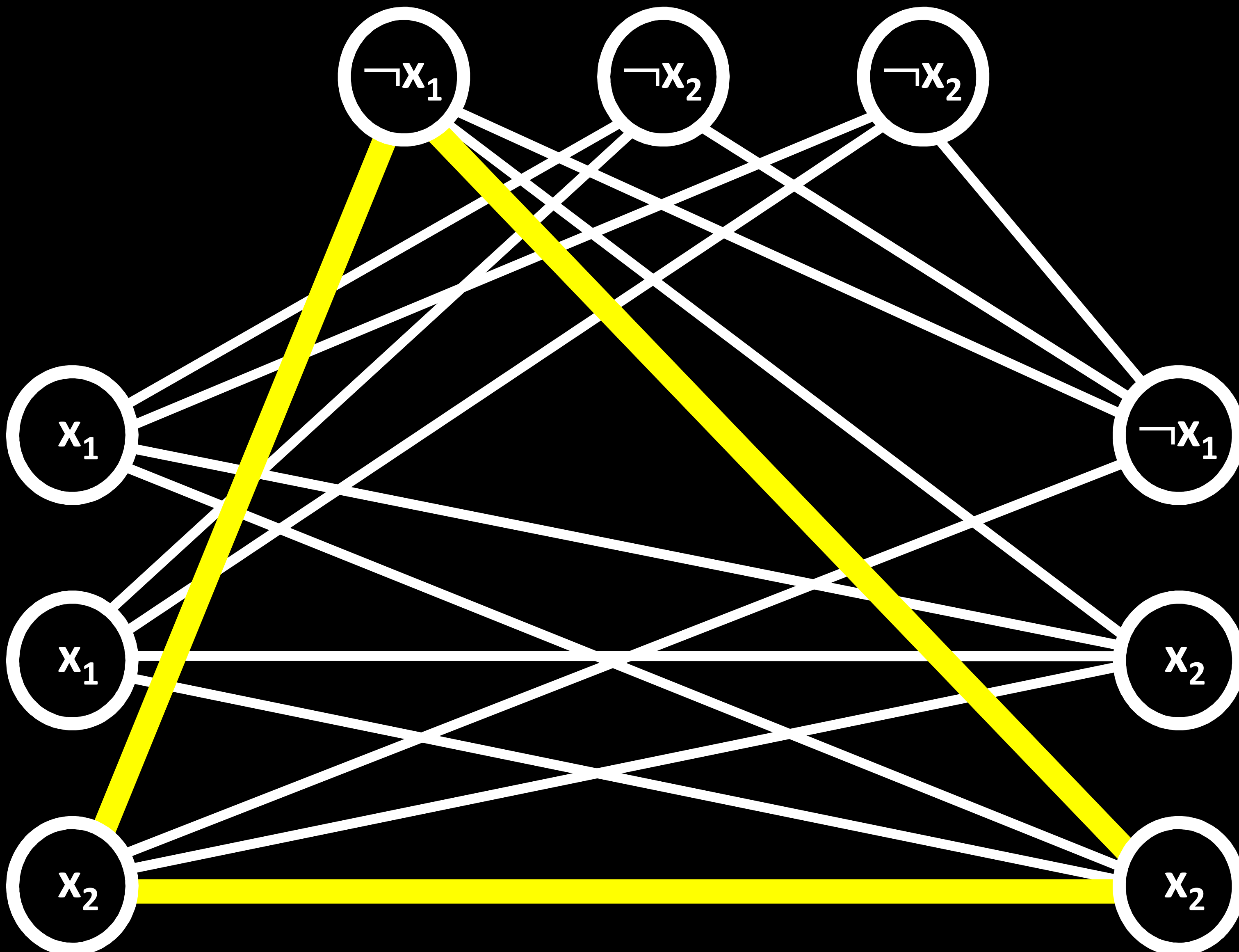
Each node in group *i* is labeled with a literal of C_i

We put edges between all pairs of nodes in different groups, *except those pairs of nodes with labels x and $\neg$x*

We put no edges between nodes in the same group

When done putting in all the edges, erase the labels

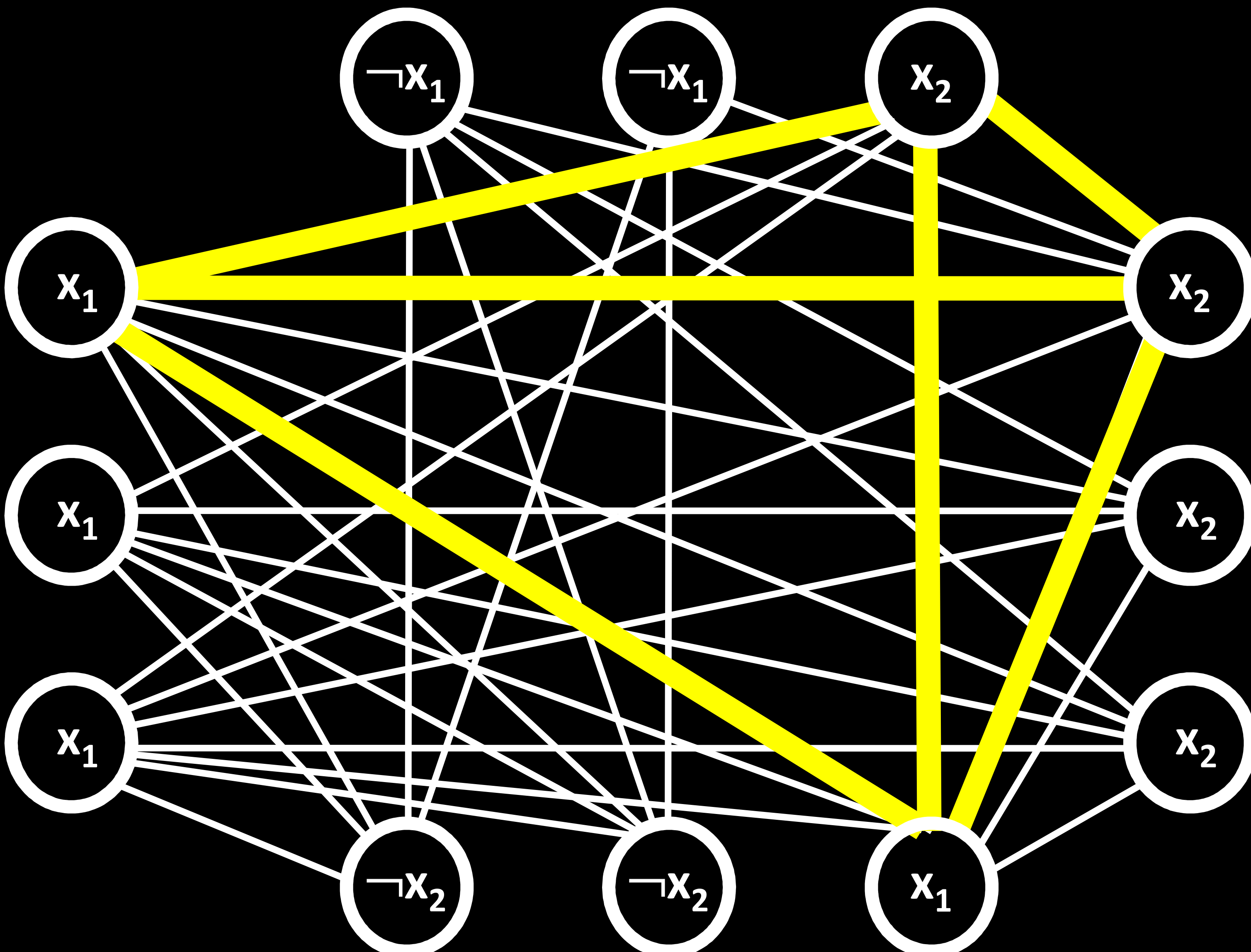
$(x_1 \vee x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_2)$



|V| = 3(number of clauses)

k = number of clauses

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



Claim: $\phi \in$ 3SAT $\Leftrightarrow$ (G,m) $\in$ CLIQUE

Claim: If $\phi \in$ 3SAT then (G,m) $\in$ CLIQUE

Proof: Given a SAT assignment A of ϕ, for every clause C there is at least one literal in C that's set true by A

For each clause C, let v_C be a vertex from group C whose label is a literal that is set true by A

Claim: $S = \{v_C : C \in \phi\}$ is an m-clique

Proof: Let $v_C, v_{C'}$ be in S. Suppose $(v_C, v_{C'}) \notin E$.

Then v_C and $v_{C'}$ must label *inconsistent* literals, call them x and $\neg$x

But assignment A cannot satisfy both x and $\neg$x

Therefore $(v_C, v_{C'}) \in E$, for all $v_C, v_{C'} \in S$.

Hence S is an m-clique, and (G,m) $\in$ CLIQUE

Claim: $\phi \in$ 3SAT $\Leftrightarrow$ (G,m) $\in$ CLIQUE

Claim: If (G,m) $\in$ CLIQUE then $\phi \in$ 3SAT

Proof: Let S be an m-clique of G.

We construct a satisfying assignment A of ϕ.

Claim: S contains *exactly one node* from each group.

Now for each variable x of ϕ, make assignment A:

Assign x to 1 $\Leftrightarrow$ There is a vertex v ϵ S with label x

For all i = 1,…,m, at least one vertex from group i is in S.

Therefore, for all i = 1,…,m

A satisfies at least one literal in the ith clause of ϕ

Therefore A is a satisfying assignment to ϕ

Independent Set

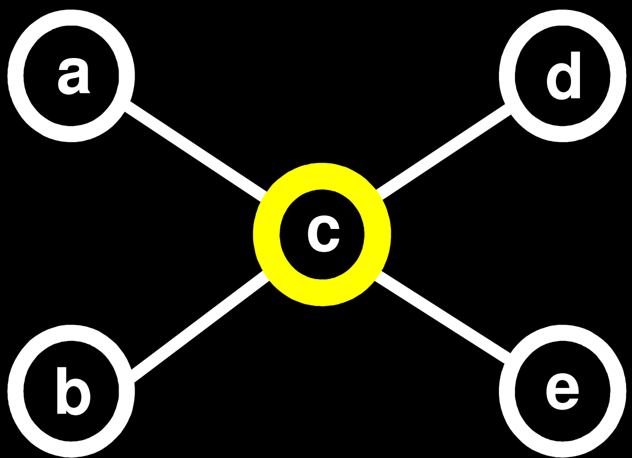
IS: Given a graph $G = (V, E)$ and integer k,
is there $S \subseteq V$ such that $|S| \geq k$ and
***no* two vertices in S have an edge?**

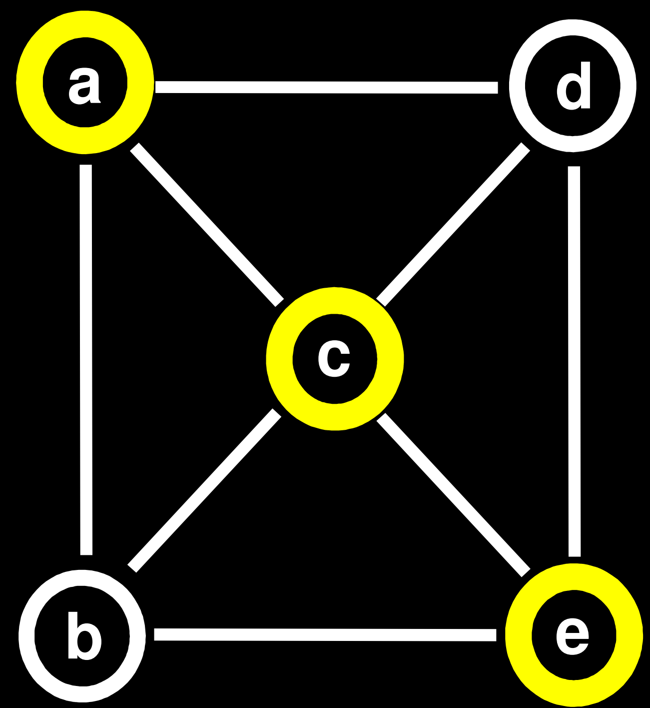
CLIQUE: Given $G = (V, E)$ and integer k,
is there $S \subseteq V$ such that $|S| \geq k$
and every pair of vertices in S have an edge?

CLIQUE $\leq_P$ IS:
Given $G = (V, E)$, output $G' = (V, E')$ where
$E' = \{(u,v) \mid (u,v) \notin E\}$.

$(G, k) \in$ CLIQUE iff $(G', k) \in$ IS

Vertex Cover





vertex cover = set of nodes C that cover all edges

For all edges, at least one endpoint is in C

VERTEX-COVER = { (G,k) | G is a graph with a vertex cover of size at most k}

Theorem: VERTEX-COVER is NP-Complete

(1) VERTEX-COVER $\in$ NP

(2) IS $\leq_P$ VERTEX-COVER

IS $\leq_P$ VERTEX-COVER

Want to transform a graph G and integer k into G' and k' such that

$$(G,k) \in \text{IS} \Leftrightarrow (G',k') \in \text{VERTEX-COVER}$$

IS $\leq_P$ VERTEX-COVER

Claim: For every graph $G = (V,E)$, and subset $S \subseteq V$,

S is an independent set
if and only if $(V - S)$ is a vertex cover

Proof: S is an independent set
$\Leftrightarrow$ $(\forall\, u, v \in V)[\, (u \in S \text{ and } v \in S) \Rightarrow (u,v) \notin E\,]$
$\Leftrightarrow$ $(\forall\, u, v \in V)[\, (u,v) \in E \Rightarrow (u \notin S \text{ or } v \notin S)\,]$
$\Leftrightarrow$ $(V - S)$ is a vertex cover

Therefore $(G,k) \in \text{IS} \Leftrightarrow (G,|V| - k) \in \text{VERTEX-COVER}$

Our polynomial time reduction: $f(G,k) := (G, |V| - k)$