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 l'étoile. Pour tout langage **régulier** $\mathcal{L}$ sur une alphabet Σ, on a :

$$\exists n \geq 1, \forall w \in \mathcal{L}, |w| \geq n \Rightarrow \exists x, y, z \in \Sigma^*, w = xyz \wedge \left(y \neq \epsilon \wedge |xy| \leq n \wedge \mathcal{L}_{ER}(xy^* z) \subseteq \mathcal{L}\right) \quad (1)$$

Démonstration. Soit $\mathcal{L}$ un langage régulier sur un alphabet Σ. D'après le théorème de Kleene, il existe un automate fini $\mathcal{A}$ à n états qui reconnaît $\mathcal{L}$. 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.

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

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 $x = a_1 \ldots a_{i-1}$, $y = a_i \ldots a_{j-1}$ et $z = a_j \ldots a_m$. 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 **??**), on peut le parcourir autant de fois que l'on veut, 0 ou k fois, le mot sera toujours reconnu par l'automate. ■

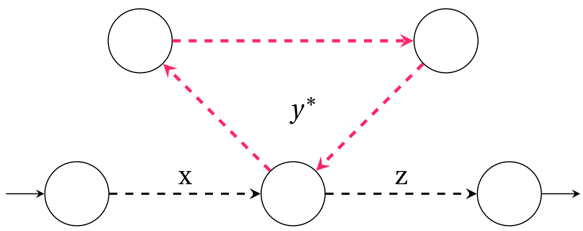

FIGURE 1 – Illustration du lemme de l'étoile : si le nombre de lettres d'un mot reconnu w est plus grand que le nombre d'états de l'automate n, alors il existe une boucle sur laquelle on peut itérer.

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.

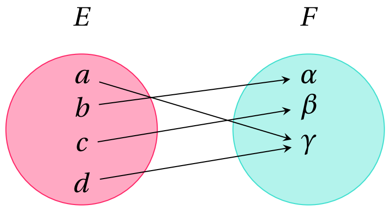

FIGURE 2 – Illustration du principe des tiroirs : on ne peut pas ranger les éléments de E dans les tiroirs de F sans en mettre deux dans un tiroir.

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 $\rightsquigarrow$ Lemme de l'étoile

R Il faut remarquer que le lemme de l'étoile peut être vérifié par un langage non régulier : il s'agit d'une condition **nécessaire pour être régulier mais pas suffisante**. C'est pourquoi, la plupart du temps, on utilise le lemme de l'étoile dans sa forme contraposée 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 1 — 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** $\mathcal{A}$ **un automate à** n **état qui reconnaît** $\mathcal{L}$. Considérons le mot $w = a^n b^n \in \mathcal{L}_p$. On a bien $|w| = 2n \geqslant n$. On peut donc appliquer le lemme de l'étoile à w.

D'après ce lemme, il existe une décomposition de w en xyz qui vérifie $|xy| \leqslant n$ et $y \neq \epsilon$. Soit i et j deux entiers naturels tels que $i + j \leqslant n$ et $j > 0$. Cette décomposition de w est nécessairement de la forme générale $w = a^i a^j a^{n-i-j} b^n = xyz$, avec $x = a^i$, $y = a^j$ et $z = a^{n-i-j} b^n$.

Les conditions du lemme sont vérifiées et il est donc possible d'itérer sur y : un tel mot appartient 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. ■

R Le théorème **??** un résultat théorique important à connaître car :

- on peut s'en servir pour démontrer la non régularité d'autres langages en utilisant la stabilité de l'intersection pour les langages réguliers.
- la démonstration est canonique, c'est-à-dire typique de l'utilisation du lemme de l'étoile.