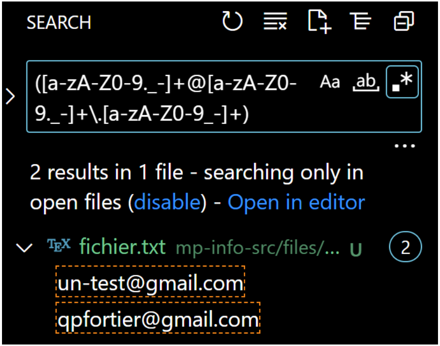
Langages réguliers

Quentin Fortier

December 8, 2023

Motivation

- Recherche de motif dans un texte.



Recherche d'email dans Visual Code

- Formalisation de la syntaxe d'un langage de programmation (et conception de nouveau langage) : BNF d'OCaml, de Python.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1...m_n$.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1 ... m_n$.
  n est la **longueur** de m, qu'on note $|m|$.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1 ... m_n$.
  n est la **longueur** de m, qu'on note $|m|$.
- Le **mot vide** (contenant aucune lettre) est noté ε.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1 ... m_n$.
  n est la **longueur** de m, qu'on note $|m|$.
- Le **mot vide** (contenant aucune lettre) est noté ε.
- On note Σ^* l'ensemble des mots de Σ et $\Sigma^+ = \Sigma^* \backslash \{\varepsilon\}$.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1...m_n$.
  n est la **longueur** de m, qu'on note $|m|$.
- Le **mot vide** (contenant aucune lettre) est noté ε.
- On note Σ^* l'ensemble des mots de Σ et $\Sigma^+ = \Sigma^* \backslash \{\varepsilon\}$.
- On note Σ^n l'ensemble des mots de Σ de taille n.

Mots

Définitions

- Un **alphabet** est un ensemble Σ **fini**, dont les éléments sont des **lettres**.
- Un **mot** m d'un alphabet Σ est une suite finie m_1, ..., m_n de lettres de Σ, et on note $m = m_1 ... m_n$.
  n est la **longueur** de m, qu'on note $|m|$.
- Le **mot vide** (contenant aucune lettre) est noté ε.
- On note Σ^* l'ensemble des mots de Σ et $\Sigma^+ = \Sigma^* \backslash \{\varepsilon\}$.
- On note Σ^n l'ensemble des mots de Σ de taille n.

Remarque : si Σ est un alphabet et $a \in \Sigma$, on confond parfois la lettre a et le mot a... C'est le contexte qui permet de faire la différence.

Mots

Définition : Égalité de mots

Deux mots $u = u_1 ... u_n$ et $v = v_1 ... v_p$ sur le même alphabet Σ sont **égaux** s'ils ont la même longueur ($n = p$) et si pour tout $i \in \{1, ..., n\}$, $u_i = v_i$.

Mots

Définition : Égalité de mots

Deux mots $u = u_1 ... u_n$ et $v = v_1 ... v_p$ sur le même alphabet Σ sont **égaux** s'ils ont la même longueur $(n = p)$ et si pour tout $i \in \{1, ..., n\}$, $u_i = v_i$.

Définition : Concaténation et puissance

- La **concaténation** de deux mots $u = u_1 ... u_n$ et $v = v_1 ... v_p$ est :
$$uv = u_1 ... u_n v_1 ... v_p$$
Elle est aussi parfois notée $u \cdot v$.

Mots

Définition : Égalité de mots

Deux mots $u = u_1...u_n$ et $v = v_1...v_p$ sur le même alphabet Σ sont **égaux** s'ils ont la même longueur ($n = p$) et si pour tout $i \in \{1, ..., n\}$, $u_i = v_i$.

Définition : Concaténation et puissance

- La **concaténation** de deux mots $u = u_1...u_n$ et $v = v_1...v_p$ est :
$$uv = u_1...u_n v_1...v_p$$
Elle est aussi parfois notée $u \cdot v$.
- Si u est un mot, on définit $u^0 = \varepsilon$ et $u^k = \underbrace{uu...u}_{k}$

Mots

Définition : Égalité de mots

Deux mots $u = u_1 \dots u_n$ et $v = v_1 \dots v_p$ sur le même alphabet Σ sont **égaux** s'ils ont la même longueur ($n = p$) et si pour tout $i \in \{1, \dots, n\}$, $u_i = v_i$.

Définition : Concaténation et puissance

- La **concaténation** de deux mots $u = u_1 \dots u_n$ et $v = v_1 \dots v_p$ est :
  $$uv = u_1 \dots u_n v_1 \dots v_p$$
  Elle est aussi parfois notée $u \cdot v$.
- Si u est un mot, on définit $u^0 = \varepsilon$ et $u^k = \underbrace{uu \dots u}_{k}$

Question

Soient Σ un alphabet, a, $b \in \Sigma$ et $u \in \Sigma$. On suppose $au = ub$. Montrer que $a = b$ et qu'il existe $k \in \mathbb{N}$ tel que $u = a^k$.

Mots

Un monoïde (HP) est comme un groupe, sauf qu'il n'y a pas forcément d'inverse.

Lemme

$(\Sigma^*, \cdot)$ est un monoïde, où $\cdot$ est la concaténation de mots, c'est-à-dire :

- $\varepsilon \cdot m = m \cdot \varepsilon = m$ (ε est élément neutre)
- $(m \cdot n) \cdot p = m \cdot (n \cdot p)$ (associativité)

Mots

Un monoïde (HP) est comme un groupe, sauf qu'il n'y a pas forcément d'inverse.

Lemme

$(\Sigma^*, \cdot)$ est un monoïde, où $\cdot$ est la concaténation de mots, c'est-à-dire :

- $\varepsilon \cdot m = m \cdot \varepsilon = m$ (ε est élément neutre)
- $(m \cdot n) \cdot p = m \cdot (n \cdot p)$ (associativité)

Lemme

La fonction qui à un mot associe sa longueur est un morphisme de monoïde de $(\Sigma^*, \cdot)$ vers $(\mathbb{N}, +)$, c'est-à-dire :

- $|\varepsilon| = 0$
- $|m \cdot n| = |m| + |n|$

Mots

Soit u et m deux mots de Σ^*.

Définitions

- u est un **préfixe** de m s'il existe un mot v tel que $m = uv$.

Mots

Soit u et m deux mots de Σ^*.

Définitions

- u est un **préfixe** de m s'il existe un mot v tel que $m = uv$.
- u est un **suffixe** de m s'il existe un mot v tel que $m = vu$.

Mots

Soit u et m deux mots de Σ^*.

Définitions

- u est un **préfixe** de m s'il existe un mot v tel que $m = uv$.
- u est un **suffixe** de m s'il existe un mot v tel que $m = vu$.
- u est un **facteur** (*substring* en anglais) de m s'il existe des mots v, w tels que $m = vuw$.

Mots

Soit u et m deux mots de Σ^*.

Définitions

- u est un **préfixe** de m s'il existe un mot v tel que $m = uv$.
- u est un **suffixe** de m s'il existe un mot v tel que $m = vu$.
- u est un **facteur** (*substring* en anglais) de m s'il existe des mots v, w tels que $m = vuw$.
- u est un **sous-mot** (*subsequence* en anglais) de m si u est une sous-suite (ou : suite extraite) de m.

Exemple : abc est un sous-mot de $aabacb$, mais pas un facteur.

Mots

Soit u et m deux mots de Σ^*.

Définitions

- u est un **préfixe** de m s'il existe un mot v tel que $m = uv$.
- u est un **suffixe** de m s'il existe un mot v tel que $m = vu$.
- u est un **facteur** (*substring* en anglais) de m s'il existe des mots v, w tels que $m = vuw$.
- u est un **sous-mot** (*subsequence* en anglais) de m si u est une sous-suite (ou : suite extraite) de m.

<u>Exemple</u> : abc est un sous-mot de $aabacb$, mais pas un facteur.

Exercice

Soit m un mot de taille n dont toutes les lettres sont différentes. Quel est son nombre de préfixes, de suffixes, de facteurs et de sous-mots ? Et si des lettres peuvent être répétées ?

Mots

Rappels d'utilisation d'une chaîne de caractères (**string**) :

```
let s = "abc" (* défini une chaîne de caractères *)
s.[1] (* donne 'b' *)
String.length s (* donne 3 *)
"abc" ^ "def" (* concaténation *)
```

Remarque : Contrairement à **array**, le type **string** est immuable (on ne peut pas modifier un caractère d'une chaîne de caractères).

Mots

Rappels d'utilisation d'une chaîne de caractères (`string`) :

```
let s = "abc" (* défini une chaîne de caractères *)
s.[1] (* donne 'b' *)
String.length s (* donne 3 *)
"abc" ^ "def" (* concaténation *)
```

Remarque : Contrairement à `array`, le type `string` est immuable (on ne peut pas modifier un caractère d'une chaîne de caractères).

Question

Écrire une fonction `sous_mot : string -> string -> bool` qui teste si un mot est un sous-mot d'un autre, en complexité linéaire.

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ, ce qui est équivalent à $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ, ce qui est équivalent à $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Exemples :

1. L'ensemble L_1 des mots du dictionnaire français sur $\Sigma = \{a, b, ..., z\}$.

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ, ce qui est équivalent à $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Exemples :

1. L'ensemble L_1 des mots du dictionnaire français sur $\Sigma = \{a, b, ..., z\}$.
2. L'ensemble L_2 des formules arithmétiques sur $\Sigma = \{0, ..., 9, +, -, /, *\}$.

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ, ce qui est équivalent à $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Exemples :

1. L'ensemble L_1 des mots du dictionnaire français sur $\Sigma = \{a, b, ..., z\}$.
2. L'ensemble L_2 des formules arithmétiques sur $\Sigma = \{0, ..., 9, +, -, /, *\}$.
3. L'ensemble L_3 des programmes OCaml sur $\Sigma = \{a, ..., z, , !, <, >, ...\}$.

Langages

Définition

Un **langage** L sur un alphabet Σ est un ensemble de mots de Σ, ce qui est équivalent à $L \subseteq \Sigma^*$ ou encore $L \in \mathcal{P}(\Sigma^*)$.

Exemples :

1. L'ensemble L_1 des mots du dictionnaire français sur $\Sigma = \{a, b, ..., z\}$.
2. L'ensemble L_2 des formules arithmétiques sur $\Sigma = \{0, ..., 9, +, -, /, *\}$.
3. L'ensemble L_3 des programmes OCaml sur $\Sigma = \{a, ..., z, , !, <, >, ...\}$.
4. L'ensemble L_4 des ADN sur $\Sigma = \{A, C, G, T\}$.

Langages

Exercice : Résumé des définitions

Soit $\Sigma = \{a, b\}$.

1. Σ est
2. a est
3. ε est
4. $abaaabb$ est
5. $\{a, b, abaaabb\}$ est

Exercice : Résumé des définitions

Soit $\Sigma = \{a, b\}$.

1. Σ est un alphabet.
2. a est une lettre (et aussi un mot de longueur 1...).
3. ε est le mot vide.
4. $abaaabb$ est un mot.
5. $\{a, b, abaaabb\}$ est un langage.

Langages

On veut des algorithmes pour résoudre les problèmes suivants :

Problème

Étant donné un langage L et un mot m, est-ce que $m \in L$?

Applications : correcteur orthographique, coloration syntaxique...

Langages

On veut des algorithmes pour résoudre les problèmes suivants :

Problème

Étant donné un langage L et un mot m, est-ce que $m \in L$?

Applications : correcteur orthographique, coloration syntaxique...

Problème

Étant donné un texte s et un langage L, s contient-il un mot de L?

Langages

On veut des algorithmes pour résoudre les problèmes suivants :

Problème

Étant donné un langage L et un mot m, est-ce que $m \in L$?

Applications : correcteur orthographique, coloration syntaxique...

Problème

Étant donné un texte s et un langage L, s contient-il un mot de L?
(Cas particulier $L = \{m\}$: le mot m apparaît-il dans un texte s?)

Application : recherche de motif (adresse mail, séquence d'ADN...) dans un texte.

Opérations sur les langages

Soient L_1 et L_2 deux langages de même alphabet.

Définition : Concaténation

La concaténation L_1L_2 de L_1 et L_2 est défini par :

$$L_1L_2 = \{m_1m_2 \mid m_1 \in L_1,\ m_2 \in L_2\}$$

Opérations sur les langages

Soient L_1 et L_2 deux langages de même alphabet.

Définition : Concaténation

La concaténation L_1L_2 de L_1 et L_2 est défini par :

$$L_1L_2 = \{m_1m_2 \mid m_1 \in L_1,\ m_2 \in L_2\}$$

L_1L_2 est donc l'ensemble des mots obtenus par concaténation d'un mot de L_1 et d'un mot de L_2.

Opérations sur les langages

Soient L_1 et L_2 deux langages de même alphabet.

Définition : Concaténation

La concaténation $L_1 L_2$ de L_1 et L_2 est défini par :

$$L_1 L_2 = \{m_1 m_2 \mid m_1 \in L_1,\ m_2 \in L_2\}$$

$L_1 L_2$ est donc l'ensemble des mots obtenus par concaténation d'un mot de L_1 et d'un mot de L_2.

Exercice

1. Soit $L_1 = \{a, ab\}$ et $L_2 = \{\varepsilon, b, bba\}$. Déterminer $L_1 L_2$.
2. Quel lien a t-on entre $|L_1 L_2|$ et $|L_1||L_2|$, dans le cas général ?

Opérations sur les langages

Soient L un langage et $n \in \mathbb{N}$.

Définition : Puissance

On définit la puissance L^n de L par récurrence :

$$L^0 = \{\varepsilon\}$$
$$L^n = L^{n-1}L, \text{ pour } n \geq 1$$

Opérations sur les langages

Soient L un langage et $n \in \mathbb{N}$.

Définition : Puissance

On définit la puissance L^n de L par récurrence :

$$L^0 = \{\varepsilon\}$$
$$L^n = L^{n-1}L, \text{ pour } n \geq 1$$

Dit autrement : $L^n = \underbrace{L \cdot \ldots \cdot L}_{n} = \{m_1 \cdots m_n \mid m_1 \in L, \ldots, m_n \in L\}$.

Opérations sur les langages

Soient L un langage et $n \in \mathbb{N}$.

Définition : Puissance

On définit la puissance L^n de L par récurrence :

$$L^0 = \{\varepsilon\}$$
$$L^n = L^{n-1}L, \text{ pour } n \geq 1$$

Dit autrement : $L^n = \underbrace{L \cdot ... \cdot L}_{n} = \{m_1 \cdots m_n \mid m_1 \in L, \ldots, m_n \in L\}$.

Exemple : Σ^n est l'ensemble des mots de longueur n sur l'alphabet Σ.

Opérations sur les langages

Soient L un langage et $n \in \mathbb{N}$.

Définition : Puissance

On définit la puissance L^n de L par récurrence :

$$L^0 = \{\varepsilon\}$$
$$L^n = L^{n-1}L, \text{ pour } n \geq 1$$

Dit autrement : $L^n = \underbrace{L \cdot ... \cdot L}_{n} = \{m_1 \cdots m_n \mid m_1 \in L, \dots, m_n \in L\}$.

Exemple : Σ^n est l'ensemble des mots de longueur n sur l'alphabet Σ.

Exercice

1. À quelle condition a t-on $L \subseteq L^2$?
2. Quel lien a t-on entre L^2 et $\{u^2 \mid u \in L\}$?

Opérations sur les langages

Soit L un langage.

Définition : Étoile de Kleene

On définit l'**étoile de Kleene** L^* de L par :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

Opérations sur les langages

Soit L un langage.

Définition : Étoile de Kleene

On définit l'**étoile de Kleene** L^* de L par :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

L^* est donc l'ensemble des mots obtenus par concaténation d'un nombre quelconque de mots de L.

Opérations sur les langages

Soit L un langage.

Définition : Étoile de Kleene

On définit l'**étoile de Kleene** L^* de L par :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

L^* est donc l'ensemble des mots obtenus par concaténation d'un nombre quelconque de mots de L.
Remarque : L^* contient toujours ε, car $L^0 = \{\varepsilon\}$.

Opérations sur les langages

Soit L un langage.

Définition : Étoile de Kleene

On définit l'**étoile de Kleene** L^* de L par :

$$L^* = \bigcup_{k \in \mathbb{N}} L^k$$

L^* est donc l'ensemble des mots obtenus par concaténation d'un nombre quelconque de mots de L.
Remarque : L^* contient toujours ε, car $L^0 = \{\varepsilon\}$.

Question

Montrer que $(L^*)^* = L^*$.

Langages réguliers

Soit Σ un alphabet.

Définition

L'ensemble des **langages réguliers** de Σ est le plus petit ensemble de langages de Σ **contenant les langages finis** de Σ et stable par **concaténation**, **union**, **étoile de Kleene**.

Langages réguliers

Soit Σ un alphabet.

Définition

L'ensemble des **langages réguliers** de Σ est le plus petit ensemble de langages de Σ **contenant les langages finis** de Σ et stable par **concaténation**, **union**, **étoile de Kleene**.

Dit autrement :

Définition

- Tout langage fini est régulier
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier
- L régulier $\implies$ L^* régulier

Langages réguliers

Soit Σ un alphabet.

Définition

L'ensemble des **langages réguliers** de Σ est le plus petit ensemble de langages de Σ **contenant les langages finis** de Σ et stable par **concaténation**, **union**, **étoile de Kleene**.

Dit autrement :

Définition

- Tout langage fini est régulier
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier
- L régulier $\implies$ L^* régulier

Attention : une union infinie de langages réguliers n'est pas forcément régulière.

Langages réguliers

Définition

- Tout langage fini est régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier.
- L régulier $\implies$ L^* régulier.

Exemples :

1. Soit m un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi m par abus de langage.

Langages réguliers

Définition

- Tout langage fini est régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier.
- L régulier $\implies$ L^* régulier.

Exemples :

1. Soit m un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi m par abus de langage.
2. Σ est fini donc est régulier. Σ^* est l'étoile d'un langage régulier donc est régulier.

Langages réguliers

Définition

- Tout langage fini est régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier.
- L régulier $\implies$ L^* régulier.

Exemples :

1. Soit m un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi m par abus de langage.
2. Σ est fini donc est régulier. Σ^* est l'étoile d'un langage régulier donc est régulier.
3. Soit m un mot. L'ensemble des mots ayant comme facteur m

Langages réguliers

Définition

- Tout langage fini est régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier.
- L régulier $\implies$ L^* régulier.

Exemples :

1. Soit m un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi m par abus de langage.
2. Σ est fini donc est régulier. Σ^* est l'étoile d'un langage régulier donc est régulier.
3. Soit m un mot. L'ensemble des mots ayant comme facteur m est égal à $\Sigma^* m \Sigma^*$ donc est un langage régulier.
4. Soit $m = m_1 \cdots m_n$ un mot. L'ensemble des mots ayant comme sous-mot m

Langages réguliers

Définition

- Tout langage fini est régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 \cup L_2$ régulier.
- L_1 et L_2 réguliers $\implies$ $L_1 L_2$ régulier.
- L régulier $\implies$ L^* régulier.

Exemples :

1. Soit m un mot. Alors $\{m\}$ est fini donc est un langage régulier, qu'on note aussi m par abus de langage.
2. Σ est fini donc est régulier. Σ^* est l'étoile d'un langage régulier donc est régulier.
3. Soit m un mot. L'ensemble des mots ayant comme facteur m est égal à $\Sigma^* m \Sigma^*$ donc est un langage régulier.
4. Soit $m = m_1 \cdots m_n$ un mot. L'ensemble des mots ayant comme sous-mot m est égal à $\Sigma^* m_1 \Sigma^* m_2 \cdots \Sigma^* m_n \Sigma^*$ donc est un langage régulier.

Langages réguliers

Exercice

Montrer que les langages suivants sont réguliers sur $\Sigma = \{a, b\}$:

1. Mots commençants par a.
2. Mots commençants par a et finissant par b
3. Mots de taille paire.
4. Mots de taille impaire.

Expressions régulières

Les expressions régulières sont une notation plus concise pour représenter un langage régulier :

Expressions régulières

L'ensemble des **expressions régulières** sur un alphabet Σ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (,)\}$ vérifiant :

- $\forall a \in \Sigma$, $a \in \mathcal{R}$
- $\emptyset \in \mathcal{R}$, $\varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R}$, $(e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R}$, $e^* \in \mathcal{R}$

Expressions régulières

Les expressions régulières sont une notation plus concise pour représenter un langage régulier :

Expressions régulières

L'ensemble des **expressions régulières** sur un alphabet Σ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (,)\}$ vérifiant :

- $\forall a \in \Sigma$, $a \in \mathcal{R}$
- $\emptyset \in \mathcal{R}$, $\varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R}$, $(e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R}$, $e^* \in \mathcal{R}$

Remarques :

- On utilisera seulement les parenthèses nécessaires. Ainsi, $(((ab)c)|d)$ sera noté $abc|d$.
- $e_1|e_2$ est aussi noté $e_1 + e_2$.

Expressions régulières

Les expressions régulières sont une notation plus concise pour représenter un langage régulier :

Expressions régulières

L'ensemble des **expressions régulières** sur un alphabet Σ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (,)\}$ vérifiant :

- $\forall a \in \Sigma$, $a \in \mathcal{R}$
- $\emptyset \in \mathcal{R}$, $\varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R}$, $(e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R}$, $e^* \in \mathcal{R}$

Remarques :

- On utilisera seulement les parenthèses nécessaires. Ainsi, $(((ab)c)|d)$ sera noté $abc|d$.
- $e_1|e_2$ est aussi noté $e_1 + e_2$.

Exemples : a^*, $(a|aba)^*$, $a(a|b)^*b$ sont des expressions régulières.

Expressions régulières

Expressions régulières

L'ensemble des **expressions régulières** sur un alphabet Σ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (,)\}$ vérifiant :

- $\forall a \in \Sigma,\ a \in \mathcal{R}$
- $\emptyset \in \mathcal{R},\ \varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R},\ (e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R},\ e^* \in \mathcal{R}$

```
type 'a regexp =
    | Vide | Epsilon | L of 'a (* L a est la lettre a *)
    | Union of 'a regexp * 'a regexp
    | Concat of 'a regexp * 'a regexp
    | Etoile of 'a regexp
```

Expressions régulières

Expressions régulières

L'ensemble des **expressions régulières** sur un alphabet Σ est le plus petit langage $\mathcal{R}$ sur $\Sigma \cup \{\emptyset, \varepsilon, |, ^*, (,)\}$ vérifiant :

- $\forall a \in \Sigma,\ a \in \mathcal{R}$
- $\emptyset \in \mathcal{R},\ \varepsilon \in \mathcal{R}$
- $\forall e_1, e_2 \in \mathcal{R},\ (e_1|e_2) \in \mathcal{R}$ et $(e_1 e_2) \in \mathcal{R}$
- $\forall e \in \mathcal{R},\ e^* \in \mathcal{R}$

```
type 'a regexp =
    | Vide | Epsilon | L of 'a (* L a est la lettre a *)
    | Union of 'a regexp * 'a regexp
    | Concat of 'a regexp * 'a regexp
    | Etoile of 'a regexp
```

Par exemple, $a(a|b)^*b$ est représenté par :

```
Concat(L 'a', Concat(Etoile(Union(L 'a', L 'b')), L 'b'))
```

Expressions régulières

```
type 'a regexp =
    | Vide | Epsilon | L of 'a
    | Union of 'a regexp * 'a regexp
    | Concat of 'a regexp * 'a regexp
    | Etoile of 'a regexp
```

Question

Écrire une fonction `lettres : 'a regexp -> 'a list` qui renvoie la liste des lettres utilisées dans une expression régulière.

Expressions régulières

Langage d'une expression régulière

Le langage $L(e)$ d'une expression régulière e est définie récursivement :

- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

Par abus de langage, on oublie souvent le $L(e)$ et on confond expression régulière et langage associé.

Expressions régulières

Langage d'une expression régulière

Le langage $L(e)$ d'une expression régulière e est définie récursivement :

- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

Par abus de langage, on oublie souvent le $L(e)$ et on confond expression régulière et langage associé.

Équivalence entre langage régulier et expression régulière

Un langage L est régulier

$$\iff$$

Il existe une expression régulière e telle que $L = L(e)$

Expressions régulières

Langage d'une expression régulière

Le langage $L(e)$ d'une expression régulière e est définie récursivement :

- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

Exemples avec $\Sigma = \{a, b\}$:

- $(a|b)^*$: ensemble de tous les mots $(= \Sigma^*)$.

Expressions régulières

Langage d'une expression régulière

Le langage $L(e)$ d'une expression régulière e est définie récursivement :

- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

<u>Exemples</u> avec $\Sigma = \{a, b\}$:

- $(a|b)^*$: ensemble de tous les mots $(= \Sigma^*)$.
- $(a|b)^* bb$:

Expressions régulières

Langage d'une expression régulière

Le langage $L(e)$ d'une expression régulière e est définie récursivement :

- $L(a) = \{a\}$ si $a \in \Sigma$
- $L(\emptyset) = \emptyset$, $L(\varepsilon) = \{\varepsilon\}$
- $L(e|e') = L(e) \cup L(e')$
- $L(ee') = L(e)L(e')$
- $L(e^*) = L(e)^*$

Exemples avec $\Sigma = \{a, b\}$:

- $(a|b)^*$: ensemble de tous les mots ($= \Sigma^*$).
- $(a|b)^* bb$: mots finissant par bb.

Expressions régulières

Exercice

Donner une expression régulière pour les langages suivants, sur $\Sigma = \{a, b\}$:

1. Mots contenant au plus un a.
2. Mots de taille $n \equiv 1 \mod 3$.
3. Mots contenant un nombre pair de a
4. Mots contenant un nombre impair de a

Expressions régulières

Exercice

Donner une expression régulière pour les langages suivants, sur $\Sigma = \{a, b\}$:

1. Mots contenant au plus un a.
2. Mots de taille $n \equiv 1 \mod 3$.
3. Mots contenant un nombre pair de a : $L((ab^*a|b)^*)$.
4. Mots contenant un nombre impair de a : $L(b^*a(ab^*a|b)^*)$

Question

Donner une expression régulière pour les écritures en base 2 d'entiers divisibles par 4.

Induction structurelle

Méthode

On peut démontrer une propriété $\mathcal{P}(L)$ sur les langages réguliers L par **induction structurelle** ($\approx$ récurrence), en montrant :

- $\mathcal{P}(L)$ est vraie pour les langages L finis (cas de base)
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 L_2)$
- $\mathcal{P}(L_1) \wedge \mathcal{P}(L_2) \implies \mathcal{P}(L_1 \cup L_2)$
- $\mathcal{P}(L) \implies \mathcal{P}(L^*)$

Induction structurelle

Méthode

On peut démontrer une propriété $\mathcal{P}(L)$ sur les langages réguliers L par **induction structurelle** ($\approx$ récurrence), en montrant :

- $\mathcal{P}(L)$ est vraie pour les langages L finis (cas de base)
- $\mathcal{P}(L_1) \land \mathcal{P}(L_2) \implies \mathcal{P}(L_1 L_2)$
- $\mathcal{P}(L_1) \land \mathcal{P}(L_2) \implies \mathcal{P}(L_1 \cup L_2)$
- $\mathcal{P}(L) \implies \mathcal{P}(L^*)$

$\{L \mid \mathcal{P}(L)\}$ est alors un ensemble contenant les langages finis et stable par union, concaténation et étoile de Kleene.
Donc il contient tous les langages réguliers, par définition.

Induction structurelle

Méthode

De même, on peut démontrer une propriété $\mathcal{P}(e)$ sur les expressions régulières e en montrant :

- $\mathcal{P}(\emptyset)$, $\mathcal{P}(\varepsilon)$ sont vraies (cas de base)
- $\mathcal{P}(a)$ est vraie pour $a \in \Sigma$ (cas de base)
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 e_2)$
- $\mathcal{P}(e_1) \wedge \mathcal{P}(e_2) \implies \mathcal{P}(e_1 \cup e_2)$
- $\mathcal{P}(e) \implies \mathcal{P}(e^*)$

Exercice : Miroir

Si $m = m_1 ... m_n$ est un mot, on définit son miroir $\widetilde{m} = m_n ... m_1$.
Si L est un langage, on définit son miroir $\widetilde{L} = \{\widetilde{m} \mid m \in L\}$.

1. Donner une expression régulière du miroir de $a(a|b)^* b$.
2. Soit e une expression régulière de langage L. Montrer que $\widetilde{L}$ est régulier.
3. Écrire une fonction Caml `miroir : 'a regexp -> 'a regexp` renvoyant le miroir d'une expression régulière.

Induction structurelle

En notant $e_1 \equiv e_2 \iff L(e_1) = L(e_2)$:

Propriétés sur les expressions régulières

- $\emptyset e \equiv e\emptyset \equiv \emptyset$
- $\varepsilon e \equiv e\varepsilon \equiv e$
- $(e_1|e_2)e_3 \equiv e_1 e_3|e_2 e_3$ (distributivité)
- $e_1(e_2 e_3) \equiv (e_1 e_2)e_3$ (associativité)

Induction structurelle

Méthode

- Pour montrer une égalité de deux mots, on peut faire une récurrence sur la longueur du mot.
- Pour montrer une égalité de deux langages, on peut montrer une double inclusion.

Question

Soient e_1 et e_2 deux expressions régulières.
Montrer que $(e_1^* e_2)^* e_1^* \equiv (e_1|e_2)^*$.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

Σ^+ (et donc aussi Σ^*) est infini dénombrable.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

Σ^+ (et donc aussi Σ^*) est infini dénombrable.

Preuve : on identifie les p lettres de Σ avec les entiers de 0 à $p-1$.
L'écriture en base p donne une bijection de $\mathbb{N}$ vers Σ^+.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

Σ^+ (et donc aussi Σ^*) est infini dénombrable.

Preuve : on identifie les p lettres de Σ avec les entiers de 0 à $p-1$. L'écriture en base p donne une bijection de $\mathbb{N}$ vers Σ^+.

On en déduit :

Théorème

Un langage $L \subseteq \Sigma^*$ est au plus dénombrable.

Par exemple, le langage des programmes Caml est dénombrable.

Complément : Dénombrabilité et langages non réguliers

Théorème de Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Complément : Dénombrabilité et langages non réguliers

Théorème de Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Preuve : si $f : E \longrightarrow \mathcal{P}(E)$ alors $Y = \{x \in E \mid x \notin f(x)\}$ n'a pas d'antécédent par f.

Complément : Dénombrabilité et langages non réguliers

Théorème de Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Preuve : si $f : E \longrightarrow \mathcal{P}(E)$ alors $Y = \{x \in E \mid x \notin f(x)\}$ n'a pas d'antécédent par f.

Corollaire

L'ensemble $\mathcal{P}(\Sigma^*)$ des langages sur Σ n'est pas dénombrable.

Complément : Dénombrabilité et langages non réguliers

Théorème de Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Preuve : si $f : E \longrightarrow \mathcal{P}(E)$ alors $Y = \{x \in E \mid x \notin f(x)\}$ n'a pas d'antécédent par f.

Corollaire

L'ensemble $\mathcal{P}(\Sigma^*)$ des langages sur Σ n'est pas dénombrable.

Comme l'ensemble des langages est indénombrable alors que l'ensemble des programmes Caml est dénombrable, il existe des langages L pour lesquels le problème suivant ne peut pas être résolu par un algorithme :

Problème

Étant donné un mot m, est-ce que $m \in L$?

Complément : Dénombrabilité et langages non réguliers

Théorème de Cantor

Un ensemble E ne peut pas être en bijection avec $\mathcal{P}(E)$.

Preuve : si $f : E \longrightarrow \mathcal{P}(E)$ alors $Y = \{x \in E \mid x \notin f(x)\}$ n'a pas d'antécédent par f.

Corollaire

L'ensemble $\mathcal{P}(\Sigma^*)$ des langages sur Σ n'est pas dénombrable.

Comme l'ensemble des langages est indénombrable alors que l'ensemble des programmes Caml est dénombrable, il existe des langages L pour lesquels le problème suivant ne peut pas être résolu par un algorithme :

Problème

Étant donné un mot m, est-ce que $m \in L$?

On va donc se restreindre à un ensemble plus simple de langages.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

L'ensemble des langages réguliers sur Σ est infini dénombrable.

<u>Preuve</u> :

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

L'ensemble des langages réguliers sur Σ est infini dénombrable.

Preuve : l'ensemble des expressions régulières sur Σ est dénombrable (c'est un langage) donc l'ensemble des langages réguliers aussi.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

L'ensemble des langages réguliers sur Σ est infini dénombrable.

Preuve : l'ensemble des expressions régulières sur Σ est dénombrable (c'est un langage) donc l'ensemble des langages réguliers aussi.

Comme l'ensemble de tous les langages sur Σ est non dénombrable :

Corollaire

Il existe des langages non réguliers sur Σ.

Complément : Dénombrabilité et langages non réguliers

Soit Σ un alphabet non vide.

Théorème

L'ensemble des langages réguliers sur Σ est infini dénombrable.

Preuve : l'ensemble des expressions régulières sur Σ est dénombrable (c'est un langage) donc l'ensemble des langages réguliers aussi.

Comme l'ensemble de tous les langages sur Σ est non dénombrable :

Corollaire

Il existe des langages non réguliers sur Σ.

On verra plus tard comment montrer qu'un langage **n'est pas** régulier...