What does this mean:

if we have a program A' that solves L', then we can use A' to solve L:

on input x (to L):

- compute $x' = f(x)$
- run A' on input x'.

Thus: we only have to write a program for the function f.

Theorem: If $L \leq_P L'$ and $L' \in P$, then $L \in P$.

Proof: [intuition:

$$\left.\begin{array}{l} L' \in P : L' \text{ is easy} \\ L \leq_P L' : L \text{ is easier than } L' \end{array}\right\} \begin{array}{l} \therefore L \text{ is easy} \\ \therefore L \in P \end{array}$$

]

Since $L' \in \mathcal{P}$: there is a polynomial-time algorithm A' such that

$$x' \in L' \iff A'(x') \text{ returns YES.}$$

Consider the following algorithm A:

on input x:

- compute $f(x)$
- run $A'(f(x))$.

Then:

$$\begin{aligned} x \in L &\iff f(x) \in L' && [\text{by definition of reduction}] \\ &\iff A'(f(x)) \text{ returns YES} && [\text{by definition of } A'] \\ &\iff A(x) \text{ returns YES} && [\text{by definition of } A] \end{aligned}$$

Running time of A = polynomial in the length of x.

$\therefore L \in \mathcal{P}$.

$\square$

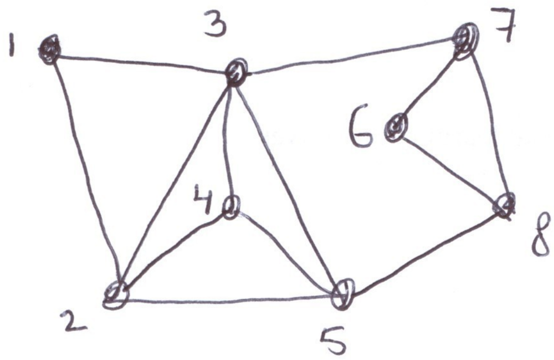
Example of a reduction:

CLIQUE = { (G, K) : graph G has a clique with K vertices }

INDEP-SET = { (G, K) : graph G has an independent set with K vertices }

clique: each pair of vertices is connected by an edge

independent set: no pair of vertices is connected by an edge



{2,3,4,5}: clique of size 4

{1,4,6}: independent set of size 3.

We will show that

$$\text{CLIQUE} \leq_P \text{INDEP-SET}.$$

We need a function f such that

1. $f : \underbrace{(G, K)}_{\substack{\text{input for} \\ \text{CLIQUE}}} \longrightarrow \underbrace{(G', K')}_{\substack{\text{input for} \\ \text{INDEP-SET}}}$

2. G has a clique of size K $\Longleftrightarrow$
   G' has an independent set of size K'

3. time to construct (G', K'), when given (G, K) is polynomial in the length of (G, K).

Here is f:

$$f(G, K) = (\overline{G}, K) \quad [\text{thus: } G' = \overline{G}, K' = K]$$

where $\overline{G}$ is the complement of G

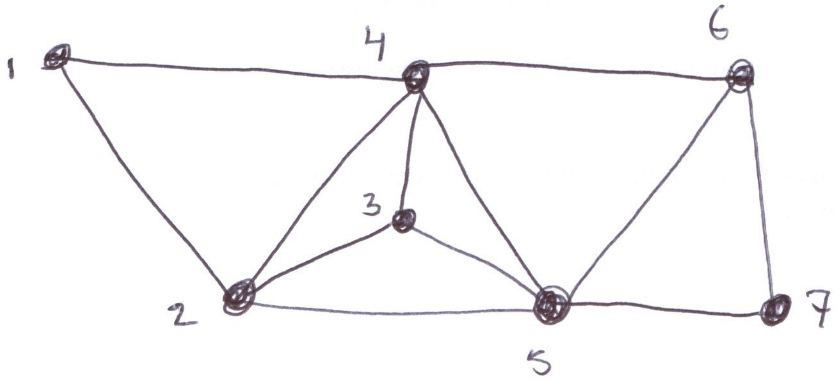
(replace edges by non-edges, and non-edges by edges)

<u>Exercise</u>: Show that 1, 2, 3 hold.

VERTEX-COVER =

$\{(G,K)$: graph G contains a vertex cover with K vertices$\}$

vertex cover: subset V' of the vertex set such that for each edge $\{u,v\}$ of G, at least one of u and v is in V'.



$\{2,4,5,7\}$ is a vertex cover of size 4.

We will show that

CLIQUE $\leq_P$ VERTEX-COVER.

We need a function f such that

① $f : \underbrace{(G, K)}_{\text{input for CLIQUE}} \rightarrow \underbrace{(G', K')}_{\text{input for VERTEX-COVER}}$

② G has a clique of size K $\Longleftrightarrow$
G′ has a vertex cover of size K′

③ f can be computed in polynomial time.

Here is f:

$$f(G, K) = (\bar{G}, n-K)$$

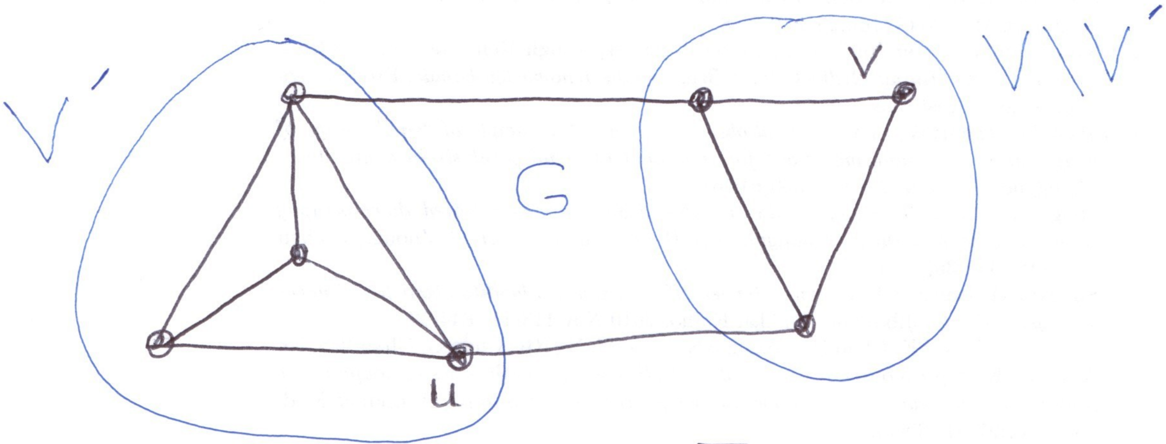
where n = # vertices of G.

Properties ① + ③ hold.

For ②, we have to show:

G has a clique of size K $\iff$ $\overline{G}$ has a vertex cover of size $n-K$.

$\Rightarrow$ Assume $G=(V,E)$ contains a clique V' of size K.



Let $\{u,v\}$ be an edge in $\overline{G}$. Then $\{u,v\}$ is not an edge in G $\therefore$ u and v are not both in V'

$\therefore$ at least one of u and v is in $V\setminus V'$.

$\therefore$ $V\setminus V'$ is a vertex cover in $\overline{G}$

$$|V\setminus V'| = |V| - |V'| = n-K.$$

$\Leftarrow$ Assume $\overline{G} = (V, \overline{E})$ contains a vertex cover V' of size $n-K$.

Let $u, v \in V \setminus V'$, $u \neq v$.

If $\{u,v\}$ is an edge of $\overline{G}$, then V' is not a vertex cover of $\overline{G}$ $\lightning$

$\therefore$ $\{u,v\}$ is not an edge of $\overline{G}$

$\therefore$ $\{u,v\}$ is an edge of G.

$\therefore$ any pair of vertices in $V \setminus V'$ is connected by an edge in G

$\therefore$ $V \setminus V'$ is a clique in G

$|V \setminus V'| = |V| - |V'| = n - (n-K) = K.$

Example of a reduction:

Boolean formula φ with variables $x_1, \dots, x_n$ of the form

$$\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m,$$

where each C_i is of the form

$$C_i = \ell_1^i \vee \ell_2^i \vee \ell_3^i,$$

each ℓ_j^i is a variable or the negation of a variable.

C_i is called a <u>clause</u>

ℓ_j^i is called a <u>literal</u>

$$\varphi = (x_1 \vee \neg x_1 \vee \neg x_2) \wedge (x_2 \vee x_3 \vee x_4)$$
$$\wedge (x_1 \vee x_3 \vee \neg x_4)$$

φ is <u>satisfiable</u> if there exists a truth value for each of $x_1, \ldots, x_n$ such that φ is true.

for the example: $x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 0$

then $\varphi = 1 \therefore \varphi$ is satisfiable.

3SAT = $\{\varphi : \varphi$ is of the form above, φ is satisfiable$\}$.

We will show that 3SAT $\leq_{P}$ INDEP-SET.

We need a function f such that

① $f : \varphi \to (G, K)$

② φ satisfiable $\iff$ G has an independent set of size K

③ φ can be computed in polynomial time.

Thus: we have to "encode" a satisfiable formula φ as an independent set of size K in a graph G.

Let φ be an input for 3SAT;

$$\varphi = C_1 \wedge C_2 \wedge \ldots \wedge C_m,$$

each clause C_i is the $\vee$ of 3 literals.

(G, K) is obtained as follows:

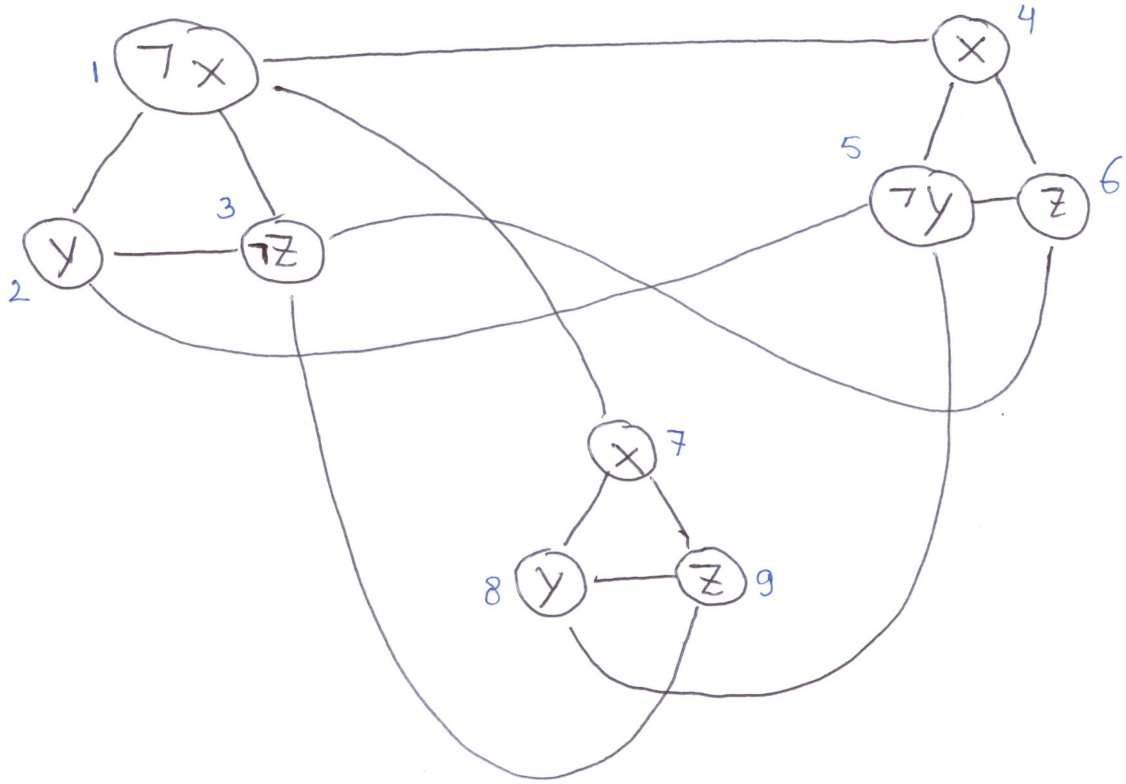
$K = m$ (= number of clauses)

G: $3m$ vertices; one vertex for each literal.

for each clause: literals in this clause form a triangle in G.

additionally: edge between any pair of opposite literals.

$$\varphi = (\neg x \vee y \vee \neg z) \wedge (x \vee \neg y \vee z) \wedge (x \vee y \vee z)$$



φ is satisfiable; e.g., $x=1$, $y=0$, $z=0$

first clause: $\neg z = \text{true}$: choose vertex 3
second clause: $\neg y = \text{true}$: choose vertex 5
third clause: $x = \text{true}$: choose vertex 7

} independent set of size 3

In general: Assume φ is satisfiable. Then there is a truth-assignment for the variables such that φ is true.

$\therefore$ every clause is true

$\therefore$ in every clause, at least one literal is true

choose one literal which is true and pick the corresponding vertex.

This gives $m = K$ vertices in G which form an independent set [Why?].

Thus:

$$\varphi \in 3SAT \Rightarrow (G, K) \in \text{INDEP-SET}.$$

In the example, take an independent set of size 3 : $\{1, 6, 9\}$.

vertex 1 : set x such that $\neg x = 1$: $x = 0$
vertex 6 : set z such that $z = 1$: $z = 1$
vertex 9 : set z such that $z = 1$: $z = 1$

for these values, $\varphi = 1$

In general, assume G contains an independent set of size $K = m$.

Since G consists of m triangles; the independent set contains one vertex of each triangle.

For each triangle: let v be the vertex that is in the independent set.

if v is of the form x : set $x = 1$

if v is of the form $\neg x$: set $x = 0$

In this way, (a subset of) the variables get a truth value such that φ = true.

Can this happen: v and w in the independent set

$v = x$: set $x = 1$

$w = \neg x$: set $x = 0$?

No, because v and w are connected by an edge in G.

Thus:

$(G,K) \in$ INDEP-SET $\Rightarrow \varphi \in$ 3SAT.

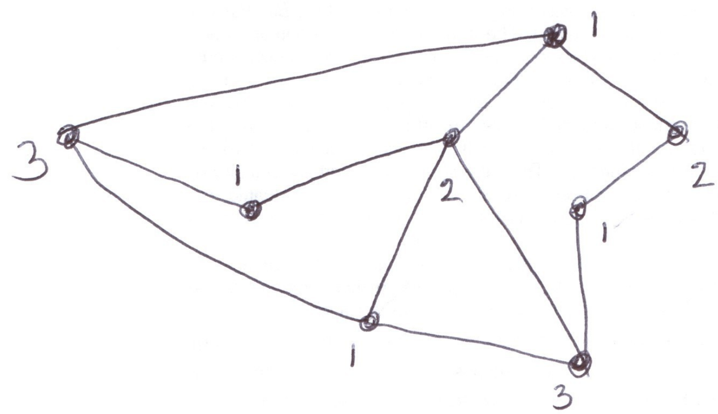
Time to compute $(G,K) = f(\varphi)$:

$O(\underbrace{(3m)^2}_{\text{use brute-force to compute the edges of } G}) = O(m^2) = O((\#\text{clauses in } \varphi)^2)$

polynomial.

Another example of a reduction

3COLOR = { G : G is a graph whose vertices can be colored using three colors such that any 2 adjacent vertices have distinct colors }



"colors": 1, 2, 3

We will show that 3COLOR $\leq_P$ 3SAT.

We need a function f such that

① f : graph G $\longrightarrow$ Boolean formula $\varphi = f(G)$

② G has a 3-coloring $\Longleftrightarrow$ φ is satisfiable

③ time to compute φ is polynomial in the length of G.

Consider a graph $G = (V, E)$, and write $V = \{v_1, \dots, v_n\}$.

We use $3n$ Boolean variables x_{ik}, $1 \le i \le n$, $1 \le k \le 3$.
We use colors 1, 2, 3.

Idea:

$$x_{ik} = \begin{cases} 1 & \text{if vertex } v_i \text{ gets color } k, \\ 0 & \text{otherwise.} \end{cases}$$

Consider vertex v_i. In a 3-coloring, v_i must have exactly one of the colors 1, 2, 3.

$x_{i1} \vee x_{i2} \vee x_{i3}$ is true $\Longleftrightarrow$ v_i has at least one color.

$\neg(x_{i1} \wedge x_{i2}) \wedge \neg(x_{i1} \wedge x_{i3}) \wedge \neg(x_{i2} \wedge x_{i3})$ is true

$\Longleftrightarrow$ v_i has at most one color.

Using De Morgan, this becomes

$$(\neg x_{i1} \vee \neg x_{i2}) \wedge (\neg x_{i1} \vee \neg x_{i3}) \wedge (\neg x_{i2} \vee \neg x_{i3}).$$

Thus: The statement "every vertex has exactly one color" is equivalent to the statement

$$\text{"}\quad \varphi_1 \left\{ \begin{array}{l} \displaystyle\bigwedge_{i=1}^{n} \left(x_{i1} \vee x_{i2} \vee x_{i3} \right) \quad \wedge \\[2ex] \displaystyle\bigwedge_{i=1}^{n} \Big((\neg x_{i1} \vee \neg x_{i2}) \wedge (\neg x_{i1} \vee \neg x_{i3}) \wedge (\neg x_{i2} \vee \neg x_{i3}) \Big) \end{array} \right.$$

is true".

Consider an edge $\{v_i, v_j\}$ in G. In a 3-coloring, v_i does not have the same color as v_j. This is described by

$$\neg (x_{i1} \wedge x_{j1}) \wedge \neg (x_{i2} \wedge x_{j2}) \wedge \neg (x_{i3} \wedge x_{j3}).$$

Using De Morgan, this becomes

$$(\neg x_{i1} \vee \neg x_{j1}) \wedge (\neg x_{i2} \vee \neg x_{j2}) \wedge (\neg x_{i3} \vee \neg x_{j3}).$$

Thus: The statement "any 2 adjacent vertices have distinct colors" is equivalent to the statement

"
$$\bigwedge_{\{v_i, v_j\} \in E} \Big((\neg x_{i1} \vee \neg x_{j1}) \wedge (\neg x_{i2} \vee \neg x_{j2}) \wedge (\neg x_{i3} \vee \neg x_{j3}) \Big) \quad \Big\} \varphi_2$$

is true".

If we define $\varphi = f(G) = \varphi_1 \wedge \varphi_2$, then

G has a 3-coloring $\Longleftrightarrow$ φ is satisfiable.

φ is (almost) in the correct format for 3SAT:

replace any clause $\ell \vee \ell'$ of length 2 by the equivalent clause $\ell \vee \ell' \vee \ell'$ of length 3.

Length of φ : $\underbrace{O(n)}_{\varphi_1} + \underbrace{O(|E|)}_{\varphi_2} = O(|V| + |E|)$.

φ can be constructed in $O(|V| + |E|)$ time, which is polynomial in the length of G.