



# Complexity IBC028, Lecture 5

H. Geuvers

Institute for Computing and Information Sciences  
Radboud University Nijmegen

Version: spring 2026





# Outline

Proving that a problem is NP-complete

More **NP**-complete satisfiability problems

Some other **NP**-complete problems





## Recap: **P** and **NP**

**P** :=

$$\{A \subseteq \{0, 1\}^* \mid \exists f, f \text{ polynomial}, x \in A \iff f(x) = 1\}$$

**NP** :=

$$\{A \subseteq \{0, 1\}^* \mid \exists f, f \text{ polynomial}, \\ x \in A \iff \exists y \in \{0, 1\}^* (|y| \text{ polynomial in } |x| \wedge f(x, y) = 1)\}$$

- **P** = the class of polynomial time decision problems.
- **NP** = the class of non-deterministic polynomial time decision problems.
- Property: **P**  $\subseteq$  **NP**



## Recap: NP-hard and NP-complete

### DEFINITION

A (polynomially) **reduces to** B, notation  $A \leq_P B$  if there is a **polynomial** function  $f : \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that

$$x \in A \iff f(x) \in B$$

### DEFINITION

- **NPH** :=  $\{A \mid \forall X \in \mathbf{NP} (X \leq_P A)\}$   
A is **NP-hard** if  $A \in \mathbf{NPH}$ .
- **NPC** :=  $\mathbf{NP} \cap \mathbf{NPH}$   
A is **NP-complete** if  $A \in \mathbf{NP}$  and A is **NP-hard**.

### THEOREM

If  $B \in \mathbf{NPH}$  and  $B \leq_P A$ , then  $A \in \mathbf{NPH}$ .



# NP and co-NP

## DEFINITION

**co-NP** :=  $\{A \mid \bar{A} \in \mathbf{NP}\}$ . ( $\bar{A}$  is the complement of  $A$ .)

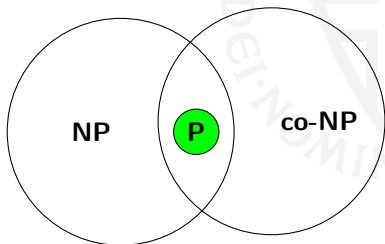
The precise relations between **P**, **NP** and **co-NP** are a major open question in Computer Science.

Most notably there is the open question:

$$\mathbf{P} \stackrel{??}{=} \mathbf{NP}.$$

This is how most people view the world:

$\mathbf{P} \subsetneq \mathbf{NP} \cap \mathbf{co-NP}$ ,  
and we assume it for the course  
(but it has not been proven).





# SAT

SAT is a known **NP**-complete problem.

- The **boolean formulas** are built from
  - Atoms,  $p, q, r, \dots$
  - Boolean connectives  $\wedge, \vee, \neg$  (and possibly  $\rightarrow, \leftrightarrow, \perp, \top$ ).
- A formula  $\varphi$  is **satisfiable** if there is an **assignment**  $v : \text{Atoms} \rightarrow \{0, 1\}$  such that  $v(\varphi) = 1$  (" $\varphi$  is true in  $v$ ").
- **SAT** is the problem of deciding whether a boolean formula  $\varphi$  is satisfiable. **SAT** =  $\{\varphi \mid \exists v(v(\varphi) = 1)\}$ .
- **SAT** is in **NP**: the assignment  $v$  is the certificate, polynomial in  $|\varphi|$ , and  $v(\varphi) = 1$  can be decided in polynomial time.
- **SAT** is in **NPH**. This is the famous Cook-Levin theorem, (showing that every problem in **NP** can be reduced to a SAT-problem; the proof will be given in Lecture 7).



# CNF-SAT

**CNF-SAT** is also **NP**-complete and will be used more often.

- The **conjunctive normal forms** (CNF) are a subset of the boolean formulas:
  - A CNF is a **conjunction of clauses**
  - A clause is a **disjunction of literals**
  - a literal is an atom or a negated atom.
- A CNF-formula  $\varphi$  is **satisfiable** if there is an **assignment**  $v : \text{Atoms} \rightarrow \{0, 1\}$  such that  $v(\varphi) = 1$  (" $\varphi$  is true in  $v$ ").
- **CNF-SAT** is the problem of deciding if a CNF-formula  $\varphi$  is satisfiable.
- **CNF-SAT** is in **NP**: again the assignment  $v$  is the certificate.
- That **CNF-SAT** is in **NPH** will be shown in Lecture 7, and is a direct corollary of the proof of the Cook-Levin theorem.



# Proving that a problem is NP-complete

How to prove that a given problem  $A$  is **NP**-complete?

- 1 Prove that  $A \in \mathbf{NP}$ : give a polynomial algorithm  $f$  such that  $f$  verifies  $A$  with polynomial certificates, that is:

$$x \in A \iff \exists y \in \{0, 1\}^*(|y| \text{ polynomial in } |x| \wedge f(x, y) = 1)$$

Concretely:

- 1 define what a **certificate**  $y$  for an input  $x$  is,
  - 2 show that  $|y|$  is polynomial in  $|x|$ ,
  - 3 define the function  $f$  and show that it is polynomial,
  - 4 prove that  $x \in A \iff f(x, y) = 1$ .
- 2 Prove that  $A$  is **NP-hard**: Pick a well-known **NP-hard** problem  $B$  and prove that  $B \leq_P A$ . This is the **hard part!**



# $\leq_3$ CNF-SAT is NP-complete

## DEFINITION

A **conjunctive normal form with  $\leq 3$  literals**,  $\leq_3$ CNF, is

- a conjunction of clauses where
- every clause is a disjunction of **at most three** literals

$\leq_3$ CNF-SAT is the problem of deciding for an  $\leq_3$ CNF formula whether it is satisfiable.

## THEOREM

$\leq_3$ CNF-SAT is **NP**-complete

## Proof

- $\leq_3$ CNF-SAT is **NP**: an assignment  $v : \text{Atoms} \rightarrow \{0, 1\}$  that makes the formula true is the certificate. (Checking is easy.)
- $\leq_3$ CNF-SAT is **NP**-hard: We prove  $\text{CNF-SAT} \leq_P \leq_3\text{CNF-SAT}$ .



## Proof of $\text{CNF-SAT} \leq_P \leq_3 \text{CNF-SAT}$ (I)

We define a function  $f : \text{CNF} \rightarrow \leq_3 \text{CNF}$  such that

$\varphi$  is satisfiable  $\Leftrightarrow f(\varphi)$  is satisfiable.

The definition of  $f$  is based on the following Lemma:

### LEMMA

$\bigvee_{i=1}^n l_i$  is satisfiable  $\iff$   
 $(l_1 \vee l_2 \vee a) \wedge (\neg a \vee \bigvee_{i=3}^n l_i)$  is satisfiable, where  $a$  is a fresh atom.

Proof.  $\Rightarrow$ :

$\Leftarrow$



## Proof of the Lemma

### LEMMA

$\bigvee_{i=1}^n \ell_i$  is satisfiable  $\iff$   
 $(\ell_1 \vee \ell_2 \vee a) \wedge (\neg a \vee \bigvee_{i=3}^n \ell_i)$  is satisfiable, where  $a$  is a fresh atom.

### PROOF

$\Rightarrow$ : If  $v$  is an assignment that makes  $\varphi \wedge \bigvee_{i=1}^n \ell_i$  true, then

- (1)  $v$  makes  $\ell_1 \vee \ell_2$  true or
- (2)  $v$  makes  $\bigvee_{i=3}^n \ell_i$  true (or both).

In case (1), set  $v(a) = 0$ , in case (2), set  $v(a) = 1$ . Then  $v$  makes  $(\ell_1 \vee \ell_2 \vee a) \wedge (\neg a \vee \bigvee_{i=3}^n \ell_i)$  true.

$\Leftarrow$ : If  $v$  makes  $(\ell_1 \vee \ell_2 \vee a) \wedge (\neg a \vee \bigvee_{i=3}^n \ell_i)$  true, then

- (1)  $v(a) = 1$  or
- (2)  $v(a) = 0$ .

In case (1),  $v(\bigvee_{i=3}^n \ell_i) = 1$ , in case (2),  $v(\ell_1 \vee \ell_2) = 1$ . In either case,  $v(\bigvee_{i=1}^n \ell_i) = 1$ , so we are done.



## Proof of $\text{CNF-SAT} \leq_P \leq_3 \text{CNF-SAT}$ (III)

### DEFINITION

Define  $f(\varphi)$  by recursively replacing in  $\varphi$  every disjunction  $\bigvee_{i=1}^n \ell_i$  where  $n > 3$  by  $(\ell_1 \vee \ell_2 \vee a) \wedge (\neg a \vee \bigvee_{i=3}^n \ell_i)$  for a fresh atom  $a$ .

- The function  $f$  is polynomial. It doesn't blow up the formula  $\varphi$ :  $|f(\varphi)| = \mathcal{O}(|\varphi|)$ .
- The Lemma (previous slide) proves that  $\varphi$  is satisfiable iff  $f(\varphi)$  is satisfiable. Therefore  $\text{CNF-SAT} \leq_P \leq_3 \text{CNF-SAT}$  and so  $\leq_3 \text{CNF-SAT}$  is **NP-hard**. We have already shown  $\leq_3 \text{CNF-SAT} \in \mathbf{NP}$ , so  $\leq_3 \text{CNF-SAT}$  is **NP-complete**.
- NB. One often requires that all literals in a clause are different. We will often assume that.



# 3CNF-SAT is NP-complete

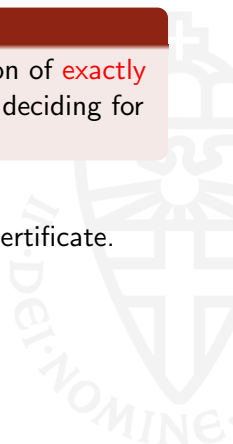
## DEFINITION

A 3CNF is a  $\leq_3$ CNF where every clause is a disjunction of **exactly three** (different) literals. 3CNF-SAT is the problem of deciding for an 3CNF whether it is satisfiable.

We prove that 3CNF-SAT is **NP**-complete

- 3CNF-SAT  $\in$  **NP**. Again, the assignment is the certificate. Checking the properties is easy.
- 3CNF-SAT is **NP**-hard. We show that

$$\leq_3\text{CNF-SAT} \leq_P \text{3CNF-SAT.}$$





# Proof of $\leq_3$ CNF-SAT $\leq_P$ 3CNF-SAT

We reduce  $\leq_3$ CNF-SAT to 3CNF-SAT by defining  $f : \leq_3\text{CNF} \rightarrow 3\text{CNF}$ .

- Choose fresh atoms  $a, a', p, q$ .
- Add clauses  $A = a \vee p \vee q, a \vee p \vee \neg q, a \vee \neg p \vee q, a \vee \neg p \vee \neg q$ , and similarly  $A'$  for  $a'$ .
- Note that:  $A \wedge A'$  can only be satisfied if  $v(a) = v(a') = 1$ .
- Replace every clause  $c$  with two literals by  $c \vee \neg a$ .
- Replace every clause  $c$  with one literal by  $c \vee \neg a \vee \neg a'$ .

Now one easily proves that

$$\varphi \text{ is satisfiable} \Leftrightarrow f(\varphi) \text{ is satisfiable.}$$



## Satisfiability problems that are NP-complete

- We have seen that SAT, CNF-SAT,  $\leq_3$ CNF-SAT and 3CNF-SAT are **NP**-complete. (Proof for SAT, CNF-SAT in Lecture 7.)
- This has been proven by showing that they are in **NP** (easy) and by showing that they are **NP**-hard (the real work).
- For showing **NP**-hardness we have used the following chain of reductions.

$$\text{CNF-SAT} \leq_P \leq_3 \text{CNF-SAT} \leq_P \text{3CNF-SAT}$$

- Not all satisfiability problem are **NP**-hard! For example, 2CNF-SAT is polynomial. (2CNF-SAT is the problem of deciding satisfiability of CNFs where every clause has **exactly 2** literals.)

There are also lots of other (“real”) problems that are **NP**-complete.



# ILP is NP-complete (I)

## DEFINITION

Integer Linear Programming, ILP is the problem of deciding if a finite set of inequalities with coefficients in  $\mathbb{Z}$  has a solution in  $\mathbb{Z}$ .

Example with 2 variables

$$E := \begin{cases} x_1 + 3x_2 & \geq 5 \\ 3x_1 + x_2 & \leq 6 \\ 3x_1 - 2x_2 & \geq 0 \end{cases}$$

NB. Has solutions in  $\mathbb{R}$ , but not in  $\mathbb{Z}$

## THEOREM

ILP is NP-complete

NB. The problem of finding solutions in  $\mathbb{R}$  is polynomial!



## ILP is NP-complete (II)

### THEOREM

ILP is **NP**-complete

### PROOF

- ILP  $\in$  **NP**. A certificate is a tuple of integers  $(r_1, \dots, r_n)$  that we substitute for  $x_1, \dots, x_n$  and check that  $E$  holds.
- ILP is **NP**-hard. We show that

$$3\text{CNF-SAT} \leq_P \text{ILP}$$



## ILP is NP-complete (III)

### PROOF(continued)

We show  $3\text{CNF-SAT} \leq_P \text{ILP}$  by defining for  $\varphi \in 3\text{CNF}$  with boolean atoms  $x_1, \dots, x_n$  a set of inequalities  $E_\varphi$  such that

$\varphi$  is satisfiable  $\iff E_\varphi$  has a solution in  $\mathbb{Z}$ .

- Add  $x_i \geq 0$  and  $x_i \leq 1$  to  $E_\varphi$ .
- For every clause  $l_1 \vee l_2 \vee l_3$ , add  $l_1 + l_2 + l_3 \geq 1$  to  $E_\varphi$ , where we replace negative literals  $\neg x_i$  by  $1 - x_i$ .
- We now have  $\varphi$  is satisfiable  $\iff E_\varphi$  has a solution (in  $\mathbb{Z}$ ).
- Conclusion:  $3\text{CNF-SAT} \leq_P \text{ILP}$  and so ILP is **NP-hard**.
- We have already proven that  $\text{ILP} \in \text{NP}$ , so ILP is **NP-complete**.



# Clique is NP-complete

## DEFINITION

Given an undirected graph  $G = (V, E)$  and a number  $k$ , is there a **clique** of size  $k$  in  $G$ ? A clique is a set of points  $W \subseteq V$  such that each pair of points in  $W$  is connected.

**Clique**( $G, k$ ) is the problem of deciding whether there is clique of size  $k$  in  $G$ , that is

$$\exists W \subseteq V (\#W = k \wedge \forall u, v \in W (u \neq v \rightarrow (u, v) \in E)).$$

## THEOREM

Clique is **NP**-complete.

- **Clique**  $\in$  **NP**. The certificate is the subset  $W \subseteq V$  that forms a  $k$ -clique. Checking whether  $W$  constitutes a  $k$ -clique can easily be done in polynomial time.
- We prove **Clique is NP-hard** by showing  $3\text{CNF-SAT} \leq_P \text{Clique}$ .



## Clique is NP-hard (I)

We define  $f : 3CNF \rightarrow \text{Graphs}$  such that

$$\varphi = \bigwedge_{i=1}^k C_i \text{ is satisfiable} \iff f(\varphi) \text{ has a } k\text{-clique.}$$

NB. We assume atoms occurs uniquely in a clause.

- We write  $C_i = \ell_1^i \vee \ell_2^i \vee \ell_3^i$  for each clause in  $\varphi$  ( $i = 1 \dots k$ ).
- $f(\varphi)$  is a graph with  $3k$  vertices; each vertex corresponds with a literal  $\ell_p^i$  ( $i = 1, \dots, k$ ,  $p = 1, 2, 3$ ) in  $\varphi$ .
- The edges in  $f(\varphi)$  are as follows.  
There is an edge between  $\ell_p^i$  and  $\ell_q^j$  iff  $i \neq j \wedge \ell_p^i \neq \neg \ell_q^j$ .
- We will now prove

$$\varphi \text{ has a satisfying assignment} \iff f(\varphi) \text{ has a } k\text{-clique.}$$



## Clique is NP-hard (II)

CLAIM 1:

If  $\varphi$  has satisfying assignment  $v$ , then  $f(\varphi)$  has a  $k$ -clique.

PROOF:

- From each clause we choose a literal  $\ell_p^i$  for which  $v(\ell_p^i) = 1$ .
- This gives us a  $k$ -clique in the graph  $f(\varphi)$ .

CLAIM 2:

If  $f(\varphi)$  has a  $k$ -clique  $W$ , then  $\varphi$  is satisfiable.

PROOF:

- A  $k$ -clique in  $W$  contains exactly one literal from each clause.
- If  $\ell_p^i \in W$ , then its negation does not occur in  $W$ .
- So a clique  $W$  gives us a  $v : \text{Atoms}(\varphi) \rightarrow \{0, 1\}$  that makes  $\varphi$  true.



# VertexCover is NP-complete

## DEFINITION

Given an undirected graph  $G = (V, E)$  and a number  $k$ , is there a **vertex cover** of size  $k$  in  $G$ ?

A vertex cover is a set of points  $W \subseteq V$  such that each edge has an endpoint (or both) in  $W$ .

**VertexCover** $(G, k)$  is the problem of deciding whether there is a vertex cover of size  $k$  in  $G$ , that is

$$\exists W \subseteq V (|W| = k \wedge \forall (u, v) \in E (u \in W \vee v \in W)).$$

## THEOREM

VertexCover is **NP**-complete

Proof. (1) VertexCover  $\in$  **NP**. The certificate is the subset  $W \subseteq V$  that forms a vertex cover of size  $k$ .

(2) We will now prove that VertexCover is **NP**-hard.



## VertexCover is NP-hard

We prove  $\text{Clique} \leq_P \text{VertexCover}$ .

We define  $f : \text{Graphs} \rightarrow \text{Graphs}$  such that

$G = (V, E)$  has a  $k$ -clique  $\iff f(G)$  has a  $(|V| - k)$ -vertex cover.

Define  $f(V, E) := (V, \bar{E})$  where

$$\bar{E} := \{(u, v) \mid u \neq v \wedge (u, v) \notin E\}.$$

**Claim:**  $(V, E)$  has a clique of size  $k$  iff  $(V, \bar{E})$  has a vertex cover of size  $|V| - k$ .

Proof.

$$\begin{aligned} W \text{ is a clique in } (V, E) &\iff \forall (u, v) \in W \times W (u \neq v \rightarrow (u, v) \in E) \\ &\iff \forall u \neq v ((u, v) \notin W \times W \vee (u, v) \in E) \\ &\iff \forall (u, v) \in \bar{E} (u \in V \setminus W \vee v \in V \setminus W) \\ &\iff V \setminus W \text{ is a vertex cover in } (V, \bar{E}) \end{aligned}$$



## 3Color is NP-complete

### DEFINITION

**3Color**: given an undirected graph  $G = (V, E)$ , is there a **3-coloring** of  $G$ , that is, a map  $c : V \rightarrow \{r, y, b\}$  such that  $\forall (u, v) \in E (c(u) \neq c(v))$ .

### THEOREM

3Color is **NP**-complete

Proof.

- 3Color  $\in$  **NP**. The certificate is the map  $c : V \rightarrow \{r, y, b\}$ . Checking that, for a given  $c$ , we have  $\forall (u, v) \in E (c(u) \neq c(v))$  can be done in polynomial time.
- We prove that 3Color is **NP**-hard by proving  $3\text{CNF-SAT} \leq_P 3\text{Color}$ . The construction of  $f : 3\text{CNF} \rightarrow \text{Graphs}$  will be done on the board, and also see the separate note on the webpage.