1. Expressions régulières

1. Ci-dessous vous trouvez des exemples d'expressions régulières valides et une explication informelle,
   (a) $1^*(01^*01^*)^*$
   La deuxième partie de cette expression régulière s'assure que les 0 apparaissent par deux dans les chaînes, ces derniers pouvant être séparées par un nombre arbitraire de 1. Notons que ϵ est reconnu, de même que les chaînes constituée de 1 uniquement car zéro est un nombre pair[1].
   (b) $(0^*10^*)(10^*10^*)^*$
   La première partie de l'expression régulière reconnaît les chaînes constituées d'un nombre arbitraire de 0 et d'un unique 1. Il reste donc, dans la deuxième partie à reconnaître un nombre pair de 1 potentiellement séparés par des nombres arbitraires de 0 et le total sera impair. Remarquez que la chaîne ϵ n'est pas reconnue car $\#1(\epsilon) = 0$ qui est pair.
   (c) $\left(1^*(01^*01^*)^*\right) + \left((0^*10^*)(10^*10^*)^*\right)$
   Ce langage est l'union des deux précédents. L'opérateur $+$ des expressions regulières représente justement l'union de deux langages.
   (d) $(0^*1^*00)^*1^*0^*$
   Dans cette expression régulière, nous nous assurons que lorsqu'un 1 est suivi d'un 0, soit il existe au moins un autre 0, soit nous avons atteint la fin de la chaîne. La sous-chaîne 101 n'apparaîtra alors pas.
   (e) $(01 + 10)^*$
   La répétition des 01 et 10 assure que le nombre de 0 et de 1 et égal. Elle assure aussi qu'il n'y aura jamais trois 0 ou 1 successifs. C'est cette dernière propriété qui permet de démontrer que la différence de 0 et de 1 pour les préfixes d'une chaîne reconnue est toujours plus petite que 2.
2. Les propriétés des chaînes reconnues par ces expressions régulières sont :
   (a) $(\epsilon + 1)(00^*1)^*0^*$
   Les chaînes reconnues par cette expressions n'ont pas de 1 adjacents. Une formulation équivalente et plus simple de cette propriété est $(0 + 10)^*(\epsilon + 1)$.
   (b) $(0^*1^*)^*000(0 + 1)^*$
   Les expressions $(0^*1^*)^*$ et $(0 + 1)^*$ ne reconnaissant rien d'autre que tous les mots sur l'alphabet Σ, cette expression régulière reconnaît donc toutes les chaînes qui ont 000 comme sous-chaîne.

2. Preuves sur les expressions régulières

Lemme 2.1 *Pour tout $a \in \Sigma$, et tout $w \in \Sigma^*$ tel que $\#a(w) = 0$ nous avons $w \in (\Sigma \setminus \{a\})^*$.* □

[1] Rappel : m pair $\Leftrightarrow \exists n \in \mathbb{N} : m = 2n$

Lemme 2.2 *Pour tout $a \in \Sigma$, et tout $w \in \Sigma^*$ tel que $\#a(w) > 1$ nous avons*

$$\exists x, y \in \Sigma^* : w = xay \wedge \#a(x) = \#a(w) - 1$$

□

Posons $L_e = L(1^*(01^*01^*)^*)$. Nous souhaitons démontrer la proposition suivante,

$$L_e = \{w \in \{0,1\}^* \mid \#0(w) \text{ est pair}\}$$

ce qui revient à démontrer, par définition de l'égalité sur les ensembles,

$$\forall w' : w' \in L_e \Leftrightarrow w' \in \{w \in \{0,1\}^* \mid \#0(w) \text{ est pair}\}$$

Preuve. Nous montrons d'abord l'implication de gauche à droite puis de droite à gauche.

1. ($\Rightarrow$) Supposons que $w \in L_e$. Nous voulons montrer que $\#0(w)$ est pair.

   Par définition de la concatenation des expressions régulières, nous savons que w peut être décomposé en $w = xy$ tel que $x \in L(1^*)$ et $y \in L((01^*01^*)^*)$. Par définition de l'opérateur * des expressions régulières et de la fermeture d'un langage cela signifie qu'il existe $l \in \mathbb{N}$ tel que $x = 1^l$ et $m \in \mathbb{N}$ tel que $y = z_1 \cdot \ldots \cdot z_m$ avec $z_i \in L(01^*01^*)$. En décomposant chaque z_i de manière idoine nous savons que pour $i \in \{1, \ldots, m\}$ il existe $a_i, b_i, \in \mathbb{N}$ tels que $z_i = 01^{a_i}01^{b_i}$. Par définition de $\#0(w)$ et par la décomposition de w, nous avons

   $$\#0(w) = \#0(x \cdot y) = \#0(1^l) + \#0(z_1 \cdot \ldots \cdot z_m) = l\#0(1) + \sum_{i=1}^{m} \#0(z_i)$$
   $$= \sum_{i=1}^{m} \#0(0) + a_i\#0(1) + \#0(0) + b_i\#0(1) = 2m$$

   m étant un nombre naturel, $\#0(w)$ est pair.

2. ($\Leftarrow$) Supposons que w est tel que $\#0(w)$ est pair. Nous voulons montrer que $w \in L_e$.

   Rappelons que par définition de la fermeture et de la concaténation des langages nous avons $L(\alpha) \subset L(\beta^*\alpha)$, $L(\alpha) \subset L(\alpha\beta^*)$ et que $L(\alpha^*) \subset L(\alpha^*\alpha)$.
   Nous démontrons la proposition suivante par récurrence sur $m \in \mathbb{N}$,

   $$\forall m \in \mathbb{N}, \forall w \in \Sigma^* : \#0(w) = 2m \Rightarrow w \in L_e$$

   $m = 0$

   Prenons une chaîne $w \in \Sigma^*$ quelconque telle que $\#0(w) = 0$. Par le lemme 2.1 nous avons $w \in \{1\}^*$. Or $\{1\}^* = L(1^*) \subset L(1^*) \cdot L((01^*01^*)^*) = L_e$ donc $w \in L_e$.

   $m \Rightarrow m + 1$

   Par l'hypothèse d'induction nous savons que

   $$\forall w' \in \Sigma^* : \#0(w) = 2m \Rightarrow w' \in L_e$$

   Prenons une chaîne quelconque w telle que $\#0(w) = 2(m + 1)$. Nous avons $\#0(w) = 2(m + 1) > 1$ et par le lemme 2.2 nous savons donc qu'il existe x et y tels que $w = xay$ avec $\#0(x) = 2(m+1)-1 > 1$. Appliquons le lemme 2.2 sur x. Au final, nous savons qu'il existe x',y' et y tels que $w = x'0y'0y$ avec $\#0(x') = 2m$, $\#0(y') = 0$ et $\#0(y) = 0$. Par le lemme 2.1 nous avons $y, y' \in L(1^*)$ et par conséquent $z = 0y'0y \in L(01^*01^*)$. Par induction nous savons que $x' \in L_e = L(1^*) \cdot L((01^*01^*)^*)$. Nous avons $w = y'z \in L(1^*) \cdot L((01^*01^*)^*) \cdot L(01^*01^*) \subset L_e$ donc $w \in L_e$.

   Nous avons donc pour tout $w \in \Sigma^*$ avec $\#0(w)$ pair, $w \in L_e$

De (1) et (2) nous déduisons la proposition à démontrer. ■

3. Langages à fermeture finie

Lemme 3.3 *Soit deux ensemble A et B finis tels que $A \subseteq B$, alors $\|A\| \leq \|B\|$.* □

Proposition 3.4 *Pour tout alphabet Σ, il n'existe que que deux langages différents L_1 et L_2 sur Σ dont la fermeture L_1^* et L_2^* est finie.* □

Preuve. Nous montrons d'abord qu'il existe deux langages dont la fermeture est finie puis que tout langage différent de ces deux langages possède une fermeture infinie.

1. Il existe deux langages L_1 et L_2 différents dont la fermeture est finie.

   Nous procédons par construction en trouvant ces deux langages et en démontrant que leur fermeture est finie. Rappellons (voir cours, session 2) qu'un ensemble est fini s'il existe un nombre naturel égal au nombre d'éléments de cet ensemble,

   $$A \text{ fini} \Leftrightarrow \exists n \in \mathbb{N} : \|A\| = n$$

   Par définition de la fermeture d'un langage L (voir cours, session 3), L^* est donc fini si

   $$\exists n \in \mathbb{N} : \|L^*\| = \|\bigcup_{i \geq 0} L^i\| = n$$

   Étant donné un alphabet Σ quelconque, considérons,

   (a) Le langage $L_1 = \emptyset$.
   En fait $L_1^* = \bigcup_{n \geq 0} L_1^n = \{\epsilon\}$. Pour démontrer cela, nous montrons d'abord par induction sur n que pour tout $n \geq 1$, $L_1^n = \emptyset$

   $n = 1$

   Par définition nous avons $L_1^1 = L_1^0 L_1$, ce qui est égal, par définition de la concatenation de deux langages à l'ensemble $\{xy \mid x \in L_1^0 \wedge y \in L_1\}$. Cependant, ce dernier ensemble est vide car étant donné que $L_1 = \emptyset$, il n'existe pas de $y \in L_1$. Dès lors $L_1^1 = \emptyset$.

   $n \Rightarrow n + 1$

   Supposons que $L_1^n = \emptyset$. Nous avons $L_1^{n+1} = L_1^n L_1$, ce qui est égal par définition de la concatenation de deux langages à l'ensemble $\{xy \mid x \in L_1^n \wedge y \in L_1\}$. Cependant, ce dernier ensemble est vide car étant donné que $L_1 = \emptyset$, il n'existe pas de $y \in L_1$. Dès lors $L_1^{n+1} = \emptyset$.

   Nous avons donc, $L_1^* = \bigcup_{n \geq 0} L_1^n = L_1^0 \cup \bigcup_{n \geq 1} \emptyset = \{\epsilon\}$ et $\|\{\epsilon\}\| = 1$. En d'autres mots, L_1^* est fini.

   (b) Le langage $L_2 = \{\epsilon\}$.
   En fait $L_2^* = \bigcup_{n \geq 0} L_2^n = \{\epsilon\}$. Pour démontrer cela, nous montrons d'abord, par induction sur n, que pour tout $n \in \mathbb{N}$, l'ensemble $L_2^n = \{\epsilon\}$.

   $n = 0$.

   Par définition nous avons $L_2^0 = \{\epsilon\}$

   $n \Rightarrow n + 1$.

   Supposons que $L_2^n = \{\epsilon\}$. Nous avons $L_2^{n+1} = L_2^n L_2$, ce qui est égal par définition de la concatenation de deux langages à l'ensemble $\{xy \mid x \in L_2^n \wedge y \in L_2\}$. Or par hypothèse $L_2^n = \{\epsilon\}$ et par définition $L_2 = \{\epsilon\}$. Dès lors $L_2^{n+1} = L_2^n L_2 = \{\epsilon \cdot \epsilon\} = \{\epsilon\}$.

   Nous avons donc $L_2^* = \bigcup_{n \geq 0} L_2^n = \bigcup_{n \geq 0} \{\epsilon\} = \{\epsilon\}$ et $\|\{\epsilon\}\| = 1$. En d'autres mots L_2^* est fini.

2. Pour tout Σ, si $L \subseteq \Sigma^*$ est différent de L_1 et L_2, alors L^* est infini. Nous considérons les deux cas suivants.

   (a) $\Sigma = \emptyset$. Par définition nous avons $\Sigma^* = \{\epsilon\}$. Dès lors les langages possibles $L \subseteq \Sigma^*$ sur Σ sont exactement L_1 et L_2 et la proposition est trivialement satisfaite.

   (b) $\Sigma \neq \emptyset$. Prenons un langage quelconque $L \subseteq \Sigma^*$ tel que $L \neq L_i$ pour i=1,2. De ce fait il existe au moins un mot $w_c \in L$ tel que $w_c \neq \epsilon$. Montrons que L^* est infini.

   Pour cela construisons les ensembles de mots W_m avec $m \in \mathbb{N}$, définis par

   $$W_m = \{w_c^k \mid 1 \leq k \leq m\}$$

   Par construction nous avons pour tout $m \in \mathbb{N}$ les propriétés suivantes, $\|W_m\| = m$ — W_m est donc fini — et $W_m \subseteq L^*$ (car chaque $w_c^k \in L^k \subseteq L^*$).

   Nous procédons par l'absurde. Supposons que la fermeture de L est finie, cela signifie $\exists n \in \mathbb{N} : \|L^*\| = n$. Soit n' ce nombre, nous avons

   $$\|L^*\| = n' < n' + 1 = \|W_{n'+1}\|$$

   Cependant étant donné que $W_{n'+1} \subseteq L^*$, nous avons par le lemme 3.3, $\|L^*\| \geq \|W_{n'+1}\|$, ce qui est contredit l'équation ci-dessus. L'hypothèse selon selon laquelle L^* est fini est donc fausse, en d'autre mots L^* est infini.

   De (a) et (b) nous déduisons (2).

De (1) et (2) nous déduisons la proposition 3.4. ■

4. Traitement de données textuelles avec les expressions régulières

Quelques exemples d'expressions pour les motifs à filtrer,

1. `"X-Mailer:"`
2. `"(Subject|From):"`
3. `"[a-zA-Z0-9]*@[a-zA-Z0-9]*\.com"`
4. Une première approximation est donnée par

   `"([0-9]{1,3}\.){3,3}[0-9]{1,3}"`

   Cependant, cette expression regulière reconnaît un sur-ensemble du langage des adresses IP. En effet, les adresses IP valides notées textuellement vont de `0.0.0.0` à `255.255.255.255` tout en incluant — généralement — les chaînes telles que `000.00.0.0`. Ainsi, par exemple, l'adresse IP invalide `1.999.0.258` sera reconnue par l'expression ci-dessus.

   Pour filtrer un nombre de `0` à `255`, tout en acceptant les chaînes `01`, `041`, ... il faut utiliser l'expression régulière suivante,

   $num =$ `([01]?[0-9]?[0-9])|(2[0-4]?[0-9])|(25[0-5])`

   Les adresses IP valides sont donc filtrées en remplaçant num par sa définition dans l'expression suivante :

   "(*num*`\.){3,3}`*num*"

5. Une première approximation est donnée par

   `"([0-9][0-9]:){2,2}[0-9][0-9]"`

   Cependant, cette expression régulière reconnaît un sur-ensemble des heures au format `HH:MM:SS`. En effet, les heures vont de `00` à `23`, les minutes et les secondes de `00` à `59`. En posant,

   $$\begin{aligned} hh &= \texttt{([0-1][0-9])|(2[0-3])} \\ mm &= \texttt{[0-5][0-9]} \end{aligned}$$

   Les heures valides sont filtrées en remplaçant hh et mm par leur définition dans l'expression suivante :

   "hh`:`mm`:`mm"