Université de M'sila
Faculté des Mathématiques et de l'Informatique
Departement des Mathématiques

Mémoire

Présenté pour l'obtention du diplôme de **Master**

Domaine: Mathématiques et Informatiques

Filière: Mathématiques

Option: Mathématiques Appliquées et discrètes

Par

FADILA BOUDRAF

THÈME

Etude sur les codes de longueurs variables

Soutenu le : 30/05/ 2016

Devant le jury composé de :

1)Mr.N.MIDOUN MCB.Univ de M'sila Président
2Mr.D.MIHOUBI. Prof.Univ de M'sila Rapporteur
3)N.GADBANE.MCB.Univ de M'sila Examinateur

Dirigé par:

Mr.Mihoubi Douadi

Année: **2015/2016**

Remerciements

Je tiens à remercier, en premier lieu, **Mon Dieu** qui m'a donné la force de rédiger ce modeste travail.

Je tiens à remercier les membres du jury, qui ont accepté d'évaluer mon travail de mémoire.

Je tiens à remercie Mr. **Mihoubi Douadi**, directeur de mon mémoire, pour sa disponibilité et ses conseils judicieux tout au long de ce travail.

J'exprime ici ma profonde gratitude à Mr. **Nacer Ghadbane**, professeur à l'université de Msila pour m'avoir fait l'honneur de présider mon jury.

Je tiens à exprimer tout mes respects à mes parents, qui m'ont toujours encouragé.

Je remercie tous les professeurs du département de Mathématiques, sans oublier aussi mes collègues et amies, ainsi tous ceux qui ont participé de loin ou de près à l'élaboration de ce mémoire.

Dédicaces

Je dédie ce modeste travail :

-A mes parents ma mère et mon pére.

- A mes soeurs

-A mes frères.

-A toute la famille.

-A toute mes amies.

- Je tiens à remercier l'ensemble de tous les étudiants et étudiantes de ma promotion,

En fin je dédie ce mémoire à mes collègues et tous ceux qui me sont chers.

NOTATIONS

S	Semi-groupe
M	Monoïde
S/R	Semi-groupe quotient
T_a	Sous monoïde engendré par a
A	Alphabet
A^*	Monoïde libre
ε	Mot vide
M/R	Monoïde quotient
$\mathcal{L}$	Langage
T	Automate
$T_{\mathcal{L}}$	Automate minimal
$[w]$	Classe d'équivalence
$\equiv_{\mathcal{L}}$	Congruence syntaxique
$A^* / \equiv_{\mathcal{L}}$	Monoïde syntaxique
C	Code

Table des matières

Introduction

Les systèmes de traitement de l'information ont toujours utilisé des techniques de codage pour différents buts : protection contre les erreurs, représentation de l'information en mémoire d'un ordinateur, compression de l'information, cryptage, etc.

Les codes de longueurs variables constituent une classe d'objets très importante, comme témoignent les difiérents domaines dans lesquels ils furent introduits :

- en théorie de l'information par SHANNON dans les années 1950.
- dans la théorie des évènements récurrents par FELLER (1950).
- dans la théorie des langages formels par SCHUTZENBERGER (1956).

Dans ce travail, on s'intéressé à l'étude de quelques définitions et propriétés et concernat les codes de longueurs variables. Dans ce qui suit, A dénote un ensemble (alphabet) d'éléments appelés lettres. A^* est le monoïde libre engendré par A c'est- à-dire l'ensemble de tous les mots en les lettres de A muni de l'opération de concaténation. ε désigne l'élémen neutre de A (c'est- à-dire le mot vide).

Soit C une partie de A^* , on dit que C est un code sur A si :

tout mot w de $C^+ = C^* \setminus \{\varepsilon\}$ s'écrit d'une maniére unique en éléments de C c'est-à-dire :

pour tout $x_1, ..., x_n; y_1, ..., y_m \in C$,

$$x_1.....x_n = y_1.....y_m \Rightarrow n = m,\ x_i = y_i,\ i = 1.....n.$$

Ce travail est composé de deux chapitres.

Le premier chapitre consiste en un rappel des notions et notations utilisées par la suite :

Semi-groupes, monoïde, mot et langage, automates finis déterministes, monoïde syntaxique.

Dans le deuxiem chapitre, on fait une étude sur les codes de longueurs variables ainsi que certaines de leurs propriétés caractéristiques et on introduit aussi au sein de ce chapitre l'algorithme de reconnaissance des codes, le codage de Huffman, mesure d'un code, codes complets .

Chapitre 1

Préliminaire

Ce premier chapitre contient les définitions et les propriétés des objets mathématiques que nous utiliserons par la suite: Semi-groupes, monoïde, mot et langage, automates finis déterministes, monoïde syntaxique.

1.1 Semi-groupes

Définition 1.1.1 ***(Semi-groupes):***

Un ***semi-groupe*** *est un ensemble non vide S muni d'une opération binaire $(*)$ associative . Nous désignerons un semi -groupe par $(S, *)$.*

Si l'opération $(*)$ d'un semi-groupe $(S, *)$ est commutative, on dit que le semi-groupe est commutatif.

Exemple 1.1.1

1. $(\mathbb{N}, +)$, $(\mathbb{Z}, +)$, $(\mathbb{R}, +)$, *sont des semi-groupes commutatifs.*

2. $(\mathbb{Z}, -)$ *n'est pas semi-groupe car $(-)$ n'est pas associative.*

Définition 1.1.2

Soit $(S, *)$ un semi-groupe et soit $e \in S$. on dit que e est un élément neutre ou élément unité de $(S, *)$ si:

$$\forall\ x \in S \qquad e * x = x * e = x.$$

Définition 1.1.3

Soit $(S, *)$un semi-groupe et soit R une relation d'équivalence sur S. On dit que R est compatible avec l'opération $(*)$, ou encore que R est une congruence, si:

$$\forall\ x, y, z, t \ \in \ S:\ xRy \text{ et } zRt \Rightarrow xzRyt \ \text{ ie:}$$
$$\overline{x} \ = \ \overline{z} \text{ et } \overline{y} = \overline{t} \text{ on a donc } \overline{x * y} = \overline{z * t}.$$

Si R est une congruence, on peut définir sur l'ensemble quotient S/R une opération $(*)$ par:

$$\overline{x} * \overline{y} = \overline{x * y}.$$

L'opération $(*)$ ainsi définie sur S/R est encore associative. Ainsi $(S/R, *)$ est un semi-groupe, on l'appelle ***semi-groupe quotient*** *de $(S, *)$par la congruence R.*

Exemple 1.1.2

On défini sur $\mathbb{Z}$ la relation R par

$$\forall\ x,\ y \in \mathbb{Z},\ \exists\ k \ \text{ tel que } xRy \Leftrightarrow (x - y) = nk \ \ \backslash (n \in \mathbb{N}^*).$$

La relation R est compatible avec l'addition. En effet, si xRy et zRt il existe des entiers k et m tel que: $x - y = nk$ et $z - t = nm$. Dés lors:

$$(x + z) - (y + t) = n(k + m) \text{ et donc } (x + z)R(y + t).$$

1.2 Monoïde

Définition 1.2.1 ***(Monoïde)****:*

Un ***monoïde*** *M est un semi-groupe admettant un élément neutre e.*

Exemple 1.2.1

- $(\mathbb{N}, +)$ est un monoïde d'élément neutre 0.
- $(\mathbb{R}, \times)$ est un monoïde d'élément neutre 1.

- $(2\mathbb{Z}, \times)$ n'est pas monoïde.

Définition 1.2.2

Soit $(S, *)$ un semi-groupe, une partie non vide B de S est appelée un ***sous -semi-groupe*** *si elle stable par l'opération $(*)$ c-à-d:*

$$\forall\ x, y \in (B)^2 :\ x * y \in B$$

Si M est un monoïde de neutre e et si de plus $e \in B$, on dit que B est un ***sous monoïde*** *de M.*

Exemple 1.2.2

1 *Dans tout monoïde M de neutre e, les parties $\{e\}$ et M sont des sous-monoïdes de M, dits triviaux.*

2 *La partie $\mathbb{Z}$ est un sous-monoïde du monoïde $(\mathbb{R}, +)$.*

Proposition 1.2.1 *Soit S un semi-groupe et soit $a \in S$, pour $n \in \mathbb{N}^*$ on définit a^n de manière récursive en posant:*

1 $a^1 = a$.

2 *pour $n \geq 2$: $a^n = a^{n-1} * a$.*

De plus, si M est un monoïde, avec neutre e, on pose $a^0 = e$. Il est clair qu'avec cette définition, on a pour des entiers positifs n et p :

$$a^n * a^p = a^{n+p} \text{ et } (a^n)^p = a^{n \times p}$$

Remarque 1.2.1 *Si l'opération $(*)$ n'est pas commutative, rien ne permet d'écrire des identités comme:*

$$(a * b)^n = a^n * b^n$$

Si a est un élément d'un semi-groupe S, on note T_a la partie de S définie par :

$$T_a = \{a^n \backslash n \in \mathbb{N}\}.$$

T_a est un sous-semi-groupe de M, appelé sous-semi-groupe engendré par a.

Si M est un monoïde, on note encore T_a, la partie de M définie par:

$$T_a = \{a^n \backslash n \in \mathbb{N}\}.$$

T_a est cette fois un sous monoïde de M, c'est le sous monoïde engendré par a.

Exemple 1.2.3

- Dans le monoïde $(\mathbb{Z}, +), T_3$ est un sous-monoïde $3\mathbb{Z}$ de multiple de 3.
- Dans le monoïde $(\mathbb{Z}, \times)$, T_3 est le sous- monoïde constitué de toutes les puissance (positive) de $3 : T_3 = \{1, 3, 9, 27, ...\}$.

Définition 1.2.3 ***(Homomorphisme de monoïde):*** *Soit M , M' deux monoïdes, on appelle homomorphisme de M dans M' tout application h de M dans M' telle que:*

- $h(e_M) = e_{M'}$.
- $\forall (x, y) \in M$, $h(x * y) = h(x) * h(y)$

Un isomorphisme de monoïde est un homomorphisme bijectif de monoïdes.

Exemple 1.2.4

- Soit $(M, *)$ est un monoïde quelconque et soit a un élément fixé de M , la fonction:

$$\begin{array}{rccc} f & : & \mathbb{N} & \rightarrow & M \\ & & n & \longmapsto & a^n \end{array}$$

est un homomorphisme de monoïde de $(\mathbb{N}, +)$ vers $(M, *)$ car:

♦ $\forall n, p \in \mathbb{N} : a^{n+p} = a^n * a^p$.

♦ $a^0 = e$.

Définition 1.2.4

Soit R est une congruence sur le monoïde $(M , *)$, alors $(M/R , *)$ est encore un monoïde. En effet, si e est le neutre de M, $\overline{e}$ est manifestement élément neutre pour l'opération $(*)$ de M/R.

Le monoïde $(M/R , *)$ est appelé monoïde quotient.

Exemple 1.2.5

Soit le monoïde $(\mathbb{N}, +)$ *et soit la fonction* $f : \mathbb{N} \to \mathbb{Z}$ *définie par*

$$f(x) = x^2 - 3x + 2.$$

Posons : $aRb \Leftrightarrow f(a) = f(b)$. *La relation* R *ainsi définie sur* $\mathbb{N}$ *est manifestement d'équivalence mais n'est pas une congruence du monoïde* $(\mathbb{N}, +)$. *Par exemple,* R *contient les couples* $(1, 2)$ *et* $(0, 3)$, *mais ne contient pas le couple somme* $(1, 5)$.

1.3 Mot

Définition 1.3.1 ***(Alphabet)*** *Un alphabet, noté* A, *est un ensemble fini non vide des symboles.*

Exemple 1.3.1

1 $A_1 : \{\bullet, *, \blacklozenge\}$.

2 $A_2 : \{a, b, c, ..., z\}$.

3 $A_3 : \{if, then, else, id, nb, =, +\}$.

Définition 1.3.2 ***(Mot):*** *Un mot, défini sur un alphabet* A, *est un suit finie d'élément de* A.

Exemple 1.3.2

1 *Sur l'alphabet* A_1 *:le mot* $\bullet * \blacklozenge$.

2 *Sur l'alphabet* A_2 *:le mot* $fadila$.

3 *Sur l'alphabet* A_3 *:le mot* $idif = nb$.

Définition 1.3.3 ***(Longueur d'un mot)***

La longueur d'un mot w *défini sur un alphabet* A, *notée* $\mid w \mid$, *est le nombre de symboles qui composent* w.

Exemple 1.3.3

1 *Sur l'alphabet* A_1 : $| \bullet * \blacklozenge | = 3$.

2 *Sur l'alphabet* A_2 :$| fadila | = 6$.

3 *Sur l'alphabet* A_3 :$| idif = nb | = 4$.

Définition 1.3.4 ***(Mot vide):***

Le mot vide, noté ε, *est le mot de longueur* 0 *(autrement dit* $| \varepsilon | = 0$*).*

Définition 1.3.5 *(*A^+*)*

On note A^+ *l'ensemble des mots de longueur supérieure ou égale à* 1 *que l'on peut construire à partire de l'alphabet* A.

Définition 1.3.6 *(*A^**)*

On note A^* *l'ensemble des mots que l'on peut construire à partir de* A, *y compris le mot vide:* $A^* = \{\varepsilon\} \cup A^+$.

Définition 1.3.7 ***(Concaténation)***

Soient deux mots u *et* v *définis sur un alphabet* A. *La concaténation de* u *avec* v, *notée* $u.v$ *ou simplement* uv *s'il n'ya pas d'ambigüité, est le mot formé en faisant suivre les symboles de* u *par les symboles de* v.

On notera un le mot u *concaténé* n *fois* ($u^{\,0} = \varepsilon$, $u^n = u \times (u^{n-1})$ *pour* $n \geq 1$*).*

On écrire par fois u^{-1} *pour désigner (l**e mot -miroir)** de* u, *c'est-à-dire le mot obtenu à partire de* u *en inversant l'ordre des lettres.*

Exemple 1.3.4

1 *Sur l'alphabet* A_2, *si* $u = fadila$ *et* $v = boudraf$ *alors* $uv = fadilaboudraf$

2 *Soit* $u = abc$ *et* $v = aa$ *des mots sur l'alphabet* $A = \{a, b, c\}$ *on aura:*

▶ $vu = aaabc$.

▶ $uv = abcaa$.

- $v^2 = aaaa$.

- $u^{-1}v = cbaaa = (vu)^{-1}$.

Exemple 1.3.5

Considérons l'alphabet $A = \{a, b, c\}$ et le morphisme $\varphi : A^* \to A^*$ défini par $\varphi(a) = abc$, $\varphi(b) = ac$, $\varphi(c) = b$. En effet, pour définir un tel morphisme, on remarquera qu'il suffit de se donner l'image de lettres. On a, par exemple,

$$\varphi(abbc) = \varphi(a)\varphi(b)\varphi(b)\varphi(c) = abcacacb.$$

Remarque 1.3.1 *Le mot u^{-1} n'est évidement pas l'inverse de u pour la concatination (sauf si $u = \varepsilon$).*

Définition 1.3.8 ***Un palindrome*** *est un mot u tel que $u = u^{-1}$.*

Remarque 1.3.2 *La concaténation est associative mais non commutatif (sauf si $\mid A \mid \leq 1$). La concaténation est un loi de composition interne de A^* et ε est son élément neutre . La loi concaténation possède les propriétés suivantes:*

1. *$\forall u, v \in A^*$, $\mid u \cdot v \mid = \mid u \mid + \mid v \mid$.*

2. *$\forall u \in A^*$, $u \cdot u = u \Leftrightarrow u = \varepsilon$.(le mot vide ε est le seul mot idempotent).*

Définition 1.3.9 ***(Préfixe, suffixe et facteur):***

Soient deux mots u et v définis sur un alphabet A.

- u est un préfixe de v si et seulement si $\exists\ w \in A^*$ tel que $uw = v$.

- u est un suffixe de v si et seulement si $\exists\ w \in A^*$ tel que $wu = v$.

- u est un facteur de v si et seulement si $\exists\ w_1 \in A^*$, $\exists\ w_2 \in A^*$ tels que $w_1uw_2 = v$, si $u \neq v$ et $u \neq \varepsilon$, alors u est dit ***facteur propre.***

Définition 1.3.10 *Pour tout alphabet A, $(A^*, \cdot, \varepsilon)$ est un monoïde. Dit le monoïde libre engendré par A.*

Proposition 1.3.1 *Soit M un monoïde quelconque et h une application d'un alphabet A dans M alors il existe un homomorphisme unique $\hat{h}$ de A^* dans M qui prolonge h, c'est-à-dire tel que*

$$\forall\ a \in A :\ \hat{h}(a) = h(a).$$

Preuve.

Existence : Posons $\hat{h}(\varepsilon) = e$ et $\hat{h}(x_1...x_n) = h(x_1)\cdot....\cdot h(x_n)$. Il est facile de voir que $\hat{h}$ est bien un homomorphisme.

Unicité :Soient g et $g\prime$ deux homomorphismes de A^* dans M tels que $\forall x \in A$, $g(x) = g\prime(x)$. Alors $g(\varepsilon) = g\prime(\varepsilon) = e$ et pour tout mot $u = x_1...x_n, g(u) = g(x_1)\cdot....\cdot g(x_n) = g\prime(x_1)\cdot....\cdot g\prime(x_n) = g\prime(u)$. C'est à cause de cet proposition que le monoïde A^* est appelé le monoïde libre, engendré par A. ■

Lemme 1.3.1 ***(Lemme de levy):*** *Soient x, y, z, t des mots tel que*

$$xy = zt.$$

Alors il existe un mot w tel que:

1 *Ou bien $xw = z$ et $y = wt$.*

2 *Ou bien $x = zw$ et $wy = t$.*

Il en résulte en particulier que si $\mid x \mid = \mid z \mid$, le mot w est vide, et donc $x = z$ et $y = t$.

Preuve.

Posons $x = a_1a_2.....a_n$, $y = a_{n+1}.....a_m$ avec $a_i \in A$ et de même $z = b_1b_2.....b_p$, $t = b_{p+1}.....b_q$ avec $b_i \in A$, comme $xy = zt$, on a $m = q$ et $a_i = b_i$ pour $i = 1.....m$, de sorte que $z = a_1.....a_p$ et $t = a_{p+1}.....a_m$. Si $\mid z \mid = p \leq n = \mid x \mid$, posons $w = a_{p+1}.....a_n$. Alors

$$x = zw \text{ et } wy = t.$$

Si $\mid z \mid > \mid x \mid$. Posons $w = a_{n+1}.....a_p$. Alors

$$xw = z \text{ et } y = wt.$$

■

Proposition 1.3.2 *Les conditions nécessaires et suffisantes pour qu'un monoïde M soit un monoïde libre*

1. *il existe un homomorphisme λ de M sur $\mathbb{N}$ ensemble des entiers positifs avec $\lambda^{-1}(0) = \varepsilon$.*

2. *quelque soit $f_1, f_2, f_3, f_4 \in M$ tels que $f_1 f_2 = f_3 f_4$ on a l'une des deux situations suivantes :*

$$\blacklozenge\ \exists f_5 \in M : f_1 = f_3 f_5 \ \text{ et } f_5 f_2 = f_4.$$
$$\blacklozenge\ \exists f_6 \in M : f_3 = f_1 f_6 \ \text{ et } f_6 f_4 = f_2.$$

Exemple 1.3.6

Soit $M \subset A^*$, le monoïde engendré par M est défini par :
$M^* = \{w = x_1 x_2 .....x_n,\ \text{ou pour } 1 \leq i \leq n,\ x_i \in M,\ n \in \mathbb{N}\} \cup \{\varepsilon\}$.
Montrons que M^* est un monoïde libre.

1. *on définit l'homomorphisme λ comme suite :*

$$\begin{array}{rccc} \lambda & : & M^* & \rightarrow & \mathbb{N} \\ & & w & \mapsto & \mid w \mid \end{array}$$

$\lambda^{-1}(0) = \{\varepsilon\}$

2. *quelque soit $f_1, f_2, f_3, f_4 \in M^*$tels que $f_1 f_2 = f_3 f_4$d'après le lemme de levy on a l'une des deux situations suivantes :*

$$\blacklozenge\ \exists f_5 \in M : f_1 = f_3 f_5 \ \text{ et } f_5 f_2 = f_4.$$
$$\blacklozenge\ \exists f_6 \in M : f_3 = f_1 f_6 \ \text{ et } f_6 f_4 = f_2.$$

Proposition 1.3.3 *Soit A un alphabet, soit B une partie de A, pour tout mot $w \in A^*$, la longueur en B de w est le nombre d'occurrence de lettres de B dans mot w . Ce nombre est noté $\mid w \mid_B$.*

En particulier, $\mid w \mid = \mid w \mid_B$. Pour tout lettre $a \in A$, $\mid w \mid_a$est le nombre d'occurrence de a dans w.

On a :

$$\mid w \mid_B = \sum_{b \in B} \mid w \mid_b$$

1.4 Langage

Définition 1.4.1 ***(langage)***

Un ***langage,*** *défini sur un alphabet A est un ensemble de mots définis sur A . Autrement dit un langage est un sous-ensemble de A^*.*

Deux langage particuliers sont indépendants de l'alphabet A :

-Le langage vide ($\mathcal{L} = \phi$).

-Le langage contenant le seul mot vide ($\mathcal{L} = \{\varepsilon\}$).

Exemple 1.4.1

- *L'ensemble $\mathcal{L}_1 = \{a^n b^n \setminus n > 0\}$ est un langage.*
- *L'ensemble $\mathcal{L}_2 = \{01^n \setminus n \in \mathbb{N}\}$ est un langage.*
- *L'ensemble $\mathcal{L}_3 = \{ a, aa, ab, ba, bb\}$ est un langage sur l'alphabet $A = \{a, b\}$.*

Remarque 1.4.1 *: A^* est le plus grand langage sur A au sens de l'inclusion.*

Opérations ensemblistes définies sur les langages:

Soient deux langages $\mathcal{L}_1$et $\mathcal{L}_2$ respectivement définis sur les alphabets A_1 et A_2 :

▶ L'union de $\mathcal{L}_1$ et $\mathcal{L}_2$ est langage défini $A_1 \cup A_2$ contenant tout les mots qui sont contenus dans $\mathcal{L}_1$, soit contenus dans $\mathcal{L}_2$:

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{u \setminus u \in \mathcal{L}_1 \text{ ou } u \in \mathcal{L}_2 \}.$$

▶ L'intersection de $\mathcal{L}_1$ et $\mathcal{L}_2$ est le langage défini sur $A_1 \cap A_2$ contenant tout les mots qui sont contenus à la fois dans $\mathcal{L}_1$ et dans $\mathcal{L}_2$:

$$\mathcal{L}_1 \cap \mathcal{L}_2 = \{u \setminus u \in \mathcal{L}_1 \text{ et } u \in \mathcal{L}_2 \}.$$

▶ Le complément de $\mathcal{L}_1$ est le langage défini sur A_1 contenant tous les mots qui ne sont pas dans $\mathcal{L}_1$:

$$C(\mathcal{L}_1) = \{u \setminus u \in A_1^* \text{ et } u \notin \mathcal{L}_1 \}.$$

▶ La différence de $\mathcal{L}_1$ et $\mathcal{L}_2$ est le langage défini sur A_1 contenant tous les mots de $\mathcal{L}_1$ qui ne sont pas dans $\mathcal{L}_2$:

$$\mathcal{L}_1 - \mathcal{L}_2 = \{u \setminus u \in \mathcal{L}_1 \text{ et } u \notin \mathcal{L}_2 \}.$$

Exemple 1.4.2

- *Sur l'alphabet* $A = \{0, 1\}$, *on considère les langages* $\mathcal{L}_1$ *et* $\mathcal{L}_2$ *définis par :*

-$\mathcal{L}_1 = \{01^n \setminus n \in \mathbb{N}\}$.
-$\mathcal{L}_2 = \{0^n 1 \setminus n \in \mathbb{N}\}$.
$\mathcal{L}_1 \cap \mathcal{L}_2 = \{01\}$.

Définition 1.4.2 ***(Produit de deux langage)***

Le produit de deux langages $\mathcal{L}_1$ *et* $\mathcal{L}_2$ *est le langage*

$$\mathcal{L}_1\mathcal{L}_2 = \{xy \setminus x \in \mathcal{L}_1,\ y \in \mathcal{L}_2\}.$$

On a en particulier $\mathcal{L}_1\{\varepsilon\} = \{\varepsilon\}\mathcal{L}_1$, *on vérifie que*

$$\mathcal{L}_1(\mathcal{L}_2 \cup \mathcal{L}_3) = \mathcal{L}_1\mathcal{L}_2 \cup \mathcal{L}_1\mathcal{L}_3.$$

Le produit des langages est associative, mais non commutatif.

Exemple 1.4.3

▶ On considère les deux langages $\mathcal{L}_1 = \{00, 11\}$ et $\mathcal{L}_2 = \{0, 1, 01\}$ définis sur $\{0, 1\}$:

-$\mathcal{L}_1\mathcal{L}_2 = \{000, 001, 0001, 110, 111, 1101\}$.

► $\mathcal{L}_1 = \{01^n \setminus n \in \mathbb{N}\}$, $\mathcal{L}_2 = \{0^n 1 \setminus n \in \mathbb{N}\}$ sur $A = \{0, 1\}$:

-$\mathcal{L}_1\mathcal{L}_2 = \{01^n 0^m 1 \setminus n \in \mathbb{N},\ m \in \mathbb{N}\}$.

Définition 1.4.3 ***(Puissance d'un langage)***

La puissance de $\mathcal{L}_1$ est définie par :

$\mathcal{L}_1^0 = \{\varepsilon\}$, $\mathcal{L}_1^1 = \mathcal{L}_1$ *et* $\mathcal{L}^{n+1} = \mathcal{L}^n\mathcal{L}$, *pour* $n \geq 1$. *En particulier, si A un alphabet, A^n est l'ensemble des mots de Longueur n*

- $\mathcal{L}^{-1} = \{u^{-1} \setminus u \in \mathcal{L}\}$, *sauf dans le cas où* $\mathcal{L} = \{\varepsilon\}$, $\mathcal{L}^{-1}$ *n'est pas évidement l'inverse de $\mathcal{L}$ pour $(*)$.*

Exemple 1.4.4

► *On considère* $\mathcal{L} = \{00, 11\} : \mathcal{L}^2 = \{0000, 0011, 1100, 1111\}$.

► *On considère* $\mathcal{L} = \{01^n\} : \mathcal{L}^2 = \{01^n 01^m \setminus n \in \mathbb{N},\ m \in \mathbb{N}\}$.

Définition 1.4.4 ***(Fermeture itérative d'un langage):***

La fermeture itérative d'un langage $\mathcal{L}$ est l'ensemble des mots formés par une concaténation de mots de $\mathcal{L}$

$$\mathcal{L}^* = \bigcup_{n\geq 0} \mathcal{L}^n = \{l_1 l_2 \ldots\ldots\ldots\ldots l_n \setminus\ n \geq 0,\ l_1, l_2, \ldots\ldots\ldots\ldots l_n \in \mathcal{L}\}.$$

L'opération plus est définie de manière similaire :

$$\mathcal{L}^+ = \bigcup_{n\geq 1} \mathcal{L}^n = \{l_1 l_2 \ldots\ldots\ldots\ldots l_n \setminus\ n \geq 1,\ l_1, l_2, \ldots\ldots\ldots\ldots l_n \in \mathcal{L}\}.$$

Exemple 1.4.5 *Soit $\mathcal{L} = \{a, ba\}$. Les mots de $\mathcal{L}^*$, classés par longueur, sont:*

Longueur	*Mots*
0	ε
1	a
2	aa, ba
3	aaa, aba, baa
4	$aaaa, aaba, abaa, baaa, baba$
5	$aaaaa,\ aaaba,\ aabaa,\ abaaa,\ ababab, aaaa,\ baaba,\ babaa$

Définition 1.4.5 *Soient $\mathcal{L}_1$ et $\mathcal{L}_2$ deux langages sur A. On appelle* ***quotient gauche*** *de $\mathcal{L}_2$par $\mathcal{L}_1$ et l'on note $\mathcal{L}_1^{-1} \cdot \mathcal{L}_2$ le langage défini par :*

$$\mathcal{L}_1^{-1} \cdot \mathcal{L}_2 = \{u \in A^*,\ \exists v \in \mathcal{L}_1,\ v \cdot u \in \mathcal{L}_2\}.$$

De la même façon, on appelle ***quotient droit*** *de $\mathcal{L}_1$par $\mathcal{L}_2$ et l'on note $\mathcal{L}_1 \cdot \mathcal{L}_2^{-1}$ le langage défini par :*

$$\mathcal{L}_1 \cdot \mathcal{L}_2^{-1} = \{u \in A^*,\ \exists v \in \mathcal{L}_2,\ u \cdot v \in \mathcal{L}_1\}.$$

Définition 1.4.6 *Soit un alphabet. Supposons que, $0, e, +, \cdot, (,)$, sont des symboles n'appartenant pas à A . L'ensemble R_A des expressions régulières sur A est défini récursivement par :*

- 0 *et* e *appartiennent à* R_A.
- *pour tout* $\sigma \in A$, σ *appartient à* R_A.
- *si* ϕ *et* ψ *appartiennent à* R_A, *alors:*

  -$(\phi + \psi)$ appartient à R_A.
  -$(\phi \times \psi)$ appartient à R_A.
  -ϕ^* appartient à R_A.

Exemple 1.4.6

Si $A = \{a, b\}$, voici quelques exemples d'expressions régulières :

$$\begin{aligned} \alpha_1 &= (e + (a \cdot b)), \\ \alpha_2 &= (((a \cdot b) \cdot a) + b^*)^* \\ \alpha_3 &= ((a + b)^* \cdot (a \cdot b)). \end{aligned}$$

Définition 1.4.7 *Un langage $\mathcal{L}$ sur A est* ***régulier*** *s'il existe une expression régulière $\phi \in R_A$ telle que $\mathcal{L} = \mathcal{L}(\phi)$.*

Si ϕ et ψet sont deux expressions régulières telles que $\mathcal{L}(\phi) = \mathcal{L}(\psi)$, alors on dit que ϕ et ψ sont équivalentes.

1.5 Automates finis déterministes:

Définition 1.5.1 ***(Automate) :***

Un automate est un 5-uplet, $T =< A, Q, i, f, \delta >$ *où*

– A *est un ensemble fini, l'alphabet d'entrée.*

– Q *est un ensemble, l'ensemble des états.*

– $i \subset Q$ *est l'ensemble des états de initial.*

– $f \subset Q$ *est l'ensemble des états finals.*

– $\delta : Q \times A \rightarrow Q$ *est la fonction de transitions de* T*.*

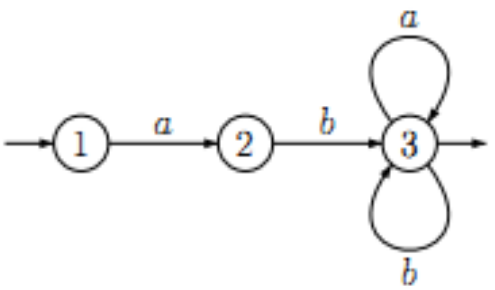
Définition 1.5.2 ***(Automate fini):***

Un automate fini est un automate $T =< A, Q, i, f, \delta >$ *dont l'ensemble d'états* Q*, est fini.*

Exemple 1.5.1

Posons, $A = \{a, b\}$, $Q = \{1, 2, 3\}$, $i = \{1\}$, $f = \{3\}$.

$\delta = \{(1, a, 2), (2, b, 3), (3, a, 3), (3, b, 3)\}$, $T =< A, Q, i, f, \delta >$*, est un automate fini.*



Définition 1.5.3 *Un automate* $T =< A, Q, i, f, \delta >$ *est déterministe(AFD) ssi:*

– $Card(i) = 1$.

– $\forall q \in Q, \forall x \in A, Card(\{q' \in Q,\ (q, x, q') \in \delta\}) \leq 1$.

T *est dit déterministe complet ssi:*

– $Card(i) = 1$.

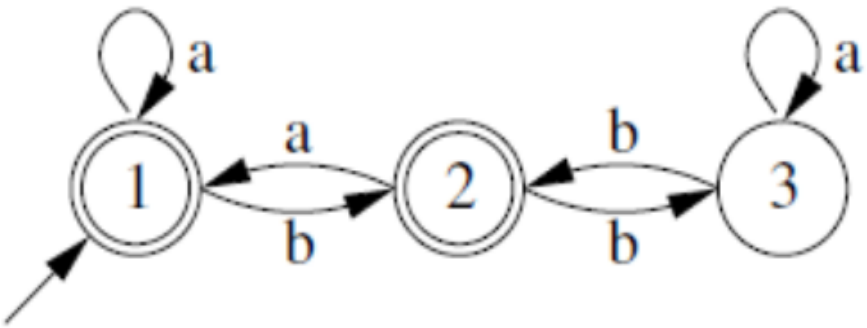
– $\forall q \in Q, \forall x \in A, Card\ (\{q' \in Q,\ (q, x, q') \in \delta\}) = 1$.

Exemple 1.5.2

L'automate $T =< A, Q, i, f, \delta >$ *où* $Q = \{1, 2, 3\}$, $i = 1$, $f = \{1, 2\}$, $A = \{a, b\}$ *et où la fonction de transition est donnée par*

δ	a	b
1	1	2
2	1	3
3	3	2

est représenté à la figure



Définition 1.5.4 *Soit* $T =< A, Q, i, f, \delta >$ *un AFD. On étend naturellement la fonction de transition* δ *à* $Q \times A^*$ *de la manière suivante :*

$$\delta(q, \varepsilon) = q$$

et

$$\delta(q, \sigma w) = \delta(\delta(q, \sigma), w), \sigma \in A,\ w \in A^*.$$

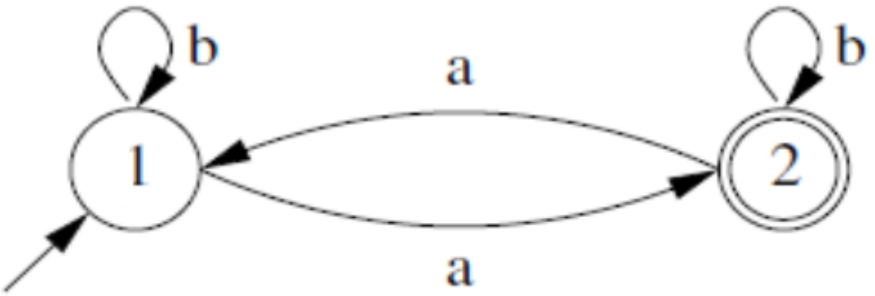
On appelle langage accepté (ou reconnu) par A, noté $\mathcal{L}(A)$, l'ensemble des mots acceptées (ou reconnnus) par A.

$$\mathcal{L}(A) = \{w \in A^* \backslash\ \delta(i, w) \in f\}.$$

Définition 1.5.5 *Un langage* $\mathcal{L} \subset A^*$ *est dit reconnaissable ssi, il existe un automate fini* T *sur* A, *tq* $\mathcal{L} = \mathcal{L}(A)$. *On note* $Rec(A^*)$ *l'ensemble des langages reconnaissables sur* A^*.

Exemple 1.5.3 *L'automate T accepte exactement le langage formé des mots sur l'alphabet $\{a, b\}$ et contenant un nombre impair de a.*

$$\mathcal{L}(A) = \{w \in \{a, b\}^* :| \ w \mid_a \equiv 1 (\mathrm{mod}\, 2)\}$$



Théorème 1.5.1 *Pour tout automate fini T il existe un automate fini déterministe et complet $\mathcal{B}$ tel que:*

$$\mathcal{L}(T) = \mathcal{L}(\mathcal{B}).$$

Preuve.

Soit $T =< A, Q, i, f, \delta >$. On définit un automate déterministe $\mathcal{B}$ qui à pour ensemble d'états l'ensemble $\mathcal{P}(Q)$ des parties de Q, pour état initial i, et pour ensemble d'états terminaux $V = \{S \subset Q \setminus S \cap f \neq \phi\}$. On définit enfin la fonction de transition de $\mathcal{B}$ pour $S \subset \mathcal{P}(Q)$ et $a \in A$ par:

$$S \cdot a = \{q \in Q \setminus \exists s \in S : (s, a, q) \in f\}.$$

Nous prouvons par récurrence sur la longueur d'un mot w que

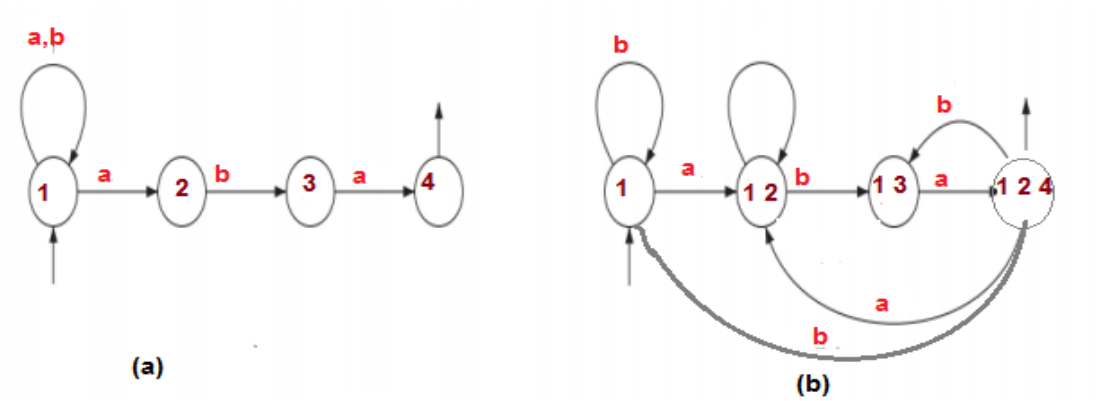
$$S \cdot w = \left\{q \in Q \ \setminus \exists s \in S : s \xrightarrow{w} q\right\}.$$

Ceci est clair si $w = \varepsilon$, et est vrai par définitions si w est une lettre. Posons $w = va$, avec $v \in A^*$ et $a \in A^*$. Alors comme par définition $S \cdot w = (S \cdot v) \cdot a$, on a $q \in S \cdot w$ si et seulement s'il existe une fléche (p, a, q), avec $p \in S \cdot v$, donc telle qu'il existe un calcul $s \xrightarrow{v} p$ pour un $s \in S$. Ainsi $q \in S \cdot w$ si et selement s'il existe un calcul $s \xrightarrow{w} q$ avec $s \in S$. Ceci prouve l'assertion.

Maintenant, $w \in \mathcal{L}(T)$ si et seulement s'il existe un calcul réussi d'étiquette w, ce qui signifie donc que $i \cdot w$ contient au moins un état final de T, en d'autres termes que $i \cdot w \cap f \neq \phi$. Ceci prouve l'égalité $\mathcal{L}(T) = \mathcal{L}(\mathcal{B})$. ■

Exemple 1.5.4

L'automate $T =< A, Q, i, f, \delta >$ *où* $Q = \{1, 2, 3, 4\}$, $i = 1$, $f = \{4\}$, $A = \{a, b\}$.



(a) Un automate non déterministe réconisable l'ensemble de mots $A = \{a, b\}^*$, *(b) un automate déterministe réconisable cet ensemble.*

Définition 1.5.6 *Soit* $\mathcal{L} \subseteq A^*$*un langage arbitraire. Si* w *est un mot sur* A^*, *on dénote par* $w^{-1} \cdot \mathcal{L}$ *l'ensemble des mots qui concaténés avec* w, *appartiennent a* $\mathcal{L}$, *i.e,*

$$w^{-1} \cdot \mathcal{L} = \{u \in A^* \setminus\ wu \in \mathcal{L}\}\,.$$

On définit une relation sur A^*, *notée* $\backsim_{\mathcal{L}}$, *de la manière suivante. Pour tous* x, $y \in A^*$

$$x \backsim_{\mathcal{L}} y \Leftrightarrow x^{-1}\mathcal{L} = y^{-1}\mathcal{L}.$$

En d'autres termes, $x \backsim_{\mathcal{L}} y$ *si et seulement si pour tout mot* $w \in A^*$,

$$xw \in \mathcal{L} \Leftrightarrow yw \in \mathcal{L}$$

Notons que la notation la plus répandue dans la littérature est $w^{-1}\mathcal{L}$.

Proposition 1.5.1 *Soit $\mathcal{L} \subseteq A^*$un langage, la relation $\backsim_{\mathcal{L}}$ est une relation d'équivalence. Il s'agit même d'une congruence à droite, i.e,*

$$\forall z \in A^*,\ x \backsim_{\mathcal{L}} y \Rightarrow xz \backsim_{\mathcal{L}} yz.$$

Remarque 1.5.1 *On parle souvent pour $\backsim_{\mathcal{L}}$ de la congruence de Nerode.*

On note $[w]_{\mathcal{L}}$ la classe d'équivalence du mot w pour la relation $\backsim_{\mathcal{L}}$,

$$[w]_{\mathcal{L}} = \{u \in A^* \backslash\ u \backsim_{\mathcal{L}} w\}$$

Exemple 1.5.5

Soit le langage: $\mathcal{L} = \{w \in \{a,b\}^* :|\ w\ |_a \equiv 0 (\mathrm{mod}\, 3)\}$.

Pour ce langage, on a par exemple:

-abbaba $\backsim_{\mathcal{L}}$ aaa car $abbaba^{-1} \cdot \mathcal{L} = aaa^{-1} \cdot \mathcal{L} = \mathcal{L}$.

-b $\nsim_{\mathcal{L}}$ ab car pour $u = aa$, $bu \notin \mathcal{L}$ *et* $abu \in \mathcal{L}$.

-aba $\nsim_{\mathcal{L}}$ bab car pour $u = a$, $abau \in \mathcal{L}$ *et* $babu \notin \mathcal{L}$.

-a $\backsim_{\mathcal{L}}$ ababaa car $a^{-1} \cdot \mathcal{L} = ababaa^{-1} \cdot \mathcal{L} = \{w \in \{a,b\}^* :|\ w\ |_a \equiv 2 (\mathrm{mod}\, 3)\}$.

En effet, pour $w \in \{a,b\}^*$,

si $|\ w\ |_a \equiv_3 0$, *alors* $w^{-1} \cdot \mathcal{L} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 0\}$

$$et \qquad [w]_{\mathcal{L}} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 0\},$$

si $|\ w\ |_a \equiv_3 1$, *alors* $w^{-1} \cdot \mathcal{L} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 2\}$

$$et \qquad [w]_{\mathcal{L}} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 1\},$$

si $|\ w\ |_a \equiv_3 2$, *alors* $w^{-1} \cdot \mathcal{L} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 1\}$

$$et \qquad [w]_{\mathcal{L}} = \{u \in \{a,b\}^* :|\ u\ |_a \equiv_3 2\}.$$

Lemme 1.5.1 *Soit $\mathcal{L}$ un langage et u, v deux mots sur A. On a*

$$(uv)^{-1} \cdot \mathcal{L} = v^{-1} \cdot (u^{-1} \cdot \mathcal{L}).$$

Démonstration. Si w appartient à $(uv)^{-1}\mathcal{L}$, cela signifie que uvw appartientn à $\mathcal{L}$. En d'autres termes, vw appartient à $u^{-1} \cdot \mathcal{L}$ et ainsi w appartient à $v^{-1} \cdot (u^{-1} \cdot \mathcal{L})$. La démonstration de l'autre inclusion est identique. ■

Définition 1.5.7 *On définit l'automate minimal*

$$T_{\mathcal{L}} = (Q_{\mathcal{L}}, A\ ,\ i_{\mathcal{L}}, f_{\mathcal{L}},\ \delta_{\mathcal{L}}\ )$$

d'un langage $\mathcal{L} \subseteq A^*$ comme suit :

- $Q_{\mathcal{L}} = \{w^{-1} \cdot \mathcal{L}\ \backslash w \in A^*\}$
- $i_{\mathcal{L}} = \varepsilon^{-1} \cdot \mathcal{L} = \mathcal{L}$
- $f_{\mathcal{L}} = \{w^{-1} \cdot \mathcal{L}\ \backslash w \in A^*\} = \{q \in Q_{\mathcal{L}}\ \backslash \varepsilon \in q\}$
- $\delta_{\mathcal{L}}(q,\, \sigma) = \sigma^{-1} \cdot q$, *pour tout* $q \in Q_{\mathcal{L}}$, $\sigma \in A$

la fonction de transition de l'automate s'étend à $Q_{\mathcal{L}}$ par

$$\delta_{\mathcal{L}}(q, w) = w^{-1} \cdot q,\ \forall q \in Q_{\mathcal{L}},\ w \in A^*.$$

Proposition 1.5.2 *L'automate minimal d'un langage $\mathcal{L} \subseteq A^*$ accepte $\mathcal{L}$.*

Démonstration. En effet, soit $w \in A^*$

$$w \in \mathcal{L}(T_{\mathcal{L}}) \Leftrightarrow \delta_{\mathcal{L}}(i_{\mathcal{L}}, w) \in f_{\mathcal{L}} \Leftrightarrow w^{-1} \cdot \mathcal{L} \in f_{\mathcal{L}} \Leftrightarrow w \in \mathcal{L}.$$

On a utilisé le fait que

$$\delta_{\mathcal{L}}(i_{\mathcal{L}}, w) = \delta_{\mathcal{L}}(\varepsilon^{-1} \cdot \mathcal{L}, w) = w^{-1} \cdot (\varepsilon^{-1} \cdot \mathcal{L}) = (\varepsilon w)^{-1} \cdot \mathcal{L}.$$

■

Proposition 1.5.3

Un langage $\mathcal{L} \subset A^*$ est régulier si et seulement si son automate minimal $T_{\mathcal{L}}$ est fini.

1.6 Monoïde syntaxique

Définition 1.6.1 *Soit $\mathcal{L} \subseteq A^*$ un langage (régulier ou non). On définit sur A^* la relation suivante. Soient $u, v \in A^*$. On a*

$$u \equiv_{\mathcal{L}} v \Leftrightarrow (\forall x, y \in A^* : xuy \in \mathcal{L} \Leftrightarrow xvy \in \mathcal{L}$$

Il est facile de vérifier qu'il s'agit d'une relation d'équivalence sur et même d'une congruence (à droite et à gauche), i.e, pour tout $\sigma \in A$,

$$u \equiv_{\mathcal{L}} v \Rightarrow u\sigma \equiv_{\mathcal{L}} v\sigma \text{ et } \sigma u \equiv_{\mathcal{L}} \sigma v.$$

On parle souvent de la congruence syntaxique $\equiv_{\mathcal{L}}$ et on dit que u et v sont syntaxiquement équivalents.

Remarque 1.6.1 *Nous avons montré que $\equiv_{\mathcal{L}}$ est une congruence à gauche et à droite. Mais en toute généralité, une congruence doit respecter la propriété suivante: si $x \equiv_{\mathcal{L}} x^{'}$ et si $y \equiv_{\mathcal{L}} y,^{'}$ alors $xy \equiv_{\mathcal{L}} x^{'}y^{'}$, c'est- a-dire qu'elle doit bien se comporter par rapport au produit envisagé, à savoir ici, la concaténation. Et ceci est bien le cas car pour tous α, $\beta \in A^*$ il vient*

$$\alpha xy\beta \in \mathcal{L} \Leftrightarrow \alpha x^{'}y\beta \in \mathcal{L} \Leftrightarrow \alpha x^{'}y^{'}\beta \in \mathcal{L}.$$

On notera simplement $[w]$ la classe d'équivalence pour $\equiv_{\mathcal{L}}$ étant convenu que la relation $\equiv_{\mathcal{L}}$ est sous-entendue. Bien évidemment, $[w]$ est un ensemble de mots, donc un langage.

Exemple 1.6.1

Soit $\mathcal{L}$, le langage formé des mots sur $\{a, b\}$ ne contenant pas deux bb consécutifs. On remarque tout d'abord que

$$xay \in \mathcal{L} \Leftrightarrow x \in \mathcal{L} \text{ et } y \in \mathcal{L}.$$

De là, on en tire que la classe de a pour la congruence syntaxique $\equiv_{\mathcal{L}}$est de la forme

$$[a] = \{awa \setminus w \in \mathcal{L}\} \cup \{a\}.$$

En particulier, $\varepsilon \notin [a]$

Nous allons voir qu'on peut munir l'ensemble quotient $A^* / \equiv_{\mathcal{L}}$, i.e., l'ensemble des classes d'équivalence pour $\equiv_{\mathcal{L}}$,d'une structure de Monoïde

Définition 1.6.2 *.Soit l'opération*

$$\begin{array}{rcccccc} \circ & : & A^*/\equiv_{\mathcal{L}} & \times & A^*/\equiv_{\mathcal{L}} & \rightarrow & A^*/\equiv_{\mathcal{L}} \\ & & ([x] & , & [y]) & \mapsto & [x]\circ[y] \end{array}$$

définie par

$$[x]\circ[y] = [z] \text{ si } [x]\cdot[y] \subseteq [z].$$

où représente l'opération de concaténation de langages. L'application est bien définie.

Remarque 1.6.2 *Il est évident que*

$$[x]\circ[y] = [xy].$$

Proposition 1.6.1 *Muni de l'opération $\circ$, l'ensemble quotient $A^*/\equiv_{\mathcal{L}}$ possède une structure de monoïde.*

Démonstration. Le neutre est $[\varepsilon]$, i.e., pour tout $x \in A^*$, on a

$$[x]\circ[\varepsilon] = [x].$$

De plus, l'opération $\circ$ est associative, i.e., pour tous $x, y, z \in A^*$,

$$([x]\circ[y])\circ[z] = [x]\circ([y]\circ[z]).$$

■

Définition 1.6.3 *Le monoïde $(A^*/\equiv_{\mathcal{L}}, \circ)$ est le monoïde syntaxique de $\mathcal{L}$. On note simplement p le morphisme canonique :*

$$\begin{array}{rcccc} p & : & A^* & \rightarrow & A^*/\equiv_{\mathcal{L}} \\ & & w & \mapsto & [x] \end{array}$$

Corollaire 1.6.1 *Un langage $\mathcal{L}$ est régulier si et seulement si son monoïde syntaxique est fini.*

Chapitre 2

Codes de longueurs variables

2.1 Code

Cette section contient les définitions des notions de code, préfixe (suffixe ,bifix) code, code maximal, et morphisme de codage et donne des exemples.

Définition 2.1.1 *Une partie X de A^*est préfixe (resp. suffixe) si aucun facteur gauche (resp. droite) propre d'un mot de X n'est dans X, en symboles :*

$XA^+ \cap X = \phi$ ou $X^{-1}X = \{\varepsilon\}$ resp ($A^+X \cap X = \phi$ ou $X^{-1}X = \{\varepsilon\}$) .

*-X est **bipréfixe** ou **bifixe** s'il est à la fois préfixe et sufixe.*

Définition 2.1.2 *$X \subset A^*$ est dit préfixe si $\forall x,\ x' \in X\ x \leq x' \Rightarrow x = x'$ où $\leq$ signifie être un facteur gauche (ou préfixe).*

Définition 2.1.3 ***(Code):***

On appelle code toute partie C d'un monoïde libre A^*qui vérifie la condition suivante : pour tout $x_1, ..., x_n; y_1, ..., y_m \in C$,

$$x_1.....x_n = y_1.....y_m \Rightarrow n = m,\ x_i = y_i,\ i = 1.....n.$$

En d'autres termes, C est un code si tout mot de C^*se factorise, de manière unique, en un produit de mots de C. Lorsqu'un ensemble n'est pas un code, on s'en aperçoit en général assez facilement, en exhibant une double factorisation. Il est plus difficile d'établir qu'un ensemble est effectivement un code.

Exemple 2.1.1 *L'ensemble* $\{a,\ ab, ba\}$ *n'est pas un code puisque le mot* $w = aba$ *.à deux factorisations distincts*

$$w = (ab)a = a(ba).$$

L'ensemble $C = \{b, ab, baa, abaa, aaaa\}$ *est un code. En effet, un mot de* C^* *qui aurait deux factorisations commencerait par baa ou par abaa. Regardons le premier cas (le deuxième est en fait similaire). L'une des factorisations commencerait par b et l'autre par baa. Pour compléter ces factorisations, il faut compenser l'excès de la deuxième factorisation, soit le mot aa. Pour cela, on doit ajouter à la première factorisation le seul mot de C commençant par aa, à savoir* a^4*. La première factorisation commence donc par* $(b, aaaa)$*. Mais alors, la deuxième factorisation ne peut être complétée que par* a^4*, et devient* $(baa, aaaa)$*. On est alors revenu au point de départ, et il n'y a donc pas de double factorisation possible.*

Définition 2.1.4 *Soit* C *une partie de* A^**. On dit que* C *est un code si* C^* *est libre de base* C*.*

Définition 2.1.5 *Un* ***code à longueur variable*** *est tel que les différents mots de code n'ont pas nécessairement la même longueur.*

Proposition 2.1.1 ***(Premières propriétés des codes)***

Soit C *un code sur* A

i $\varepsilon \notin C$*.*

ii $\forall P \subset C, P$ *est un code.*

iii *Soit* B *un alphabet, tout morphisme* $\theta : B^* \rightarrow A^*$ *qu'induit une bijection de* B *sur* C *est injectif.*

Réciproquement, s'il existe un morphisme injectif $\theta : B^* \rightarrow A^*$ *tel que* $C = \theta(B)$ *alors* C *est un code.*

Cette dernière propriété (qui pourrait aussi servir de définition aux codes) traduit la notion intuitive de code. En effet le morphisme de codage permet de coder les mots de B^* *dans* A^* *et l'injectivité permet d'assurer que le décodage est possible.*

Démonstration.

i $\varepsilon = \varepsilon\varepsilon$ donc ε n'a pas une unique factorisation.

ii Toute factorisation d'un mot w dans P est une factorisation dans C et est donc unique.

iii Soit $\theta : B^* \rightarrow A^*$ qu'induit une bijection de B sur C. Soient $u, v \in (B^*)^2$ tels que $\theta(u) = \theta(v)$.

Si $u = \varepsilon$ supposons $v \neq \varepsilon$ alors v contient au moins une lettre b et par hypothèse $\theta(b) \in C$ or $\varepsilon \notin C$ donc $\mid \theta(b) \mid > 0$. On en déduit que $\mid \theta(v) \mid > 0$ ce qui est absurde car $\theta(u) = \varepsilon$.

Sinon $u = b_{\ 1}.....b_n$ et $v = b'_1.....b'_m$. On a alors $\theta(b_1).... \theta(b_n) = \theta(b'_1).....\theta(b'_m)$ avec $\theta(b_i)$, $\theta(b'_j) \in C$. Or C est un code donc $n = m$ et $\forall i \in \theta(b_i) = \theta(b'_i)$ or induit une bijection de B sur C donc $\forall i\ b_i = b'_i$ i.e $u = v$. Donc θ est injective. Réciproquement, soit $\theta : B^* \rightarrow A^*$ morphisme injectif, supposons qu'on a n, $m \in \mathbb{N}$ et $(x_i)_{\ i=1.....n}$, $(x'_j)_{\ j=1.....m} \in C = \theta(B)$ telle que $x_1.....x_n = x'_1.....x'_m$. Soient $(b_i)_{i=1.....n}$, $(b'_j)_{i=1.....m} \in B$ telle que $\forall i\ x_i = \theta(b_i)$ et $\forall j\ x'_j = \theta(b'_j)$.On a donc $\theta(b_1.....b_n) = \theta(b'_1.....b'_m)$ or θ est injective donc $b_1.....b_n = b'_1.....b'_m$ D'où $n = m$ et $\forall i\ b_i = b'_i$ et donc $\forall i\ x_i = \theta(b_i) = \theta(b'_i) = x'_i$.

■

Définition 2.1.6 ***(morphisme de codage)***

Un morphisme $\beta : B^* \rightarrow A^*$ qui est injective et tel que $C = \beta(B)$, est appelé morphisme codage pour C. Pour tout code $C \subset A^*$, L'existence d' un morphisme de codage pour C est simple :il suffit de prendre une bijection d'un ensemble B sur C et à l'étendre à un morphisme de B^*dans A^*. Dans ce contexte, l'alphabet B est appelé le source de l'alphabet, et l'alphabet A est l'alphabet de canal.

Corollaire 2.1.1 *Soit $\alpha : A^* \rightarrow X^*$ un morphisme injectif. Si C est un code sur A, alors $\alpha(C)$ est un code sur X. Si Y est un code sur X, puis $\alpha^{-1}(Y)$ est un code sur A.*

Preuve.

Soit $\beta : B^* \rightarrow A^*$ un morphisme de codage pour C. alors $\alpha(\beta(B)) = \alpha(C)$ depuis $\alpha \circ \beta : B^* \rightarrow X^*$ est un morphisme injectif, montre que $\alpha(C)$ est un code.

Inversement, soit $C = \alpha^{-1}(Y)$, soit n, $m \geq 1$, $x_1x_2....x_n$, $x_1^{'}x_2^{'}....x_m^{'} \in C$ telle que

$$x_1x_2....x_n = x_1^{'}x_2^{'}....x_m^{'}.$$

Alors

$$\alpha(x_1)\alpha(x_2)....\alpha(x_n) = \alpha(x_1^{'})(x_2^{'})....\alpha(x_m^{'}).$$

Maintenant Y est un code, donc $n = m$ et $\alpha(x_i) = \alpha(x_i^{'})$ pour $i = 1, ..., n$. L' injectivité de α implique que $x_i = x_i^{'}$ pour $i = 1, ..., n$, montrant que C est un code. ■

Proposition 2.1.2

$\forall C \subset A^*$, *C préfixe* $\Rightarrow$ *C est un code.*

Démonstration.

Supposons que C n'est pas un code. Soit w de longueur minimale tel que w ait deux factorisations dans C. On a donc $n, m \in \mathbb{N}$ et $(x_i)_{\ i=1.....n}$, $(x_j^{'})_{\ j=1.....m} \in C$ tels que $w = x_1....x_n = x_1^{'}.....x_m^{'}$.Comme w est de longueur minimale, on $x_1 \neq x_1^{'}$ et donc $x_1 < x_1^{'}$ ou $x_1^{'} < x_1$ce qui rentre en contradiction avec C préfixe. ■

Exemple 2.1.2 *Soit $A = \{a, b\}$ et $C = \cup_{n\geq 0} a^nbA^n$, C est préfixe car $a^nbu = a^mbv \Rightarrow m = n$ et donc $u = v$. C'est donc un code sur A.*

Proposition 2.1.3

Soit C une partie de A^*

$$C \text{ est un code} \Leftrightarrow (\forall w \in A^* : (C^*w \cap C^* \neq \phi) \text{ et } (wC^* \cap C^* \neq \phi)) \Rightarrow w \in C^*. \qquad (1)$$

Démonstration.

Supposons que C vérifie la condition (1).

Soit $w \in C^+$tel que : $w = x_1x_2....x_n$,où pour $1 \leq i \leq n$, $x_i \in C$, $n \in \mathbb{N}$.

Pour montrer que C est un code, il suffit de vérifier que cette factorisation est unique.

Si $w = x_i$ avec $x_i \in C$ et $i \in \mathbb{N}^*$, le résultat est vrai.

On peut supposer que ceci est déjà établi pour tous les mots de C^+ qui sont strictement plus courts que w.

Soit donc $w = y_1y_2....y_m$ où pour $1 \leq j \leq m$, $y_j \in C$, $m \in \mathbb{N}$, une autre factorisation de w, d'après l'hypothèse d'induction , cette factorisation est la même que la précédente si et seulement si $x_1 = y_1$.

Si x_1 est différent de y_1, on pourrait supposer sans perte de généralité que x_1 est plus court que y_1. On aurait donc $y_1 = x_1f$, pour un certain mot f de A^*, $f \notin C^*$, $C^*w \cap C^* \neq \phi$, $x_1x_2....x_n = x_1fy_2....y_m$ donc $fC^* \cap C^* \neq \phi$. Ceci est en contradiction avec le fait que : $\forall w \in A^* : \ (C^*w \cap C^* \neq \phi)$ et $(wC^* \cap C^* \neq \phi)) \Rightarrow w \in C^*$.

Réciproquement :

Supposons que C ne vérifie pas la condition (1), donc il existe $w \notin C^*$ tel que :

$a_{i_1}.....a_{i_m}w = a_{j_1}.....a_{jn}$, $a_{i_1} \neq a_{j_1}$, $wa_{k_1}.....a_{k_p} = a_{l_1}.....a_{lq}$, ou'pour $1 \leq \alpha \leq m$, $a_{i_\alpha} \in C$, $m \in \mathbb{N}$, et ou'pour $1 \leq \beta \leq n$, $a_{j_\beta} \in C$, $n \in \mathbb{N}$, et ou'pour $1 \leq \gamma \leq p$, $a_{k_\gamma} \in C$, $p \in \mathbb{N}$, et ou'pour $1 \leq \delta \leq q$, $a_{l_\delta} \in C$, $q \in \mathbb{N}$, mais alors

$$a_{i_1}.....a_{i_m}wa_{k_1}.....a_{k_p} = a_{j_1}.....a_{jn}a_{k_1}.....a_{k_p} = a_{i_1}.....a_{i_m}a_{l_1}.....a_{lq}$$

et ceci entraine que C n'est pas un code. ■

Définition 2.1.7 ***(Code maximal):*** *Un code C sur A est dit* ***maximal*** *s'il n'est pas strictement inclus dans un autre code sur A,*

$$C \subseteq C' \Rightarrow C = C'.$$

Exemple 2.1.3 *Voici quelques exemples de codes maximaux:*

$C_1 = \{aa, \ ab, \ bb, \ ba\}$ *un code maximal fini .*

$C_2 = ba^*$ *est un code maximal infini.*

$C_1 = \{a, \ ba\}$ *est un code mais n'est pas maximal (cela reste un code si on lui ajout bb).*

2.2 Le codage de Huffman

David Huffman a proposé en 1952 une méthode statistique qui permet d'attribuer un mot de code binaire aux différents symboles à compresser (pixels ou caractères par exemple). La

longueur de chaque mot de code n'est pas identique pour tous les symboles : les symboles les plus fréquents (qui apparaissent le plus souvent) sont codés avec de petits mots de code, tandis que les symboles les plus rares reçoivent de plus longs codes binaires. On parle de codage à longueur variable préfixé pour désigner ce type de codage car aucun code n'est le préfixe d'un autre. Ainsi, la suite finale de mots codés à longueurs variables sera en moyenne plus petite qu'avec un codage de taille constante.

Définition 2.2.1

Le codeur de Huffman crée un arbre ordonné à partir de tous les symboles et de leur fréquence d'apparition. Les branches sont construites récursivement en partant des symboles les moins fréquents.

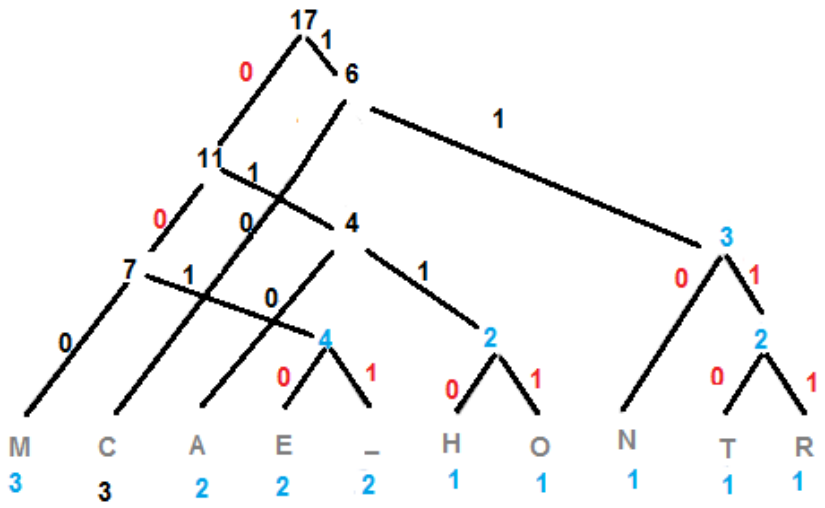
La construction de l'arbre se fait en ordonnant dans un premier temps les symboles par fréquence d'apparition. Successivement les deux symboles de plus faible fréquence d'apparition sont retirés de la liste et rattachés à un nœud dont le poids vaut la somme des fréquences des deux symboles. Le symbole de plus faible poids est affecté à la branche 1, l'autre à la branche 0 et ainsi de suite en considérant chaque nœud formé comme un nouveau symbole, jusqu'à obtenir un seul nœud parent appelé racine. Le code de chaque symbole correspond à la suite des codes le long du chemin allant de ce caractère à la racine. Ainsi, plus le symbole est "profond" dans l'arbre, plus le mot de code sera long.

Exemple 2.2.1

Soit la phrase suivante : "COMMENT_ CA_ MARCHE". Voici les fréquences d'apparitions des lettres

M	*C*	*A*	*E*	_	*H*	*O*	*N*	*T*	*R*
3	3	2	2	2	1	1	1	1	1

Voici l'arbre correspondant:



Les codes correspondants à chaque caractéres sont tels que les codes caractéres les plus fréquents sont courts et ceux correspondant aux symboles les moins fréquents sont longs:

M	C	A	E	$\_$	H	O	N	T	R
000	10	010	0010	0011	0110	0111	110	1110	1111

.

2.3 Algorithme de reconnaissance des codes

Reconnaitre si un ensemble donné est un code n'est pas toujours chose facile, mais il existe un algorithme simple qui permet de le décider. Les deux propositions qui suivent sont la preuve de correction et de terminaison de cet algorithme.

Proposition 2.3.1 *Soit $C \subset A^*$. On définit*

$$\begin{cases} U_1 = C^{-1}C \setminus \{\varepsilon\} \quad \textit{et par récurrence} \\ \forall n \geq 1 \\ U_{n+1} = C^{-1}U_n \cup U_n^{-1}C. \end{cases}$$

On a alors:

$$C \textit{ est un code } \Leftrightarrow \forall n \geq 1 \ \varepsilon \notin U_n$$

La démonstration nécessite le lemme suivant :

Lemme 2.3.1 *Soit* $C \subset A^+.\forall n \geq 1\ \forall k \in \{1.....n\}$ *on a*

$$\begin{aligned} \varepsilon &\in U_n \Leftrightarrow \exists u \in U_k\ \exists i,\ j \in \mathbb{N}^2 uC^i \cap C^j \neq \phi \\ \text{avec } i+j+k &= n \end{aligned}$$

Démonstration. On prouve le lemme à n fixé par récurrence descendante sur k.

Si $k = n$, on a évidemment $i = j = 0$. Si $\varepsilon \in U_n$ on pose $u = \varepsilon$ et on a bien $uC^0 \cap C^0 = \{\varepsilon\}$. Réciproquement si on a

$$u \in U_n \text{ tel que } uC^0 \cap C^0 = \{u\} \cap \{\varepsilon\} \text{ alors } u = \varepsilon \text{ et donc } \varepsilon \in U_n.$$

Soit $1 \leq k < n$, supposons la propriété vérifiée pour $k+1$. Si $\varepsilon \in U_n$ par hypothèse de récurrence, $\exists v \in U_{k+1,}\ i,\ j \in \mathbb{N}^2$ tels que $i+j+k+1 = n$ et $\exists x,\ y \in C^i \times C^{\ j}$tels que $vx = y \in vC^i \cap C^{\ j}$. Comme $U_{k+1} = C^{-1}U_k \cup U_k^{-1}C$, on a

$z,\ u \in C \times U_k$ tel que soit $zv = u$ soit $z = uv$.

Dans le premier cas, on a $ux = zvx = zy$ comme $z,\ y \in C \times C^{\ j}$, $zy \in C^{\ j+1}$ et $uC^i \cap C^{\ j+1} \neq \phi$. Dans le deuxième cas

$uy = uvx = zx \in C^{i+1}$ donc $uC^j \cap C^{\ i+1} \neq \phi$, avec à chaque fois $u \in U_k$.

Réciproquement, soient $w \in uC^i \cap C^{\ j}$ où $i+j+k = n$. Si $j = 0$, alors l'intersection est vide à moins d'avoir $u = \varepsilon$ et $i = 0$ car $\varepsilon \notin C$, on a alors $k = n$ mais on a supposé $k < n$ donc $j \geq 1$. On a donc $v,\ x,\ v' \in C^{\ i} \times C \times C^{\ j-1}$ tels que

$uv = xv'$. On distinque ensuite 2 cas suivant les longueurs comparées de u et x.

Si $\mid u \mid \leq \mid x \mid$ alors $\exists u' \in A^*\ uu' = x$ et alors $u' \in U_k^{-1}C \subset U_{k+1}$. De plus $v = u'v'$ donc $u'C^{\ j-1} \cap C^{\ i} \neq \phi$ et par hypothèse de récurrence $\varepsilon \in U_n$.

Sinon

$$\exists x' \in A^+ u = xx' \text{ avec } x' \in C^{-1}\ U_k \subset U_{k+1} \text{ et } x'v = v' \in x'C^i \cap C^{j-1}\ .$$

D'après l'hypothèse de récurrence $\varepsilon \in U_n$. ■

Démonstration. Supposons que C ne soit pas un code. Soit w de longueur minimale tel que w ait deux factorisations dans C. On a donc $n, m \in (\mathbb{N})$ et $(x_i), (x'_j) \in C$ tels que

$$w = x_1x_2.....x_n = x'_1x'_2.....x'_m \text{ avec } x_1 \neq x'_1.$$

On peut supposer sans perdre de généralité que $\mid x_1 \mid < \mid x_1^{'} \mid$ car w est de longueur minimale. On a alors

$\exists u \in A^+ x_1 u = x_1^{'}$ avec $u \in C^{-1}C \setminus \{\varepsilon\} = U_1$ et $uC^{m-1} \cap C^{n-1} \neq \phi$ et donc $\varepsilon \in U_{n+m-1}$ d'après le lemme (2.2.1).

Réciproquement, si $\varepsilon \in U_n$ pour un certain n, on applique le lemme avec

$$k = 1.\ \exists u \in U_1, i, j \in \mathbb{N}^2 \text{ et } v, w \in C^i \times C^j \text{ tel que } uC^i \cap C^j \neq \phi.$$

De plus comme $U_1 = C^{-1}C \setminus \{\varepsilon\}$ $\exists x, y \in C^2$ tel que $xu = y$ avec $x \neq y$ car $u \neq \varepsilon$. d'où $yC^i \cap xC^j = xuC^i \cap xC^{\,j} \neq \phi$ ce que fait que C ne peut être un code. ■

Exemple 2.3.1

▶ *Pour* $C = \{b, abb, abbba, bbba, baabb\}$*, nous obtenons*

$$U_1 = \{ba, bba, aabb\}, \quad X^{-1}U_1 = \{a, ba\}, \qquad U_1^{-1}X = \{a, ba\}$$
$$U_2 = \{a, ba, abb\}, \qquad X^{-1}U_2 = \{a, \varepsilon\}, \quad U_2^{-1}X = \{bb, bbba, abb, \varepsilon, ba\}.$$

Ainsi $\varepsilon \in U_3$*, donc* C *n'est pas un code.*

▶ *Soit* $C = \{a, ab, ba\}$ *et* $A = \{a, b\}$*. Nous avons*

$$U_1 = \{b\} \quad U_2 = \{a\} \quad U_3 = \{\varepsilon, b\} \quad U_4 = C \quad U_5 = U_3\ .$$

L'ensemble U_3 *contient le mot vide. Donc* C *n'est pas un code.*

▶ *Soit* $C = \{aa, ba, bb, baa, bba\}$ *et* $A = \{a, b\}$*.nous obtenons*

$$U_1 = \{a\}, \quad U_2 = U_1, \quad \textit{ainsi } U_n = \{a\} \forall n \geq 1.$$

Donc C *est un code.*

Proposition 2.3.2

Si C *est rationnel l'ensemble de ses* U_n *est fini.*

Preuve. On rappelle qu'un langage est rationnel est équivalent au fait que le nombre de ses quotients à gauche est fini. On montre par récurrence sur n que les U_n sont des unions finies de quotients à gauche de C auxquelles on peut avoir retiré ε.

$$\text{Pour } n = 1,\ U_1 = C^{-1}C \setminus \{\varepsilon\} = (\cup_{x \in C} x^{-1}C) \setminus \{\varepsilon\}\,.$$

Supposons que c'est vrai au rang $n-1$, alors comme le quotients à gauche d'un quotient à gauche de C est un quotient à gauche de $C^{-1}U_n$ est bien une union de quotients à gauche de C (l'absence ou la présence de ε ne change pas grand chose juste de nombreuses disjonctions de cas si on veut rentrer dans les détails). De plus $(U_n\ )^{-1}\ C$ est bien sur une union de quotient à gauche de C.

Comme le nombre de quotients à gauche de C est fini leurs unions sont aussi en nombre fini (retirer ε fine fait que doubler ce nombre au pire). ■

Exemple 2.3.2

1. *Soit $A = \{a, b\}$ et $C = ba^*$. alors C est un code reconnaissable de suffixe . En effet ,$U_1 = a^+$ et $U_2 = \emptyset$. Ainsi l'ordre (U_n) à deux éléments distincts.*

2. *Soit $A = \{a,\ b\}$, considérons $C = \{aa,\ ba, bb, baa, bba\}$. C n'est pas préfix et considérer des factorisations dans C n'est pas vraiment envisageable. Les U_n permettent pourtant de conclure très vite. En effet $U_1 = \{a\} = U_2$.*

2.4 Mesure d'un code

Cette partie introduit une mesure sur les parties de A^*, et certaines conséquences, quand ces ensembles sont des codes, quant à la mesure qu'ils peuvent avoir.

Définition 2.4.1 ***(Distribution de Bernoulli):***

Soit A un alphabet, une ***distribution de Bernoulli*** *sur A est un morphisme : $\pi : A \to \mathbb{R}_+$ (où $\mathbb{R}_+$ est considéré comme un monoïde multiplicatif) tel que :*

$$\sum_{a \in A} \pi(a) = 1.$$

Une distribution est dite positive si : $\forall a \in A\ \ \pi(a) > 0$.

Proposition 2.4.1 *Pour tout $n \geq 1$:*

$$\sum_{u \in A^n} \pi(u) = 1.$$

Démonstration. Par récurrence sur n

Pour $u \in A^n$, on a $\sum_{a \in A} \pi(ua) = \pi(u) \sum_{a \in A} \pi(a) = \pi(u)$, et donc

$$\sum_{u \in A^{n+1}} \pi(u) = \sum_{v \in A^n} \sum_{a \in A} \pi(va) = \sum_{v \in A^n} \pi(v) = 1.$$

■

Définition 2.4.2 ***(Mesure d'une partie)***

On étend π à $\mathcal{P}(A^*)$ (les parties de A^*) en posant pour tout $\mathcal{L} \subset A^*$:

$$\pi(\mathcal{L}) = \sum_{l \in \mathcal{L}} \pi(l).$$

Exemple 2.4.1

► *Soit $A = \{a, b\}$ et $C = \{b, ab, ba\}$. Définir π par $\pi(a) = 1/3$, $\pi(b) = 2/3$. Alors*

$$\pi(C) = \frac{2}{3} + \frac{2}{9} + \frac{2}{9} = \frac{10}{9}.$$

Alors C n'est pas un code.

Proposition 2.4.2

$\pi : \mathcal{P}(A^*) \to \mathbb{R}_+$ à les proposition immédiates suivantes. Les trois premières propriétés en font une mesure sur $\mathcal{P}(A^*)$:

i $\forall \mathcal{L} \subset A^* \pi(\mathcal{L}) \geq 0.$

ii $\pi(\phi) = 0.$

iii *Pour toute famille $(\mathcal{L}_i)_{i \in I}$ de sous-ensembles de A^* deux à deux disjoints :*

$$\pi(\underset{i \in I}{\cup} \mathcal{L}_i) = \sum_{i \in I} \pi(\mathcal{L}_i).$$

iv $\pi(A^*) = \pi(\underset{n \in \mathbb{N}}{\cup} A^n) = \sum_{n \in \mathbb{N}} \pi(A^n) = \infty.$

v *Si les* $(\mathcal{L}_i)_{i\in I}$ *ne sont pas deux à deux disjoints:*

$$\pi(\underset{i\in I}{\cup}\mathcal{L}_i) \leq \sum_{i\in I}\pi(\mathcal{L}_i).$$

vi *Soit* $\mathcal{L} \subset A^*$, *on pose pour* $n \in \mathbb{N}$ $S_n = \pi(\{w \in \mathcal{L} \setminus \ \mid w \mid \geq n\})$. *On a alors comme* π *est une mesure :*

$$\pi(\mathcal{L}) = \sup_{n\geq 0} S_n.$$

vii *Soient* $\mathcal{L}$ *et* M *doux langages sur* A, *comme* $\mathcal{L}M = \underset{l\in L}{\cup}\ \underset{m\in M}{\cup} lm$ *:*

$$\pi(\mathcal{L}M) \leq \sum_{l\in\mathcal{L}}\sum_{m\in M}\pi(lm) = \sum_{l\in\mathcal{L}}\pi(l)\sum_{m\in M}\pi(m) = \pi(l)\pi(m).$$

viii *On en déduit, pour tout* $X \subset A^*$ *:*

$$\pi(X^*) \leq \sum_{n\geq 0}\pi(X^n) \leq \sum_{n\geq 0}\pi(X)^n.$$

Proposition 2.4.3 *Soit* C *un code alors :*

$$\begin{aligned} \forall n \ &\geq \ 1\ \pi(C^n) = \pi(C)^n \\ \pi(C^*) \ &= \ \sum_{n\geq 0}\pi(C)^n. \end{aligned}$$

En particulier $\pi(C^*) < \infty \Leftrightarrow \pi(C) < 1$. *Réciproquement, si* π *est positive, si* $\pi(C^*) < \infty$ *et* $\forall n \geq 1\ \pi(C^n) = \pi(C)^n$ *alors* C *est un code.*

Démonstration. Supposons que C est un code.

On notera $C^{(n)} = C \times C \times ..... \times C$ le produit cartésien de C par C, n fois. Comme C est un code, la fonction

$$\begin{array}{rccc} \psi \ : & C^{(n)} & \rightarrow & C^n \\ & \underline{x} = (x_1, ..., x_n) & \mapsto & x_1.....x_n \end{array}$$

est bijective (son image est C^n par définition et son injectivité découle directement de la définition d'un code).

On en déduit que :

$$\begin{aligned}\pi(C)^n &= (\sum_{x\in C}\pi(x))^n = \sum_{(x_1,..,x_n)\in C^{(n)}}\pi(x_1).....\pi(x_n) = \sum_{\underline{x}\in C^{(n)}}\pi(\psi(\underline{x}))\\ &= \sum_{x\in C^n}\pi(x) = \pi(C^n)\end{aligned}$$

Le changement de numérotation dans la somme découle de la bijectivité de ψ .

De plus les ensembles C^nsont disjoints car C est un code donc:

$$\pi(C^*) = \pi(\cup_{n\geq 0}C^n) = \sum_{n\geq 0}\pi(C^n) = \sum_{n\geq 0}\pi(C)^n.$$

L'équivalence $\pi(C^*) < \infty \Leftrightarrow \pi(C) < 1$ est une conséquence directe du fait que $\pi(C^*)$ est une somme géométrique de raison $\pi(C)$.

Supposons maintenant que C n'est pas un code mais qu'on a bien les hypothèses de la réciproque. Il existe donc un mot $u \in C^+$ qui a deux factorisations dans C :

$$u = x_1.....x_n = x_1^{'}.....x_m^{'}.$$

Le mot

$$uu = x_1^{'}.....x_m^{'}x_1.....x_n = x_1.....x_nx_1.....x_n$$

à alors 2 factorisations en $m+n$ mots de C. D'où

$$\pi(C)^k = \sum_{\substack{\underline{x}\in C^{(n+m)}\\ \psi(\underline{x})\neq uu}}\pi(\psi(\underline{x}) + 2\pi(uu) \geq \pi(C^{m+n}) + \pi(uu)$$

Comme $\pi(C^*) < \infty$ et $\pi(C^{m+n}) = \pi(C)^{m+n}$ on a $\pi(uu) < 0$ ce qui contredit la positivité de π . ■

Proposition 2.4.4 *Soit C un code sur A. S'il existe une distribution de Bernoulli positive sur A^* telle que $\pi(C) = 1$ alors C est maximal*

Démonstration. Supposons que C n'est pas maximal. Il existe alors $y \in C$ tel que $Y = C \cup \{y\}$ est un code. D'après la proposition (2.4.3), on a $\pi(Y) \leq 1$. De plus

$$\pi(Y) = \pi(C) + \pi(y) = 1 + \pi(y).$$

Donc $\pi(y) = 0$ ce qui contredit π positive. ■

Exemple 2.4.2 *On va montrer que* $C_1 = \cup_{n\geq 0} a^n b A^n$ *est maximal. On pose* $\pi(a) = p < 1 et$ *donc* $\pi(b) = 1 - p$.

$$\pi(C_1) = \sum_{n\geq 0} \pi(a^n b A^n) = \sum_{n\geq 0} p^n (1-p)\pi(A^n) = (1-p)\sum_{n\geq 0} p^n = 1$$

Donc $\pi(C_1) = 1$ *pour toute distribution de Bernoulli positive, on a donc une hypothèse plus forte que nécessaire dans la proposition* (2.4.4). *On en déduit que* $C_1 est$ *un code maximal.*

2.5 Codes complets

Définition 2.5.1 ***(Eléments complétables)***.

Soient M *un monoïde et* P *un sous-ensemble de* M. *Un élément* m *de* M *est dit complétable dans* P *si*

$$\exists u,\ v \in M^2\ umv \in P.$$

On notera $F(P) = M^{-1}PM^{-1}$ *l'ensemble des mots complétables dans* P.

Définition 2.5.2 ***(Ensembles denses et complets et maigres)***.

$P \subset M$ *est dense dans* M *si* $M = F(P)$. *Si* P *n'est pas dense, on dit que* P *est maigre.*

$P \subset M$ *est complet dans* M *si le monoïde généré par* P *est dense. Pour les codes cela se traduit par :*

le code C *sur* A *est complet si* C^* *est dense dans* A^* .

Exemple 2.5.1

i *Soit* $A = \{a\}$, *Les sous ensembles denses de* A^* *sont les sous ensembles infinis.*

ii *Tout sous ensembles non vide de* a^+ *est complet, puisqu'il produit d'un sous monoïde infinis.*

Définition 2.5.3 ***(Mots sans bords)***.

Un mot $w \in A$ *est sans bords si aucun facteur gauche propre de* w *et aussi un facteur droit propre. En d'autres termes :*

$$\forall u \in A^*\ w \in (uA^* \cap A^*u) \Rightarrow (u = \varepsilon\ ou\ u = w).$$

Exemple 2.5.2

$w = abb$ est un mot sans bord tandis que $w = abab$ n'en es pas un.

Lemme 2.5.1 *Soient $C \subset A^*$ un code, $y \in A^*$ un mot sans bords tel que $y \notin F(C)$. Alors l'ensemble $Y = C \cup \{y\}$ est un code.*

Démonstration. On suppose que Y n'est pas un code sur A. Soit w le plus petit mot qui a deux factorisations dans Y on a alors n, $m \in \mathbb{N}$ et $(y_i)_{i=1.....n}$, $(y_j^{'})_{j=1.....m} \in Y$ tels que

$$w = y_1.....y_n = y_1^{'}.....y_m^{'}.$$

Si

$$y \notin \{y_i\}_{i=1.....n} \cup \left\{(y_j^{'})\right\}_{j=1.....m}$$

alors c'est une factorisation dans C qui est un code donc elles sont identiques ce qui est impossible.

Si

$$y \notin \{y_i\}_{i=1.....n} ety \in \left\{y_j^{'}\right\}_{j=1....m}$$

(ou vice versa) alors $y \in F(C)$ ce qui est impossible.

Enfin si

$$y \in \{y_i\}_{i=.....n} \cap \left\{(y_j^{'})\right\}_{j=1.....m}$$

soient i_0 et $j_0$les plus petits indices tels que y apparaît dans $\{y_i\}$et $\left\{y_{j}\right\}$. On ne peut pas avoir

$$y_1.....y_{i_0-1} = y_1^{'}.....y_{j_0-1}^{'}.$$

Sinon cela contredirait la minimalité de w. On peut supposer $y_1.....y_{i_0-1} < y_1^{'}.....y_{j_0-1}^{'}$.Si

$$y_1.....y_{i_0-1} < y_1^{'}.....y_{j_0-1}^{'} \text{ alors } y \in F(C)$$

ce qui contredit les hypothèses. Sinon les deux occurrences de y ont des lettres en commun (mais pas toutes) ce qui contredit le fait que y est sans bords. ■

Lemme 2.5.2 ***(Complétion en un mot sans bords).***

Soit A un alphabet contenant au moins deux lettres.

$$\forall u \in A^+ \exists v \in A^* tel\ que\ uv\ est\ sans\ bords.$$

Proposition 2.5.1

Tout code maximal est complet.

Démonstration. Si $\mid A \mid = 1$ alors on vérifie facilement que les seuls codes sont les a^n pour $n \in \mathbb{N}$ qui sont complets et ϕ qui n'est pas maximal. Sinon soient $C \in A^+$ un code qui n'est pas complet et $u \notin F(C)$. D'après le lemme (2.5.2) (on a bien$\mid A \mid \geq 2$) $\exists v \in A^*$ tel que uv soit sans bords. Comme u est un facteur de uv on a toujours $uv \notin F(C)$. On déduit du lemme (2.4.1) que $C \cup \{y\}$ est un code. Donc C n'est pas maximal. ■

Conclusion générale

Dans ce mémoire, on a pu découverir des notions mathématiques nouvelles: les codes de longueurs variàbles et leurs utilisation dans la compression des données.

Bibliographie

[1] G´ERAUDS'ENIZERGUES. Informatique théorique 2, notes de cours. Année 2010-2011.

[2] JEAN BERSTEL. Automates et grammaires.Université de Marne-la-Vallée.Année 2004-2005.

[3] JEAN BERSTEL, DOMINIQUE PERRIN. Theory of Codes. Université Paris VI, université Paris VII. Année 1985.

[4] JEAN BERSTEL, DOMINIQUE PERRIN, CHRISTOPHE REUTENAUER. CODES AND AUTOMATA, Cambridge University Press. Année 2010.

[5] MICHAL MARCHAND. mathématiques pour l'informaticien. Année 1997.

[6] MICHEL RIGO. Théorie des automates et langages formels, Université de Liège. Année 2009-2010.

[7] NACER GHADBANE. Etude sur les groupes syntaxiques de petits degrés, Memoire de Magistère,Université de M'sila, Année 2010.

[8] PIERRE BERLIOUX, MNACHO ECHENIM ET MICHEL LÉVY. Théorie des langages.Année 2009.

[9] TONY BOURDIER. Mathématiques Discrètes 1& Informatique Théorique, Université Henri Poincaré. Année 2007-2008