EXPRESSIONS RÉGULIÈRES

À la fin de ce chapitre, je sais :

- ☞ expliquer les définitions inductives des expressions régulières et des langages réguliers
- ☞ utiliser la sémantique des langages réguliers
- ☞ utiliser les identités remarquables sur les expressions régulières pour simplifier une expression
- ☞ construire un arbre représentant une expressions régulière

On peut décrire un langage de différentes manières :

- par compréhension : $\mathcal{L} = \{u \in \Sigma^*, |u| = 0 \bmod 2\}$, c'est-à-dire en utilisant une propriété spécifique au langage. Néanmoins, ceci n'est pas toujours évident et, de plus, cela n'implique pas de méthode concrète pour construire des mots de ce langage.
- pour les langages réguliers, par une expression régulière ou une grammaire régulière.

Les expressions régulières sont un moyen de caractériser de manière inductive certains langages et offre des **règles pour construire les mots** de ces langages. Tout comme en couture, on dit qu'elles constituent un **patron de conception**. À chaque expression régulière est associé un langage, c'est à dire l'ensemble des mots qu'elle permet d'élaborer. L'avantage principal des expressions régulières est qu'elles fournissent en plus une vision algébrique des langages réguliers et permettent donc le calcul.

R Régulier ou rationnel? Ces deux adjectifs sont employés de manière équivalente en français dans le cadre de la théorie des langages. Il existe des arguments en faveur de l'utilisation de chacun :

- rationnel : c'est le mot historique utilisé en France. Sa racine latine évoque le calcul et il s'agit donc des langages que l'on peut calculer.
- régulier : c'est un anglicisme mais dont la racine latine [a] évoque la conformation à une norme ou un règle. Il s'agit donc des langages que l'on peut décrire par une règle.

Ces deux adjectifs sont donc cohérents et utilisables en français pour décrire les langages et les expressions qui font l'objet de ce chapitre. L'un est plus pragmatique que l'autre!

a. via les anglo-normands et Guillaume le conquérant

A Définition des expressions régulières

■ **Définition 1 — Syntaxe des expressions régulières.** L'ensemble des expressions régulières $\mathcal{E}_R$ sur un alphabet Σ est défini inductivement par :

(Base) $\{\emptyset, \epsilon\} \cup \Sigma \in \mathcal{E}_R$,

(Règle de construction (union)) $\forall e_1, e_2 \in \mathcal{E}_R, e_1 \mid e_2 \in \mathcal{E}_R$

(Règle de construction (concaténation)) $\forall e_1, e_2 \in \mathcal{E}_R, e_1 e_2 \in \mathcal{E}_R$,

(Règle de construction (fermeture de Kleene)) $\forall e \in \mathcal{E}_R, e^* \in \mathcal{E}_R$.

R Cette définition peut s'exprimer ainsi : les expressions régulières sont constituées à la base de l'ensemble vide, du mot vide et des lettres de l'alphabet. On peut construire d'autres expressions régulières à partir de ces éléments de base en appliquant un nombre fini de fois les opérateurs d'union, de concaténation et de fermeture de Kleene.

R Le symbole de la concaténation est omis pour plus de lisibilité.

R L'utilisation des parenthèses est possible et souhaitable pour réduire les ambiguïtés. La priorité des opérateurs est l'étoile de Kleene, la concaténation puis l'union. Par défaut, l'associativité des opérateurs est choisie à gauche.

Par exemple, pour une alphabet $\{a, b, c\}$, on peut écrire :

- $a \mid b \mid c$ à la place de $a \mid (b \mid c)$
- $a \mid cb^*$ à la place de $a \mid (c(b^*))$

■ **Exemple 1 — Quelques expressions régulières.** Voici quelques expressions régulières pratiques dans la vie de tous jours :

$(P|MP|PC)SI$ sur l'alphabet latin désigne l'ensemble $\{PSI, MPSI, PCSI\}$

$(19|20)00$ sur l'alphabet $\Sigma = \{0,1,2,3,4,5,6,7,8,9\}$ désigne l'année 1900 ou l'année 2000.

1010101^* sur l'alphabet $\Sigma = \{0,1\}$ désigne l'ensemble des mots binaires préfixés par 101010 et se terminant par des 1.

$\Sigma^* k \Sigma^+ q \Sigma^*$ il y a un mot de sept lettres engendré par cette expression régulière en français. Comme quoi, il y a toujours de l'espoir au Scrabble.

■ **Exemple 2 — D'autres expressions régulières.** Ces exemples sont typiques des expressions utilisées lors des épreuves de concours.

- Σ^* tous les mots
- $a\Sigma^*$ les mots commençant par a

- $\Sigma^* a$ les mots finissant par a
- $a^*|b^*$ les mots ne comportant que des a ou que des b
- $(ab^*a|b)^*$ les mots comportant un nombre pair de a
- $(aa|b)^*$ les mots comportant des blocs de a de longueur paire
- $(b|ab)^*(a|\epsilon)$ les mots ne comportant pas deux fois a consécutivement

R Les expressions régulières sont très puissantes et il est illusoire d'imaginer qu'on peut les exprimer simplement avec des mots. C'est très rarement possible, uniquement sur des cas simples comme ci-dessus.

■ **Définition 2 — Langage dénoté par une expression régulière.** Pour toute expression régulière e, on note $\mathcal{L}_{ER}(e)$ le langage unique qui dénote cette expression. C'est en fait le résultat de l'application $\mathcal{L}_{ER} : \mathcal{E}_R(\Sigma^*) \longrightarrow \mathcal{P}(\Sigma^*)$ qui, à une expression régulière, fait correspondre l'ensemble des mots (le langage) engendré par cette expression régulière.

■ **Définition 3 — Sémantique des expressions régulières.** L'interprétation des expressions régulières en termes de langages, la sémantique d'une expression régulière, permet de définir inductivement l'ensemble des langages dénotés par une expression régulière ainsi :

(Base (i)) $\mathcal{L}_{ER}(\emptyset) = \{\} = \emptyset$,

(Base (ii)) $\mathcal{L}_{ER}(\epsilon) = \{\epsilon\}$,

(Base (iii)) $\forall a \in \Sigma, \mathcal{L}_{ER}(a) = \{a\}$,

(Règle de construction (i union)) $\forall e_1, e_2 \in \mathcal{E}_R, \mathcal{L}_{ER}(e_1 \mid e_2) = \mathcal{L}_{ER}(e_1) \cup \mathcal{L}_{ER}(e_2)$,

(Règle de construction (ii concaténation)) $\forall e_1, e_2 \in \mathcal{E}_R, \mathcal{L}_{ER}(e_1 e_2) = \mathcal{L}_{ER}(e_1).\mathcal{L}_{ER}(e_2)$,

(Règle de construction (iii fermeture de Kleene)) $\forall e \in \mathcal{E}_R, \mathcal{L}_{ER}(e^*) = \mathcal{L}_{ER}(e)^*$.

Théorème 1 — Un langage $\mathcal{L}$ est régulier si et seulement s'il existe une expression régulière e telle que $\mathcal{L}_{ER}(e) = \mathcal{L}$.

R Les apparences sont parfois trompeuses. Soit $\Sigma = \{a, b\}$ un alphabet et $\mathcal{L}$ le langage défini par $\mathcal{L} = \{a^n b^n, n \in [\![0, n]\!]\} = \{\epsilon, ab, aabb, aaabbb, \ldots\}$. $\mathcal{L}$ n'est pas un langage régulier. Si la formule entre crochets dénote bien un langage, un ensemble de mots, on ne peut cependant pas exprimer l'ensemble de ces mots par une expression régulière. Par exemple, $a^* b^*$ est une expression régulière dont le langage associé dénote l'ensemble $\mathcal{L}$ ainsi que d'autres mots comme $\{a, b, aaa, bbb, aab, abb, abbb, \ldots\}$. Le lemme de l'étoile (cf. chapitre suivant) permet de démontrer que $\mathcal{L}$ n'est pas un langage régulier.

B Définition des langages réguliers

Il est également possible de donner une définition inductive directe des langages réguliers.

■ **Définition 4 — Ensemble des langages réguliers.** L'ensemble des langages réguliers $\mathcal{L}_{ER}$ sur un alphabet Σ est défini inductivement par :

(Base (i)) $\emptyset \in \mathcal{L}_{ER}$,

(Base (ii)) $\{\epsilon\} \in \mathcal{L}_{ER}$,

(Base (iii)) $\forall a \in \Sigma, \{a\} \in \mathcal{L}_{ER}$,

(Règle de construction (i)) $\forall \mathcal{L}_1, \mathcal{L}_2 \in \mathcal{L}_{ER}, \mathcal{L}_1 \cup \mathcal{L}_2 = \mathcal{L}_{ER}$

(Règle de construction (ii)) $\forall \mathcal{L}_1, \mathcal{L}_2 \in \mathcal{L}_{ER}, \mathcal{L}_1.\mathcal{L}_2 = \mathcal{L}_{ER}$,

(Règle de construction (iii)) $\forall \mathcal{L} \in \mathcal{L}_{ER}, \mathcal{L}^* = \mathcal{L}_{ER}$.

R Cette définition peut s'exprimer ainsi : les langages réguliers sont constitués à la base de l'ensemble vide, du langage ne contenant que le mot vide et des lettres de l'alphabet. On peut construire d'autres langages réguliers à partir de ces éléments de base en appliquant un nombre fini de fois les opérateurs d'union, de concaténation et de fermeture de Kleene.

■ **Définition 5 — Opérations régulières sur les langages.** L'union, la concaténation et la fermeture de Kleene sont les trois opérations régulières sur les langages.

R On note que l'intersection et la complémentation ne sont pas des opérations régulières.

Théorème 2 — Stabilité des langages réguliers. Les langages réguliers sont stables pour les opérations d'**union**, de **concaténation** et de **fermeture de Kleene**. Cela signifie que l'union, l'intersection ou la fermeture de Kleene de langages réguliers est un langage régulier.

Démonstration. Conséquence directe de la définition inductive. ■

C Identités remarquables sur les expressions régulières

Le tableau 1 recense quelques identités remarquables à connaître sur les expressions régulières. Leurs démonstrations s'appuient sur la sémantique des expressions régulières et les opérations sur les langages. Elles constituent un excellent exercice.

Démonstration. Par exemple, démontrons que $e|\emptyset = e$. D'après la sémantique des expressions

régulières, on a :

$$\begin{aligned}\mathcal{L}_{ER}(e|\emptyset) &= \mathcal{L}_{ER}(e) \cup \mathcal{L}_{ER}(\emptyset) && (1)\\ &= \mathcal{L}_{ER}(e) \cup \emptyset && (2)\\ &= \mathcal{L}_{ER}(e) && (3)\end{aligned}$$

car l'ensemble vide est l'élément neutre de l'union ensembliste. ■

Expression régulière	**Équivalent**	**Raison**
$e\|\emptyset$	e	$\emptyset$ est l'élément neutre de l'union ensembliste
$e\emptyset$	$\emptyset$	Déf. de la concaténation, $\emptyset$ seul élément commun à $\mathcal{L}_{ER}(e)$ et $\{\emptyset\}$
$e\epsilon$	e	Déf. de la concaténation, ϵ élément neutre de la concaténation
$(e\|f)\|g$	$e\|f\|g$	Associativité de l'union ensembliste
$(e.f).g$	$e.f.g$	Associativité de la concaténation sur les langages
$e(f\|g)$	$ef\|eg$	Distributivité de la concaténation sur l'union
$(e\|f).g$	$eg\|fg$	Idem
$e\|f$	$f\|e$	Commutativité de l'union ensembliste
e^*	$\epsilon\|ee^*$	Définition de l'étoile de Kleene
e^*	$\epsilon\|e^*e$	Idem
$(\emptyset)^*$	ϵ	Définition de l'étoile de Kleene et des puissances d'un langage
$e\|e$	e	Idempotence
$(e^*)^*$	e^*	Idempotence

TABLE 1 – Identités remarquables des expressions régulières.

■ **Exemple 3 — Exemple de calcul sur les expressions régulières.** À l'aide des identités remarquables du tableau 1, il est possible de transformer des expressions régulières, de les simplifier par calcul. Voici un exemple sur $\Sigma = \{a, b\}$:

$$bb^*(a^*b^*|\epsilon)b = bb^*a^*b^*b \tag{4}$$

Cette expression est généralement exprimée ainsi $b^+a^*b^+$ et elle désigne l'ensemble des mots comportant au moins deux lettres qui commencent et terminent par un b. On notera que le langage associé $\mathcal{L}_{ER}(b^+a^*b^+)$ ne contient pas le mot vide.

D Arbre associé à une expression régulière

De part sa nature inductive, une expression régulière peut naturellement être représentée sous la forme d'un arbre dont les feuilles sont les éléments de bases et les nœuds les opérateurs réguliers. Ces arbres permettent de prouver par induction de nombreuses propriétés des expressions régulières.

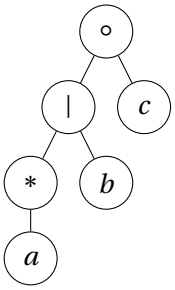

FIGURE 1 – Arbre associé à l'expression régulière $(a^*|b)c$

E Expressions régulières dans les langages --→ HORS PROGRAMME

Les expressions régulières sont intensivement utilisées par les informaticiens en pratique pour le test, le filtrage ou la transformation de l'information. Tous les langages évolués proposent une interface qui permet d'utiliser les expressions régulières.

Concrètement, les expressions rationnelles s'appuient sur un ensemble de caractères et de métacaractères qui permettent d'abstraire un motif de chaîne de caractères.

■ **Exemple 4 — Ensemble des lignes d'un fichier qui commencent au moins par un chiffre et qui se termine par Z.** On peut utiliser l'expression régulière suivante : `"^[0-9]+.*Z$"` où :

- `^` désigne le début d'une ligne
- `[0-9]` désigne l'ensemble des caractères 0,1,2,3,4,5,6,7,8,9
- `+` signifie au moins une occurrence du caractère précédent
- `.` symbolise n'importe quel caractère
- `*` signifie 0 ou un nombre quelconque d'occurrences du caractère précédent
- `Z` est le caractère Z
- `$` signifie la fin d'une ligne

■ **Définition 6 — Métacaractère.** Un métacaractère est un caractère qui a une signification abstraite, autre que son interprétation littérale.

Selon le langage utilisé, on pourra utiliser les classes ou les séquences spéciales (mode Perl).

■ **Exemple 5 — Les expressions célèbres.** Voici une liste non exhaustive d'expressions régulières couramment utilisées :

- `\d+` les entiers naturels
- `-?\d+` les entiers
- `(\d{2}|\s)]{5}` un numéro de téléphone avec des espaces

Métacaractère	Signification
`^`	début de la chaîne
`$`	fin de la chaîne
`.`	n'importe quel caractère sauf retour à la ligne
`*`	0 ou plusieurs fois le caractère précédent
`+`	au moins 1 fois le caractère précédent
`?`	0 ou 1 fois le caractère précédent
`|`	alternative (union)
`()`	groupement, motif
`[]`	ensemble de caractères
`{}`	nombre de répétition du motif

TABLE 2 – Liste des métacaractères

Expression	Signification
`e*`	zéro ou plusieurs e
`e+`	un ou plusieurs e
`e?`	0 ou 1 fois e
`e{m}`	exactement m fois e
`e{m,}`	au moins m fois e
`e{m,n}`	au minimum m fois a et au maximum n fois e

TABLE 3 – Répétition des motifs

Séquence	Signification
`\t`	tabulation
`\n`	nouvelle ligne
`\r`	retour chariot
`\b`	bordure de mot
`\B`	pas en bordure de mot
`\d`	correspond à n'importe quel chiffre [0-9]
`\D`	correpond à n'importe quel caractère sauf un chiffre
`\w`	correspond à n'importe quel mot [0-9a-zA-Z_]
`\W`	correspond à n'importe quelle séquence qui n'est pas un mot
`\s`	correspond à n'importe quel espace (espace, tabulation, nouvelle ligne)
`\S`	correspond à n'importe quel caractère qui n'est pas un espace

TABLE 4 – Séquences spéciales (mode Perl disponible en Python)

Classe	Signification
`[:alnum:]`	correspond aux caractères alphabétiques et numériques. [A-Za-z0-9]
`[:alpha:]`	correspond aux caractères alphabétiques. [A-Za-z]
`[:blank:]`	correspond à un espace ou à une tabulation
`[:cntrl:]`	correspond aux caractères de contrôle
`[:digit:]`	correspond aux chiffres [0-9]
`[:graph:]`	caractères graphiques affichables
`[:lower:]`	correspond aux caractères alphabétiques minuscules. [a-z]
`[:print:]`	caractères imprimables
`[:space:]`	correspond à tout espace blanc (espace, tabulation, nouvelle ligne)
`[:upper:]`	correspond à tout caractère alphabétique majuscule. [A-Z]
`[:xdigit:]`	correspond aux chiffres hexadécimaux. [0-9A-Fa-f]

TABLE 5 – Classes (mode expressions régulières étendues de grep)

- `[A-Za-z0-9_-]{3,16}` les noms des utilisateurs d'un système (login),
- `[[:alnum:]_-^@#!$% ]{8,42}` les mots de passe,
- `([a-z0-9_.-]+)@([0-9a-z.-]+).([a-z.]{2,24})` les adresses email