I) P and NP classes

P and NP are classes of computational problems.

P = {X | X is a decision problem for which there is a deterministic algorithm which solves P in polynomial time (in worst case)}

NP = {X | X is a decision problem for which there is a nondeterministic algorithm which solves P in polynomial time (in worst case)}

Obs1: P is included in NP (we do not know if they are equal or not ...)

Obs2: To show that a problem X is part of P it is enough to find a deterministic polynomial algorithm to solve it.
To show that a problem X is part of NP it is enough to find a nondeterministic polynomial algorithm to solve it.

Reminder: Building a nondeterministic algorithm:
1) guess a certain structure (choose instructions)
2) validate de solution (success/failure)

Obs3: NP = class of computational problems for which we can verify in polynomial time if a given input yields a TRUE output.

II) NP-hard and NP-complete problems

Remider: Karp Reductions

Let X, Y be two decision problems.

$X \propto Y$ (X polynomially reduces to Y) if:

$$\begin{tikzcd} x \in X \arrow[r, "\text{preprocess } x \text{ in poly. time}"'] & y \in Y \arrow[d, "\text{call alg. } Y \text{ once}"] \\ x' \in X & y' \in Y \arrow[l, "\text{no post. preprocessing}"'] \end{tikzcd}$$

A decision problem X is NP-hard if:
1) any problem Y in NP can be reduced in polynomial time to X

$(\forall Y \in NP, Y \propto_{Karp} X)$

A problem X is NP-complete if:
1) X is NP-hard
2) X is in NP

Obs4: Proving point 1) is hard (there are an infinity of problems in NP)...
To show that a problem X is NP-hard we only need to find a problem Y that is NP-complete and that reduces to X.

$(Y \text{ NP complete}, Y \propto_{Karp} X \Rightarrow X \text{ NP hard})$

III) Some well-known NP-complete problems

SAT
INPUT: o formulă propozițională φ (e.g., $\varphi = x_1 \wedge \neg x_2 \vee x_3$)
OUTPUT: este φ satisfiabilă?

Obs: a formula is satisfiable if there exists at least one assignment of the variables that make the formula true

Obs2: 2-CNF-SAT -> the formula is a conjuction of clauses (CNF) and each clause is a disjunction of two (2) literals

e.g.: $(x_0 \vee x_2) \wedge (x_0 \vee \neg x_3) \wedge (\neg x_1 \vee x_2)$

Obs3: 3-CNF-SAT -> the formula is a conjuction of clauses (CNF) and each clause is a disjunction of three (3) literals

e.g.: $(x_1 \vee \neg x_2 \vee x_0) \wedge (x_1 \vee \neg x_0 \vee \neg x_2)$

Obs4: SAT is NP-complete, 3-CNF-SAT is NP-complete. We also know that 2CNF-SAT is in P (There is a deterministic algorithm that solves 2-CNF-SAT in polynomial time).

***K*-COLORING**
Instance Un graf $G = (V, E)$ şi $k \in \mathbb{Z}_+$.
Question Există o colorare cu k culori a grafului G?

Obs: A graph coloring implies that no two connected vertices have the same colour

VERTEX-COVER
Domeniul problemei: Fie $G = (V, E)$ un graf. O submulţime $W \subseteq V$ se numeşte V-acoperire dacă oricare muchie din E este incidentă într-un vârf din W.

VERTEX-COVER
Instance Un graf $G = (V, E)$, un număr întreg $k \geq 0$.
Question Există o submulţime $W \subseteq V$ care este V-acoperire şi are cel mult k elemente? ($|W| \leq k$).

INDEPENDENT-SET
Domeniul problemei: Fie $G = (V, E)$ un graf. O submulţime $U \subseteq V$ se numeşte independentă dacă oricare două vârfuri din U nu sunt adiacente.

INDEPENDENT-SET
Instance Un graf $G = (V, E)$, un număr întreg $k \geq 0$.
Question Există o submulţime $U \subseteq V$ independentă cu cel puţin k elemente? ($|U| \geq k$).

CLIQUE

Input: Un graf $G = (V, E)$, $k \in \mathbb{N}$

Output: $\begin{cases} \text{DA, dc. } \exists V' \subseteq V \text{ a.î. } |V'| \geq k \text{ și } \exists \text{ muchie între } \forall \text{ 2 noduri din } V' \\ \text{NU, altfel} \end{cases}$