

# Course Name — Cours

Ivan Lejeune

7 octobre 2025

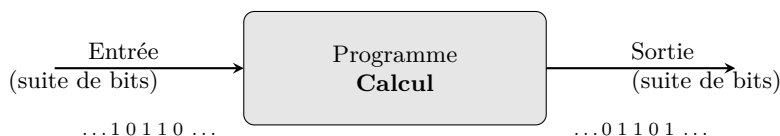
## Table des matières

Chapitre 1 — Calculabilité. . . . . 2

## Chapitre 1 — Calculabilité

On s'intéresse dans ce cours au système de boîte noire.

On attend une entrée, des calculs sont effectués, et on a une sortie.



Nos entrées et sorties correspondent à des suites de bits, on est dans le cadre discret.

On utilisera les symboles suivants :

- L'alphabet :  $\Sigma = \{0, 1\}$ ,
- L'ensemble des mots :  $\Sigma^*$ ,

On utilisera notamment les entiers à la place des mots. Pour ce faire, on veut une bijection entre  $\Sigma^*$  et  $\mathbb{N}$ . Problème ? La fonction classique

$$x_1 x_2 \dots x_n \mapsto \sum_{i=1}^n x_i 2^{i-1}$$

n'est pas une bijection car le mot 00101 et le mot 101 donnent le même entier.

On utilisera donc la méthode suivante :

Soit  $x_1, \dots, x_n$  notre suite de bits. Pour la convertir en entier, on commence par ajouter un 1 au début. Donc la chaîne de bits  $x_1 \dots x_n$  devient  $1x_1 \dots x_n$  mais problème, on ne peut pas coder le mot vide. Pour régler ce problème, on fait moins 1. Donc la chaîne de bits  $x_1 \dots x_n$  devient  $1x_1 \dots x_n$  et on fait moins 1, et voilà on a une bijection entre  $\Sigma^*$  et  $\mathbb{N}$ .

Elle ressemble à ceci :

$$x_1 x_2 \dots x_n \in \Sigma^* \mapsto \sum_{i=1}^n x_i 2^i - 1 \in \mathbb{N}$$

Pour appliquer un programme à une entrée on utilisera la notation non conventionnelle suivante :

$$[a \mid x]$$

qui représente l'exécution du programme  $a$  sur l'entrée  $x$ .

On a alors deux possibilités :

- Soit le programme s'arrête, et on note alors  $[a \mid x] \downarrow$  si le programme  $a$  sur l'entrée  $x$  s'arrête. On dit que ça converge.
- Soit le programme ne s'arrête pas, et on note alors  $[a \mid x] \uparrow$  si le programme  $a$  sur l'entrée  $x$  ne s'arrête pas. On dit que ça diverge.
- On note alors  $[a \mid x] = y$  si le programme  $a$  sur l'entrée  $x$  s'arrête et donne la sortie  $y$ .

Il existe deux autres notations conventionnelles :

- $\varphi_a(x) = y$  si  $[a \mid x] = y$  (standard historique américain),
- $U(a, x) = y$  si  $[a \mid x] = y$  (standard historique russe).

On pourra aussi utiliser  $[a \mid \cdot]$  pour désigner la fonction **partielle** suivante :

$$[a \mid \cdot] : \mathbb{N} \rightarrow \mathbb{N}$$

$$x \mapsto \begin{cases} [a \mid x] & \text{si } [a \mid x] \downarrow, \\ \text{non défini} & \text{si } [a \mid x] \uparrow. \end{cases}$$

**Définition 1.1.** Une fonction est **calculable** (ou récursive) si il existe un programme qui la calcule.

On travaillera dans l'intégralité de ce cours (ou du moins ce chapitre) sur les entiers. On pourra donc dire  $\exists a$  sans préciser que  $a$  est un entier.

**Définition 1.2.** La **fonction caractéristique** d'un ensemble  $A \subseteq \mathbb{N}$  est la fonction

$$\chi_A : \mathbb{N} \rightarrow \{0, 1\}, \quad x \mapsto \begin{cases} 1 & \text{si } x \in A, \\ 0 & \text{sinon.} \end{cases}$$

**Définition 1.3.** Un ensemble  $A \subseteq \mathbb{N}$  est **décidable** (ou **récuratif**) si sa fonction caractéristique  $\chi_A$  est calculable.

**Définition 1.4.** Un ensemble  $A \subseteq \mathbb{N}$  est **énumérable** (ou **récurivement énumérable**) si il est le domaine d'une fonction calculable.

D'autres mots pour dire la même chose :

- récurivement énumérable,
- calculatoirement énumérable,
- semi-décidable (à ne pas utiliser).

**Définition 1.5.** Le **domaine** d'une fonction  $a$  est l'ensemble

$$\text{dom}(a) = \{x \in \mathbb{N} \mid [a \mid x] \downarrow\}$$

**Théorème de Post.** Soit  $E \subseteq \mathbb{N}$ . Si  $E$  et  $\overline{E}$  sont énumérables, alors  $E$  est décidable (où  $\overline{E} = \mathbb{N} \setminus E$  est le complémentaire de  $E$  dans  $\mathbb{N}$ )

*Démonstration.* Soit  $E \subseteq \mathbb{N}$  tel que  $E$  et  $\overline{E}$  sont énumérables. Donc il existe des programmes  $a$  et  $b$  tels que  $\text{dom}[a \mid \cdot] = E$  et  $\text{dom}[b \mid \cdot] = \overline{E}$ . On veut construire un programme  $c$  qui décide  $E$ . On utilise la fonction Step de la manière suivante :

$$\text{Step}\langle a, x, t \rangle = \begin{cases} 0 & \text{si } a \text{ n'a pas terminé après } t \text{ étapes sur l'entrée } x, \\ 1 + [a \mid x] & \text{sinon.} \end{cases}$$

Ensuite, on utilise la fonction Step pour construire le programme  $c$  qui décide  $E$  :

```

c : x ↦ t ← 0
    tant que Step⟨a, x, t⟩ = 0 et Step⟨b, x, t⟩ = 0
        t ← t + 1
    si Step⟨a, x, t⟩ ≠ 0 alors return 1
    sinon return 0

```

Donc  $E$  est décidable. □