# Chapter 4 : Reductions and completeness

## ENSIIE - Computational complexity theory

Dimitri Watel (dimitri.watel@ensiie.fr)

2022

# Poynomial reduction and completeness

**Basic idea**

A way to solve a problem $\Pi_1$ is to transform it into another problem $\Pi_2$ we know how to solve.

If the transformation is fast and if $\Pi_2$ is fast to solve, then $\Pi_1$ is fast to solve

If the transformation is fast and if $\Pi_1$ is slow to solve, then $\Pi_2$ is slow to solve

# Karp polynomial reduction

**(Informal) definition**

Let $\Pi_1$ and $\Pi_2$, a polynomial reduction from $\Pi_1$ to $\Pi_2$ transforms every positive (resp. negative) instance of $\Pi_1$ into a positive (resp. negative) instance of $\Pi_2$ in polynomial time.

If $\Pi_1$ reduces to $\Pi_2$, then we say that $\Pi_2$ is *harder* than $\Pi_1$.

# Karp Polynomial reduction

**Definition**

Let $\Pi_1$ and $\Pi_2$ be two decision problems, a polynomial reduction from $\Pi_1 = (\mathcal{L}^1, \mathcal{L}^1_Y, \mathcal{L}^1_N)$ to $\Pi_2 = (\mathcal{L}^2, \mathcal{L}^2_Y, \mathcal{L}^2_N)$ is an algorithm $\mathcal{R}$ such that

- the complexity of $\mathcal{R}$ is polynomial
- the input of $\mathcal{R}$ is an instance $\mathcal{I}$ of $\Pi_1$ and return an instance $\mathcal{J}$ of $\Pi_2$
- $\mathcal{I} \in \mathcal{L}^1_Y \Leftrightarrow \mathcal{J} \in \mathcal{L}^2_Y$

We write $\Pi_1 \preccurlyeq \Pi_2$.

# Hardness

**$\mathcal{C}$-hard problem**

Let $\mathcal{C}$ be any complexity class (NP, EXPTIME, ...) and $\Pi_1$ be a decision problem. $\Pi_1$ is $\mathcal{C}$-Hard if, for every problem $\Pi_2$ of $\mathcal{C}$, $\Pi_2 \preccurlyeq \Pi_1$.

# Completeness

**$\mathcal{C}$-complete problem**

Let $\mathcal{C}$ be any complexity class (NP, EXPTIME, ...) and $\Pi$ be a decision problem. $\Pi$ is $\mathcal{C}$-Complete if $\Pi \in \mathcal{C}$ and if it is $\mathcal{C}$-Hard.

# Important results

## Show that a problem is hard

- If $\Pi_1$ is NP-Hard and if $\Pi_1 \preccurlyeq \Pi_2$ then $\Pi_2$ is $\mathcal{C}$-Hard.

## P et NP

- If $\Pi_2 \in$ P and $\Pi_1 \preccurlyeq \Pi_2$, then $\Pi_1 \in$ P.
- If $\Pi_2 \in$ NP and $\Pi_1 \preccurlyeq \Pi_2$, then $\Pi_1 \in$ NP.
- P $\neq$ NP $\Leftrightarrow \forall$ NP-Complete problem $\Pi$, $\Pi \notin P$.
- 3-SAT is NP-Complete.

# Turing polynomial reduction

**Definition: Turing polynomial reduction**

Let $\Pi_1$ and $\Pi_2$ bet two problems, a Turing polynomial reduction from $\Pi_1$ to $\Pi_2$ is an algorithme $\mathcal{R}$ such that

- $\mathcal{R}$ may call an (imaginary) algorithm $\mathcal{R}'$ called *oracle* that solves $\Pi_2$ in constant time;
- $\mathcal{R}$ solves $\Pi_1$ in polynomial time.

We write $\Pi_1 \preccurlyeq_T \Pi_2$.

$\Pi_1$ and $\Pi_2$ are not necessarily decision problems.

# Autres réductions

- Réduction exponentielle
- Réduction en espace polynomial
- Réduction probabiliste
- ...