AU-DELÀ DES LANGAGES RÉGULIERS

À la fin de ce chapitre, je sais :

- ☞ expliquer les limites des langages réguliers
- ☞ montrer qu'un langage n'est pas régulier

A Limites des expressions régulières

Les langages réguliers permettent de reconnaître un motif dans un texte. Néanmoins, ils ne permettent pas de mettre un sens sur le motif reconnu : celui-ci est reconnu par l'automate mais en quoi est-il différent d'un autre mot reconnu par cet automate ? Par exemple, on peut reconnaître les mots qui se terminent par *tion* mais on ne saura pas faire la différence sémantique entre *révolution* et *abstention*.

Un autre exemple classique est l'interprétation des expressions arithmétiques : comment comprendre que $a \times b - c$ se calcule $(a \times b) - c$ et pas $a \times (b - c)$. Les deux motifs sont des expressions arithmétiques valides mais elle ne s'interprètent pas de la même manière. C'est là une des limites des langages réguliers : une fois motif reconnu, on ne peut pas l'interpréter. Pour la dépasser, il faut utiliser les notions de grammaires ⇢ HORS PROGRAMME .

Une autre question se pose : comment savoir si un langage est régulier sans pour autant exhiber un automate ? Comment caractériser formellement un langage régulier ?

B Caractériser un langage régulier

Théorème 1 — Lemme de létoile. Soit $\mathcal{L}$ un langage sur un alphabet Σ reconnu par un automate $\mathcal{A}$ à n états. Alors on a :

$$\forall w \in \mathcal{L}, |w| > n \Longrightarrow \begin{cases} \exists x, y, z \in \Sigma^*, w = xyz, |xy| \leqslant n, y \neq \epsilon \\ \textbf{et} \text{ pour une telle décomposition de } w, xy^*z \subseteq \mathcal{L} \end{cases} \quad (1)$$

Démonstration. Soit w un mot reconnu par l'automate $\mathcal{A}$ à n états de longueur m. Il existe un chemin dans $\mathcal{A}$ qui part de l'état initial q_0 et s'achève sur un état accepteur q_m.

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} \dots \xrightarrow{a_m} q_m$$

En numérotant de manière incrémentale les états de 0 à m, on a nécessairement $m > n$. D'après le principe des tiroirs, comme l'automate ne possède que n états, ce chemin repasse par certains états. Prenons le premier état par lequel le chemin repasse et notons le i. Il existe donc deux entiers i et j tels que $0 < i < j \leqslant n < m$ et $q_i = q_j$, c'est-à-dire il existe un cycle de longueur $j - i$ sur le chemin. Comme il s'agit du premier état par lequel on repasse, les état q_0 jusqu'à q_{j-1} sont tous distincts.

On choisit alors de poser $x = a_1 \dots a_{i-1}$, $y = a_i \dots a_{j-1}$ et $z = a_j \dots a_m$. On remarque que $w = xyz$ et que x et xy vérifient les propriétés du lemme de l'étoile car y n'est pas vide et $|xy| \leqslant n$. Il reste à montrer que $xy^*z \subseteq \mathcal{L}$. Comme le chemin reconnaissant y est un cycle (cf. figure 1), on peut le parcourir autant de fois que l'on veut, 0 ou k fois, le mot sera toujours reconnu par l'automate. ■

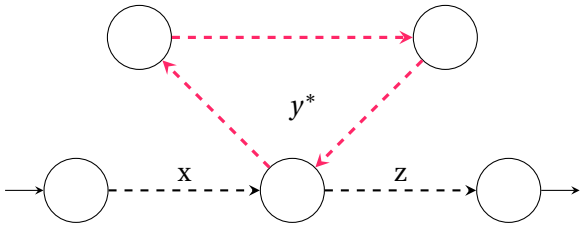

FIGURE 1 – Illustration du lemme de l'étoile : si le nombre de lettres d'un mot reconnu w est plus grand que le nombre d'états de l'automate n, alors il existe une boucle sur laquelle on peut itérer.

Théorème 2 — Principes des tiroirs. Si $n+1$ éléments doivent être placés dans n ensembles, alors il existe au moins un ensemble qui contient au moins 2 éléments. Autrement dit, si E et F sont deux ensembles finis tels que $|E| > |F|$, alors il n'existe aucune application injective de E dans F.

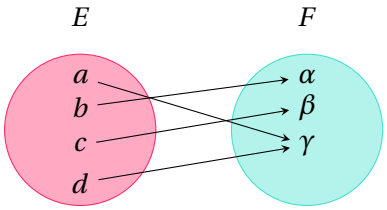

FIGURE 2 – Illustration du principe des tiroirs : on ne peut pas ranger les éléments de E dans les tiroirs de F sans en mettre deux dans un tiroir.

R Le lemme de l'étoile est parfois appelé le lemme de l'itération car on peut itérer autant de fois que l'on veut y.

Vocabulary 1 — Pumping lemma ⇝ Lemme de l'étoile

■ **Définition 1 — Constante d'itération et facteur itérant.** Soit $\mathcal{L}$ un langage régulier. D'après le lemme de l'itération, il existe un entier naturel N tel que chaque mot w de $\mathcal{L}$ tel que $|w| \geqslant N$ possède au moins un facteur non vide y pouvant être itéré.

On dit alors que N est une constante d'itération pour le langage $\mathcal{L}$ et que y est un facteur itérant.

■ **Définition 2 — Constante d'itération minimale.** Il s'agit de la plus petite constante d'itération d'un langage $\mathcal{L}$.

■ **Exemple 1 — Exemples de constantes minimales d'itération.** On considère les langages dénotés par des expressions régulières et on calcule la constante minimale d'itération du langage :

ab ⟶ 3, car le seul mot reconnu par le langage est ab. On ne peut pas itérer ce mot. De plus, il n'y a pas de mots de longueur supérieure à 2. C'est pourquoi, tous les mots (qui n'existent pas) de longueur supérieure ou égale à 3 peuvent être itérés.

aab^* ⟶ 3, car le plus petit mot du langage est aa mais il ne peut pas être itéré. Soit un mot w de longueur 3. On peut le décomposer comme suit : $w = xyz$, $x = aa$, $y = b$ et $z = \epsilon$. Cette décomposition satisfait le lemme de l'étoile.

$(a|b)^*$ ⟶ 1. On observe que $\epsilon \in \mathcal{L}_{ER}((a|b)^*)$, cependant il ne peut pas être itéré. Donc, la constante minimale ne peut pas être égale à zéro. Soit w un mot de longueur 1. Il vaut a ou b. Dans le premier cas, on peut choisir la décomposition $w = xyz$ avec $x = \epsilon$, $y = a$, $z = \epsilon$, dans le deuxième $x = \epsilon$, $y = b$, $z = \epsilon$. Dans les deux cas, la décomposition satisfait le lemme de l'étoile.

R Il faut remarquer que le lemme de l'étoile peut être vérifié par un langage non régulier. C'est pourquoi, la plupart du temps, on utilise le lemme de l'étoile pour montrer qu'un langage

n'est pas régulier : s'il ne le vérifie pas, il n'est pas régulier.

C Les langages des puissances

■ **Définition 3 — Langage des puissances.** On appelle langage des puissances le langage défini par :

$$\mathcal{L}_p = \{a^n b^n, n \in \mathbb{N}\} \tag{2}$$

Théorème 3 — Le langage des puissances n'est pas régulier.

Démonstration. Par l'absurde en utilisant le lemme de l'étoile.

Supposons que $\mathcal{L}_p$ soit régulier. Alors il vérifie le lemme de l'étoile. Soit n un entier naturel, une constante d'itération de ce langage. Considérons $w = a^n b^n \in \mathcal{L}_p$. On a bien $|w| = 2n \geqslant n$. On peut donc appliquer le lemme de l'étoile à w.

Soient x, y et z, les mots formant la décomposition de $w = xyz$. D'après le lemme de l'étoile, $|xy| \leqslant n$. Il existe donc des entiers naturels i et $j > 0$ tels que $x = a^i$, $y = a^j \neq \epsilon$, $xy = a^{i+j}$ et $i + j \leqslant n$. On peut réécrire la décomposition comme suit : $w = xyz = a^i a^j a^{n-i-j} b^n$, c'est-à-dire que $z = a^{n-i-j} b^n$.

Cette décomposition de w est telle qu'on peut itérer sur y et appartenir toujours au langage. Donc le mot $xy^2z = a^i a^{2j} z = a^i a^{2j} a^{n-i-j} b^n$ devrait appartenir à $\mathcal{L}_p$. Or ce n'est manifestement pas le cas car $i + 2j + n - i - j = n + j > n$ car $j > 0$. C'est pourquoi $\mathcal{L}_p$ n'est pas un langage régulier.

NB : on aurait pu également étudier le mot xy^0z et aboutir à la même conclusion. ■

(R) C'est un résultat à connaître car on peut s'en servir pour démontrer la non régularité d'autres langages. La démonstration est également typique de l'utilisation du lemme de l'étoile.