# Lecture 21

# Review from previous lecture

P $\subseteq$ NP $\subseteq$ Exp; at least one containment is proper

Examples in NP:

SAT, short/long paths, Euler/Ham tours, clique, indp set…

Common feature:

"… there is a …"

(and some related problems do *not* appear to share this feature: *Un*SAT, *max*Clique, *Mostly*LongPaths, …)

# Some Problem Pairs

| | |
|---|---|
| Euler Tour | Hamilton Tour |
| 2-SAT | 3-SAT |
| 2-Coloring | 3-Coloring |
| Min Cut | Max Cut |
| Shortest Path | Longest Path |

Superficially different; similar computationally

Similar pairs; seemingly different computationally

# Solving NP problems without hints

The most obvious algorithm for most of these problems is brute force:

- try all possible hints; check each one to see if it works.
- Exponential time:
  - $2^n$ truth assignments for n variables
  - n! possible TSP tours of n vertices
  - $\binom{n}{k}$ possible k element subsets of n vertices
  - etc.

…and to date, every alg, even much less-obvious ones, are slow, too

# P vs NP

## Theory

P = NP ?

Open Problem!

I bet against it

## Practice

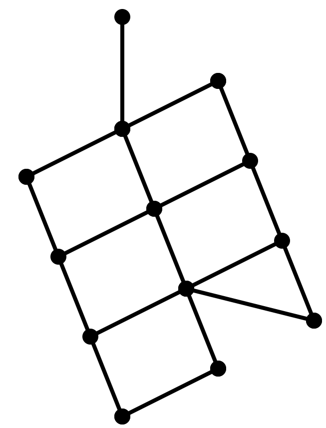
Many interesting, useful, natural, well-studied problems known to be NP-complete

With rare exceptions, no one routinely succeeds in finding exact solutions to large, arbitrary instances

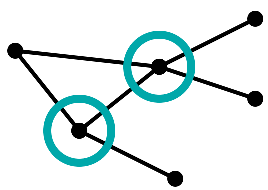
# Another NP problem: Vertex Cover

Input: Undirected graph $G = (V, E)$, integer $k$.

Output: True iff there is a subset $C$ of $V$ of size $\leq k$ such that every edge in $E$ is incident to at least one vertex in $C$.

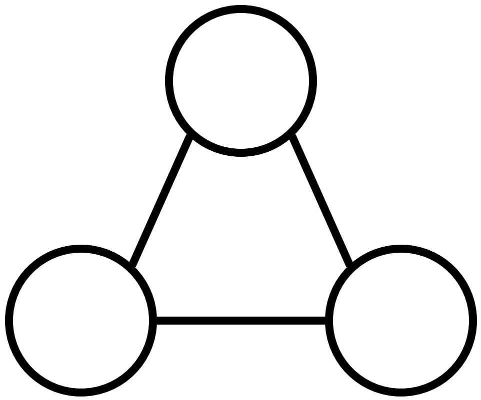


Example: Vertex cover of size $\leq 2$.

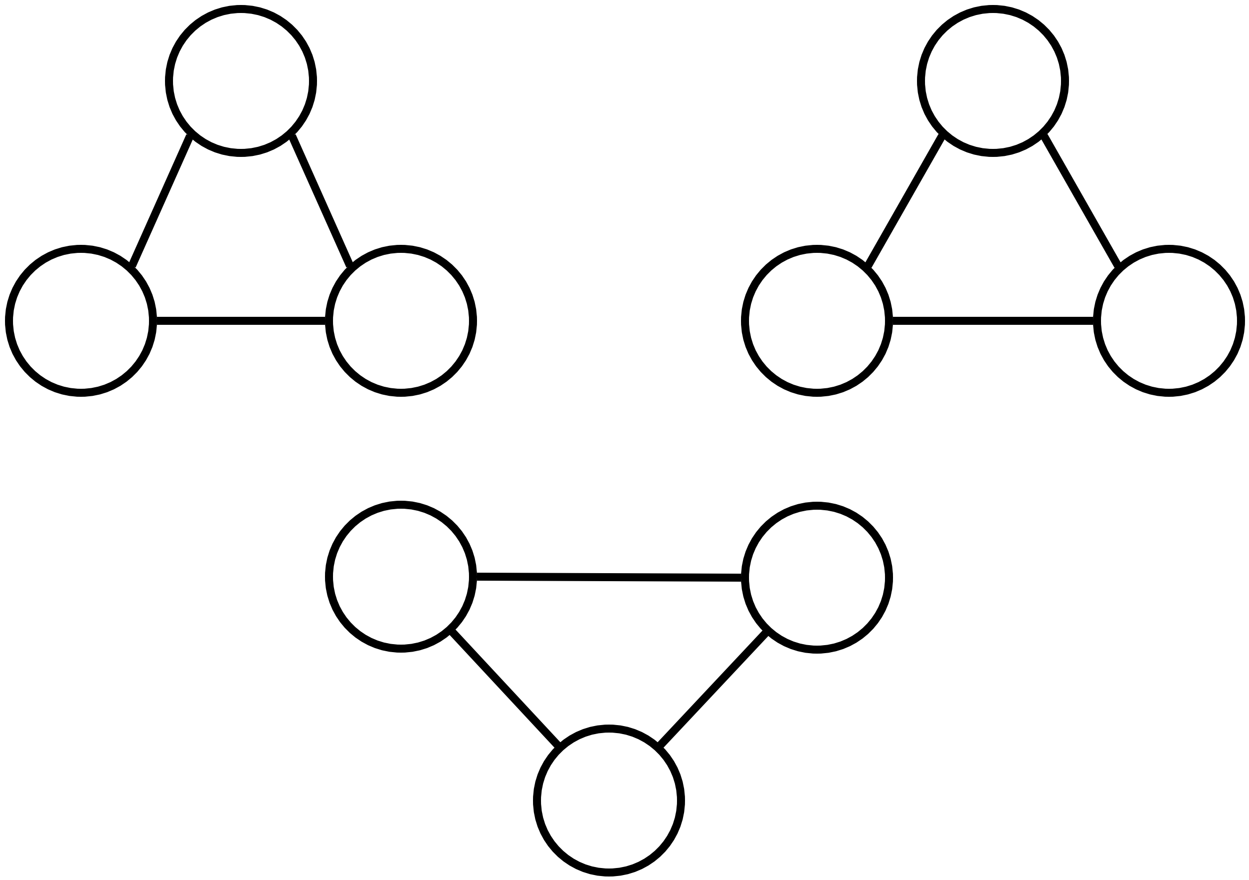


In NP? Exercise

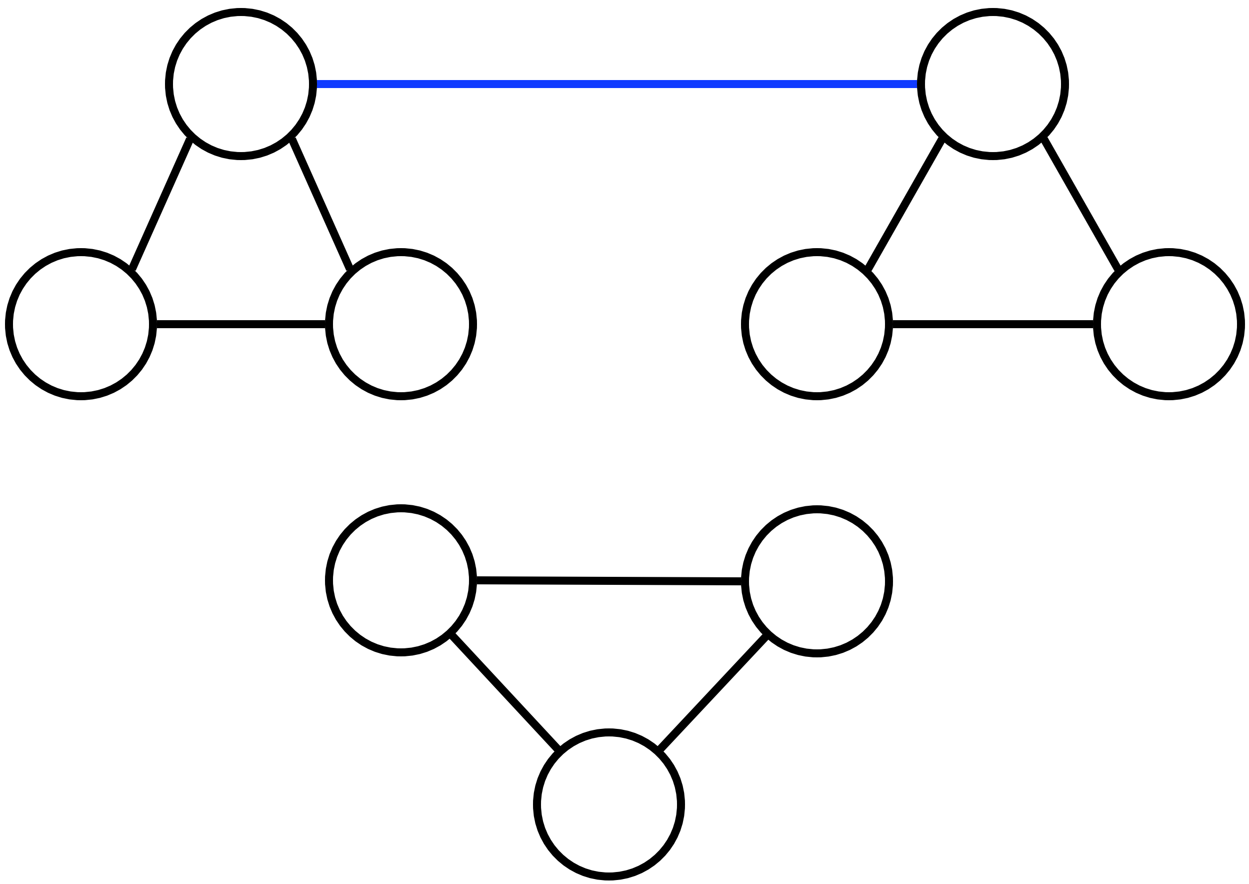
# 3SAT $\leq_p$ VertexCover



# 3SAT $\leq_p$ VertexCover

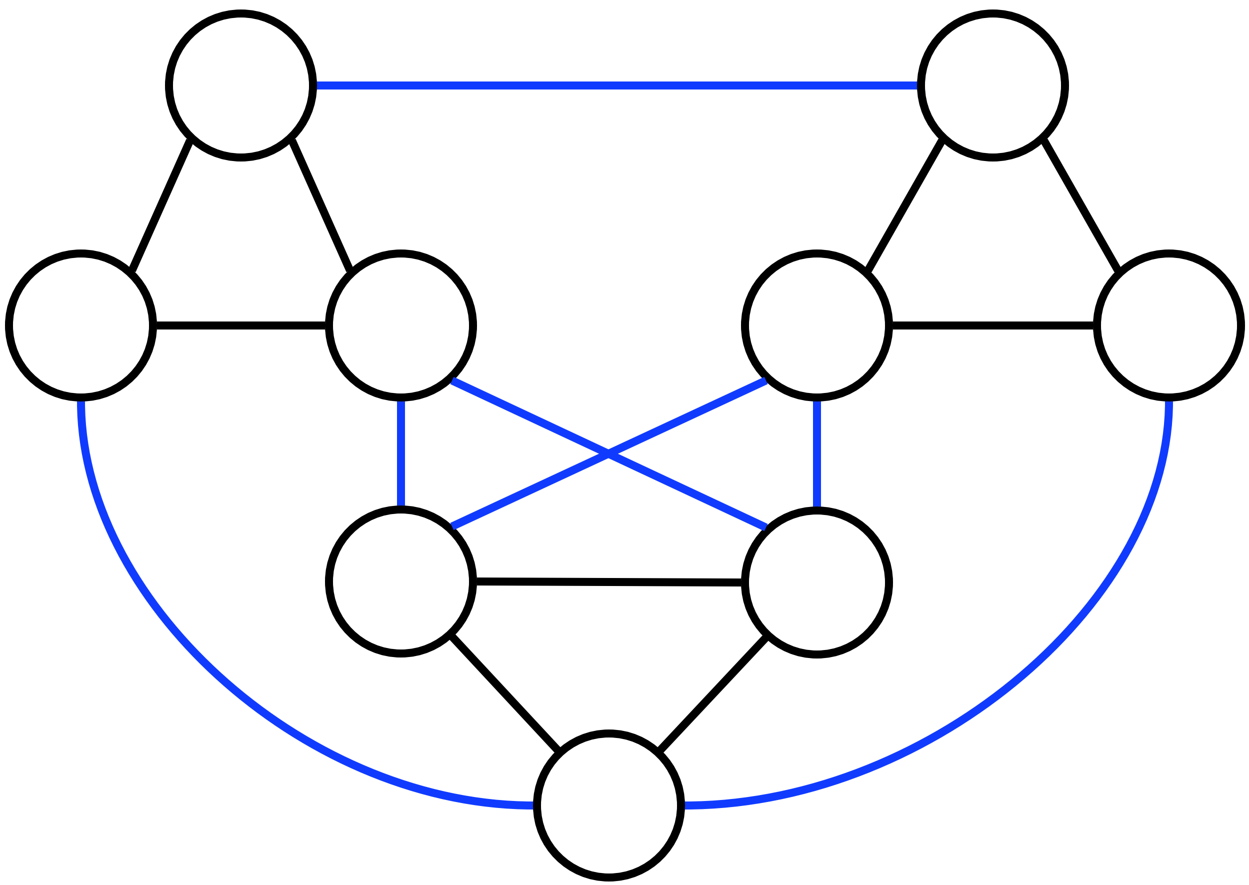


# 3SAT $\leq_p$ VertexCover



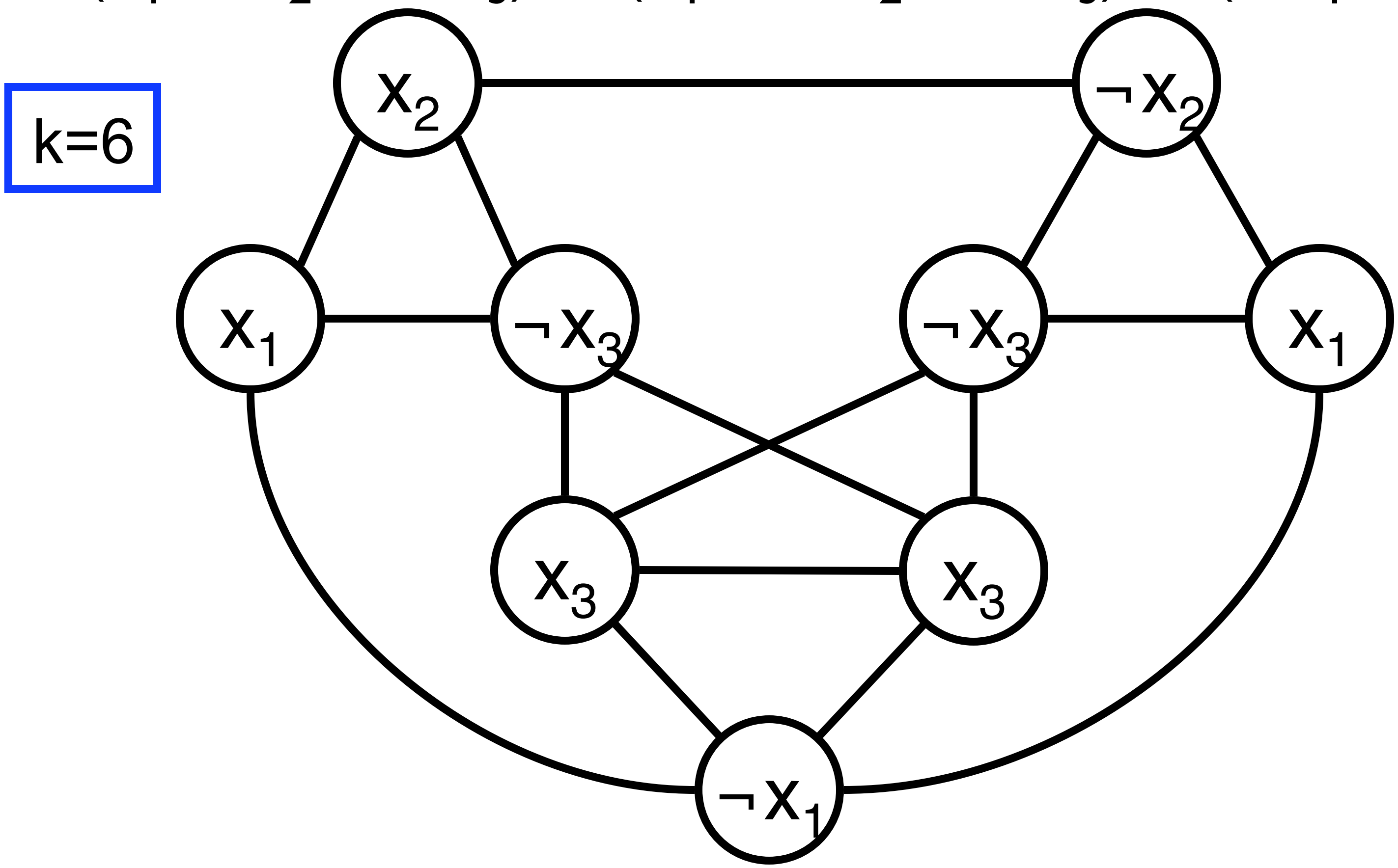
# 3SAT $\leq_p$ VertexCover

k=6



# 3SAT $\leq_p$ VertexCover

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



# $3SAT \leq_p VertexCover$

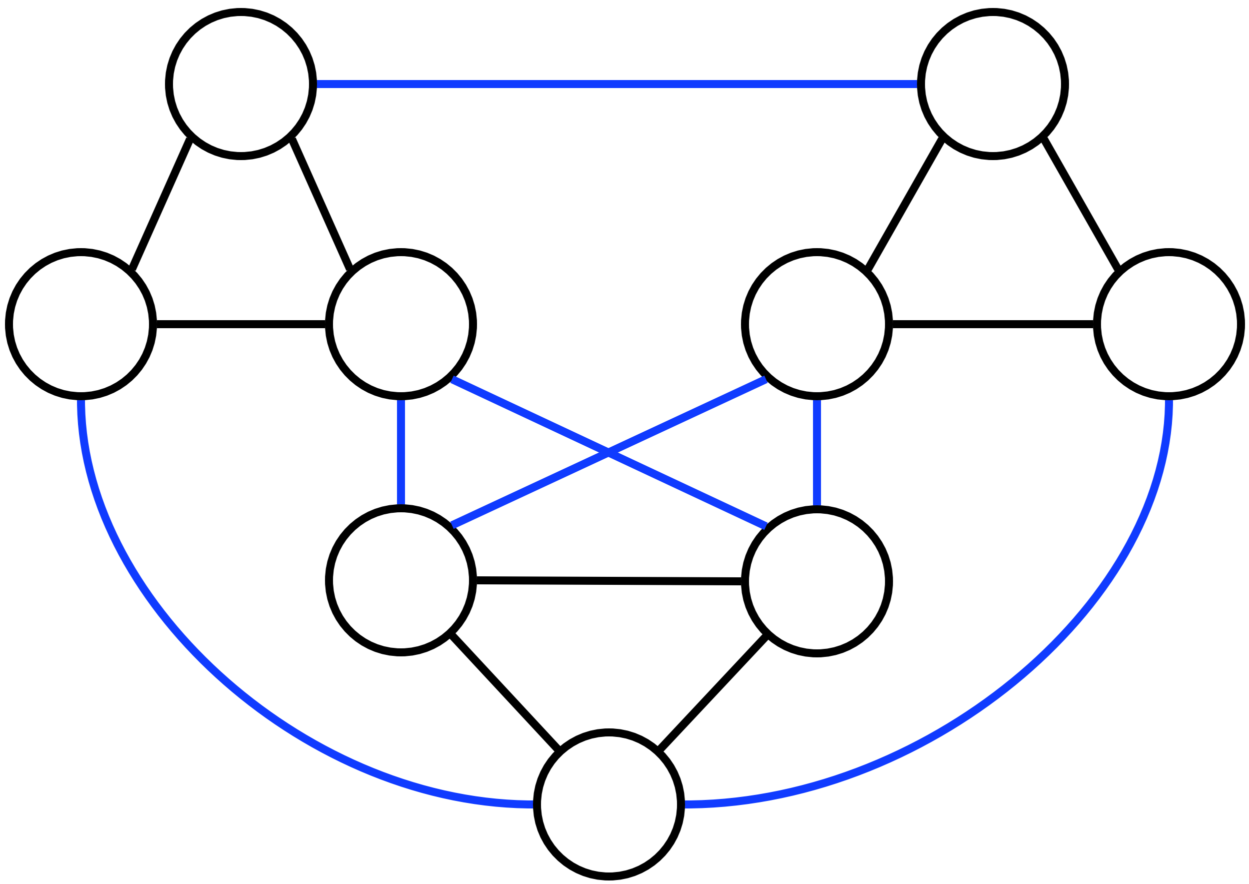
f (
3-SAT Instance:
- Variables: $x_1, x_2, \ldots$
- Literals: $y_{i,j}$, $1 \leq i \leq q$, $1 \leq j \leq 3$
- Clauses: $c_i = y_{i1} \vee y_{i2} \vee y_{i3}$, $1 \leq i \leq q$
- Formula: $c = c_1 \wedge c_2 \wedge \ldots \wedge c_q$

) =

VertexCover Instance:
- $k = 2q$
- $G = (V, E)$
- $V = \{ [i,j] \mid 1 \leq i \leq q, 1 \leq j \leq 3 \}$
- $E = \{ ( [i,j], [k,l] ) \mid i = k \text{ or } y_{ij} = \neg y_{kl} \}$

# 3SAT $\leq_p$ VertexCover

k=6



# Correctness of "3SAT $\leq_p$ VertexCover"

Summary of reduction function f: Given formula, make graph G with one group per clause, one node per literal. Connect each to all nodes in same group, plus complementary literals (x, ¬x). Output graph G plus integer k = 2 * number of clauses. Note: *f does not know whether formula is satisfiable or not; does not know if G has k-cover; does not try to find satisfying assignment or cover.*

Correctness:

- Show f poly time computable: A key point is that graph size is polynomial in formula size; mapping basically straightforward.
- Show c in 3-SAT iff f(c)=(G,k) in VertexCover:

(⇒) Given an assignment satisfying c, pick one true literal per clause. Add other 2 nodes of each triangle to cover. Show it is a cover: 2 per triangle cover triangle edges; only true literals (but perhaps not all true literals) uncovered, so at least one end of every (x, ¬x) edge is covered.

(⇐) Given a k-vertex cover in G, uncovered labels define a valid (perhaps partial) truth assignment since no (x, ¬x) pair uncovered. It satisfies c since there is one uncovered node in each clause triangle (else some other clause triangle has > 1 uncovered node, hence an uncovered edge.)