Théorie du langage et Automates

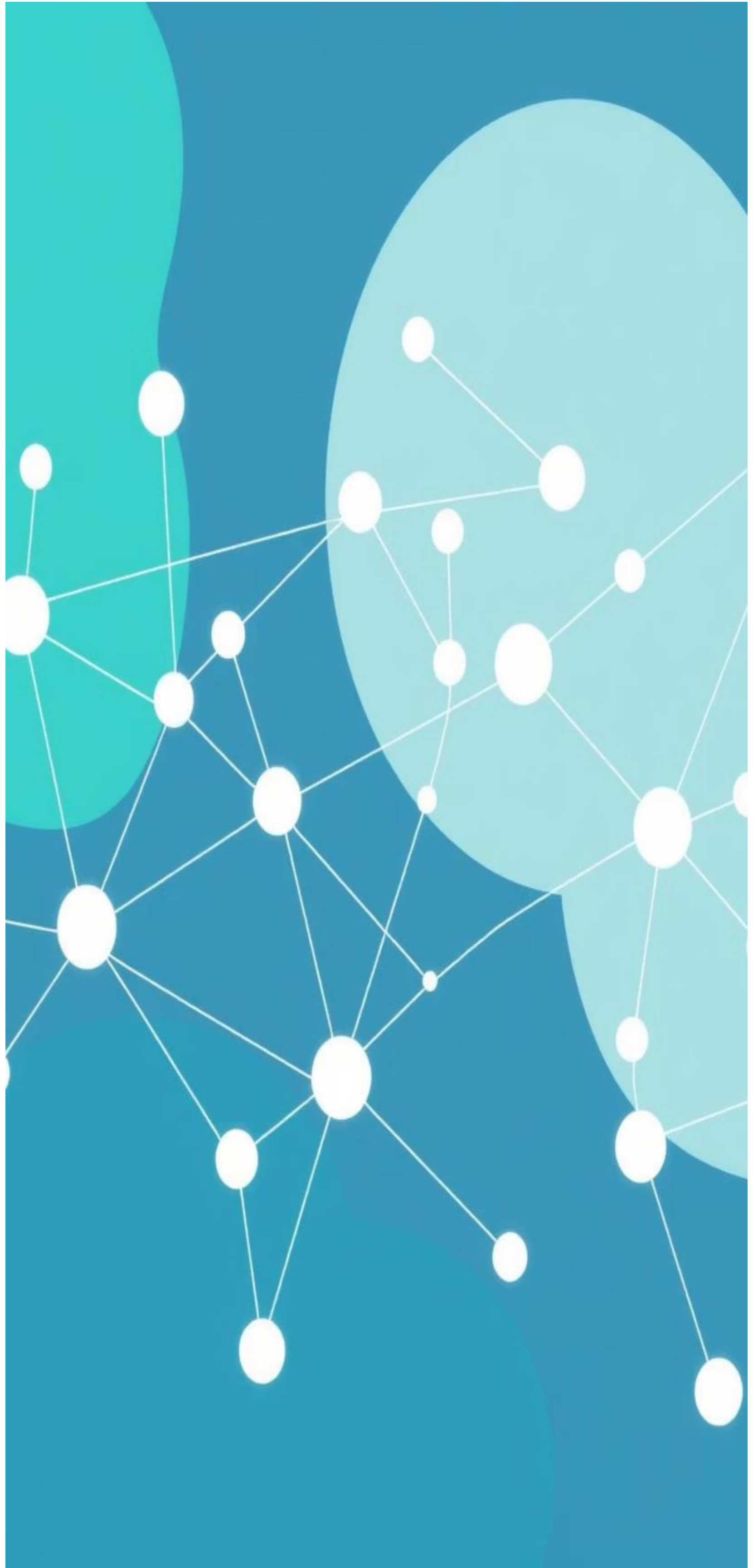
2L GLSI

H. Merdassi

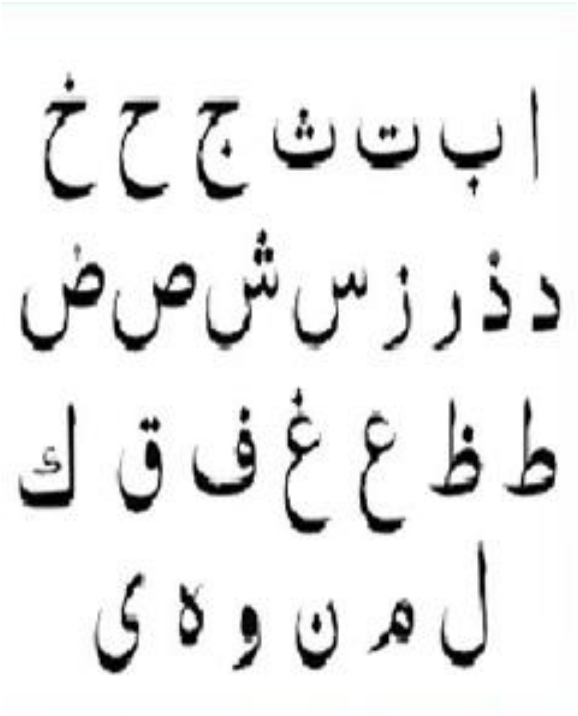
Plan de cours

- Alphabets, Mots, Langages et Grammaires
- Langages réguliers et Automates finis
- Langages hors-contexte et Automates à pile

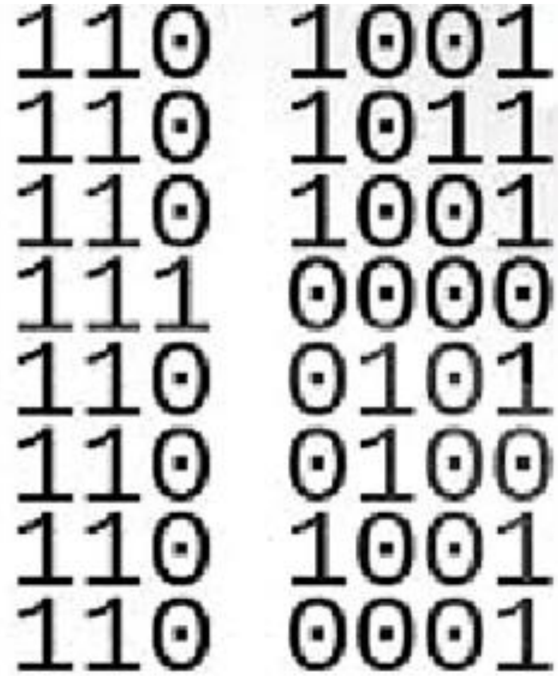
Alphabets
Mots
Langages
Grammaires



Alphabet


ا ب ت ث ج ح خ
د ذ ر ز س ش ص ض
ط ظ ع غ ف ق ك
ل م ن و ه ی


A B C D E F
G H I J K L
M N O P Q
R S T U V
W X Y Z


110 1001
110 1011
110 1001
111 0000
110 0101
110 0100
110 1001
110 0001

Alphabet

abstract	continue	for	new	switch
assert	default	goto	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Alphabet du langage JAVA

Alphabet

Définition

Un alphabet ,noté X, est un ensemble fini non vide des éléments appelés symboles.

Remarque

Un symbole est une entité abstraite qui ne peut pas être définie formellement comme les lettres, les chiffre....

Exemple

- X = {0,1}, l'alphabet binaire.
- A= {a,b,c,....,A,B,...é,ç...}, l'alphabet de la langue française
- B= {int, if, else, printf}, un alphabet du langage C.
- C={bonjour,ça,va,10,$} est un alphabet de 5 symboles.

Mot

Définition

Un mot sur un alphabet X est une suite finie et totalement ordonnées composées d'éléments de X.

Exemples

- 00, 11, 01001, ce sont des mots sur l'alphabet X= {0,1}
- bonjour est un mot sur l'alphabet de la langue française et un symbole sur l'alphabet X={bonjour,ça,va,10,$}. Donc, bonjourçava10 est un mot de 4 symboles sur ce dernier alphabet.
- L'ensemble des mots construits sur L'alphabet C du langage C est {tous les programmes (corrects, incorrects) écrits en C}.

Mot

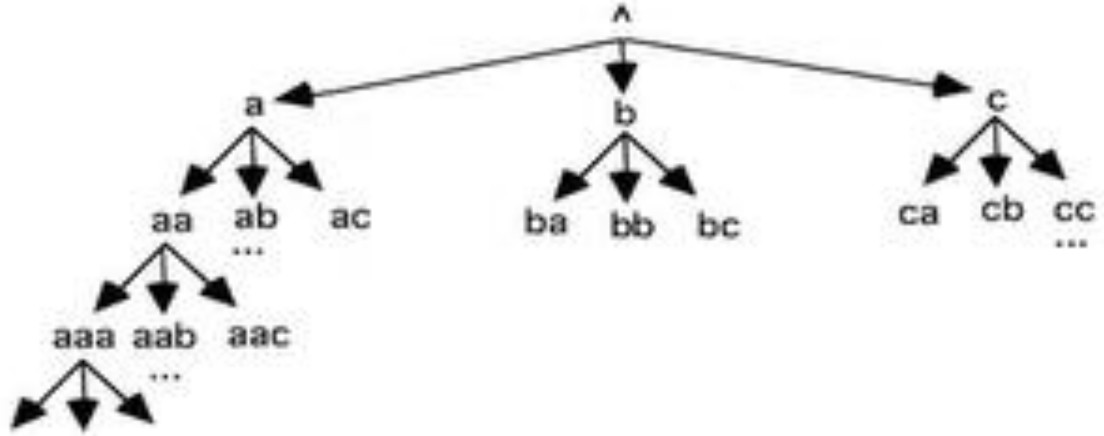
Notations

- Le mot vide, noté par ε, est le mot qui ne contient aucun élément d'un alphabet X.
- L'ensemble des mots construits sur un alphabet X est un ensemble infini noté X^*
- L'ensemble des mots construits sur un alphabet X qui ne contient pas le mot vide est un ensemble infini noté X^+ . On écrit :

$$X^*=X^+ \cup \{\varepsilon\}$$

Mot

Exemple



Soit l'alphabet X= {a, b, c}, donc X* définit sur X est :

X*={ε, a, b, c, aa, bb, cc, ac, ab, ba, bc, ca, cb, aba, abb,.....}

={ε} U{a, b, c} U{aa, ab, bb,ba,cc,..}U{aaa, bbb, cccc,...}U...U{}

={ε}+{a, b, c} +{aa, ab, bb,ba,cc,..}+{aaa, bbb, cccc,...}+...+{}

Si on note: X^i l'ensemble des mots de longueur i construits sur l'alphabet X, donc on a:

$$X^* = X^0 + \underbrace{X^1 + X^2 + \dots\dots\dots\dots + X^{+\infty}}_{X^+} = \sum_{i=0}^{\infty} X^i$$

- **Remarque:** Le symbole **+** représente l'union dans la théorie des langages.

Mot

Longueur d'un mot

Définition:

Soient X un alphabet, $w \in X^*$ et x un des symboles constituant le mot x :

- $|w|$ est la longueur du mot w.
- $|w|_x$ est le nombre de l'occurrence de x dans dans le mot w.

Exemple

$X=\{a, b, c\}$, $|abcbaa|=6$, $|abcbaa|_a=3$

Mot

Concaténation

Soient deux mots w, w′ ∊ X*, la concaténation de w et w′ est définie comme la juxtaposition de w et w′. Elle est notée par w.w′ on bien ww′.

Exemple

X= {0,1}, w=10, w′=00

w.w′=1000 ≠ w′w=0010

Mot

Puissance d'un mot

Soit un alphabet X , w ∊ X* et n ∊ IN , la puissance de W est donnée comme suit :

$$w^n = \begin{cases} \varepsilon & \text{si } n = 0 \\ w & \text{si } n = 1 \\ ww^{n-1} = w^{n-1}w & \text{si } n > 1 \end{cases}$$

Exemple

Soit X= {a, b} et w = aba

- $w^0=\varepsilon$
- $w^1=w.\ \varepsilon = w=aba$
- $w^2=ww^1=ww=abaaba$
- $w^3=ww^2=www=$ abaabaaba

Mot

Factorisation d'un mot

Soit un alphabet X et w, u $\in X^*$:

- u est un facteur gauche (préfixe) de w $\Leftrightarrow \exists v \in X^*$ tel que w =uv
- u est un facteur droit (suffixe) de w $\Leftrightarrow \exists v \in X^*$ tel que w =vu
- u est un préfixe propre de w $\Leftrightarrow \exists v \in X^+$ tel que w = uv
- u est un suffixe propre de w $\Leftrightarrow \exists v \in X^+$ tel que w = vu

Mot

Factorisation d'un mot

Exemple:

Soit l'alphabet X= {a, b}, et le mot w = babb :

- Les préfixes de w sont, b, ba, bab, babb
- Les suffixes de w sont, b, bb, abb, babb
- Les préfixes propres de w sont: b, ba , bab
- Les suffixes propres de w sont: b , bb , abb

Mot

Inverse d'un mot ou Miroir

Le miroir d'un mot $w= a_1a_2\ldots\ldots a_n$ est le mot noté $w^R= a_n \ldots\ldots a_2a_1$ obtenu en inversant les symboles de w.

Exemple

Soit w=abbc un mot de l'alphabet x={a, b}, le mot de w est w^R=cbba.

Remarque

Le miroir de n'importe quel mot composé d'un seul symbole est le mot lui-même.

- Le miroir de mot vide est lui-même $\varepsilon^R=\varepsilon$.
- Un mot est un palindrome si $w^R = w$, ex :$(aba)^R$=aba.

Langage

❑ Définition

Un langage sur un alphabet X est une partie de X*. Donc un langage est un ensemble de mots. On note $L \subseteq X^*$ ou $L \in P(X^*)$.

❑ Exemple

- Soit l'alphabet X={a, b}
- $\varnothing$ est le langage vide, il ne contient aucun mot.
- {ε} est un langage.
- {a, b, aa, bb, aba} est un langage
- $\{w \in X^* \ / \ w= a^n$ tel que $n > 0\}$ est un langage

❑ Remarque

- Un langage sur un alphabet X peut être fini ou infini
- $\varnothing$ est un langage défini sur n'importe quel alphabet.

Langage

Opérations sur les langages

Les langages étant des ensembles, toutes les opérations ensemblistes « classiques » leur sont donc applicables.

Soient $L_1, L_2 \subseteq X^*$: $L_1 \cup L_2 = \{w \in X^* \ / \ w \in L_1 \text{ ou } w \in L_2\}$

❑ **Union**

L'union de L_1 et L_2 est l'ensemble des mots de L_1 et L_2 :
Elle est :

- Associative.
- Commutative
- Elément neutre , le langage vide $\varnothing$: $L \cup \varnothing = \varnothing \cup L = L$
- Elément absorbant, le vocabulaire X^* : $L \cup X^* = X^* \cup L = X^*$
- Notée + dans la théorie des langages. On écrit aussi : $L_1 + L_2 = \{w \in X^* \ / \ w \in L_1 \text{ ou } w \in L_2\}$

Langage

Opérations sur les langages

❑ Intersection

L'intersection de L1 et L2 est l'ensemble des mots qui appartiennent à la fois à L1 et L2 :

$$L_1 \cap L_2 = L_2 \cap L_1 = \{w \in X^* \;/\; w \in L1 \text{ et } w \in L2\}$$

Elle est :

- Associative
- Commutative
- Son élément neutre est X^*
- Son élément absorbant est $\varnothing$ (puisque " $\forall L : \varnothing \cap L = L \cap \varnothing = \varnothing$)

Langage

Opérations sur les langages

❑ Produit ou concaténation

Le produit des langages appelé aussi la concaténation des langages est le résultat de la concaténation de chaque mot de L_1 avec chaque mot de L_2 (Ce n'est pas le produit cartésien) :

Elle est :

$$L_1 . L_2 = \{u . v \ / \ u \in L_1 \text{ et } v \in L_2\}$$

Le produit est :

- Associatif
- Il n'est pas commutatif
- Son élément neutre est $\{\varepsilon\}$
- Son élément absorbant est $\emptyset$

Langage

Opérations sur les langages

❑ **Théorème**

Le produit de langages est distributif par rapport à l'union

$\forall L_1, L_2, L_3 \subseteq X^*$

$L_1.(L_2 \cup L_3) = (L_1.L_2) \cup (L_1.L_3)$

$(L_2 \cup L_3).L_1 = (L_2.L_1) \cup (L_3.L_1)$

De manière générale, $\forall A, L_i \subseteq X^*$

$$A.(\bigcup_{i=1}^{\infty} L_i) = \bigcup_{i=1}^{\infty} (A.L_i)$$

$$(\bigcup_{i=1}^{\infty} L_i).A = \bigcup_{i=1}^{\infty} (L_i.A)$$

Attention ! : Le produit de langages n'est pas distributif par rapport à l'intersection.

Langage

Opérations sur les langages

- **Puissance**
  La puissance ou l'itération d'un langage est définit comme suit :

$$L^n = \begin{cases} \{\varepsilon\} & \text{si } n = 0 \\ L & \text{si } n = 1 \\ LL^{n\text{-}1} = L^{n\text{-}1}L & \text{si } n > 1 \end{cases}$$

Attention ! Ne pas confondre L^n avec le langage contenant les puissances nièmes des mots de L et qui serait défini par $\{u \in X^*, \exists v \in L, u = v^n\}$.

Langage

Opérations sur les langages

- **La fermeture positive**
  La fermeture positive de L noté L^+ et on écrit : $L^+ = \bigcup_{i=1}^{\infty} L^i$

- **La fermeture de Kleene**
  La fermeture étoile de L nommée aussi la fermeture de Kleene et notée L* est définit par :

$$L^* = \bigcup_{i=0}^{\infty} L^i$$

$$L^* = L^0 + L^1 + + L^{\cdot}$$

$$= \{\varepsilon\} + \bigcup_{i=1}^{\infty} L^i = \{\varepsilon\} + L^+$$

- $L^* = L^+ \cup \{\varepsilon\}$ et $L^+ = L.L^*$

Langage

Opérations sur les langages

- **Exemple**

Soit $X= \{0,1\}$, $L_1= \{00, 11, 01\}$ et $L_2= \{01, 10\}$

- $L_1+ L_2= \{00, 11, 01, 10\}$
- $L_1 \cap L_2= \{01\}$
- $L_1.L_2 = \{0001, 0010, 1101, 1110, 0101, 0110\} \neq L_2.L_1= \{0100, 0111, 0101, 1000, 1011, 1001\}$
- $L_2^*= \{\varepsilon\} +\{L_2\} + \{L_2. L_2\} + \{L_2. L_2.L_2\} +\ldots\ldots +\{L_2. L_2.L_2\ldots\ldots L_2. L_2.L_2\}$

Exercices

Soit l'alphabet $A = \{a, b\}$

Etant donnés les mots $u = aa$ et $v = bab$

1. Ecrire les mots uv, $(uv)^2$ et u^3v.
2. Enoncer tous les mots de longueur 2 définis sur A.
3. Soient les ensembles:

$E_1 = \{u.v \ / \ u \in A^+, v \in A^+\}$, $E_2 = \{u.v \ / \ u \in A^+, v \in A^*\}$, $E_3 = \{u.v \ / \ u \in A^*, v \in A^*\}$

Solution

1. $uv = aabab$, $(uv)^2 = uvuv = aababaabab$ et $u^3v = uuuv = aaaaaabab$.
2. Mots de longueur 2 = {aa, ab, ba, bb}
3. $E_1 = \{u.v \ / \ u \in A^+, v \in A^+\} = \{u \in A^* \ / \ |u| \geq 2\}$ = ensemble des mots d'au moins 2 symboles

   $E_2 = \{u.v \ / \ u \in A^+, v \in A^*\} = A^+ = \{u \in A^* \ / \ |u| \geq 1\}$ = ensemble des mots d'au moins 1 symbole

   $E_3 = \{u.v \ / \ u \in A^*, v \in A^*\} = A^+$

Exercice 1

Sur l'alphabet A = {0, 1}, on considère les langages L_1 et L_2 définis par

$$\begin{aligned} \mathcal{L}_1 &= \{01^n / n \in \mathbb{N}\} \\ \mathcal{L}_2 &= \{0^n 1 / n \in \mathbb{N}\} \end{aligned}$$

Définir les langages $\mathcal{L}_1\mathcal{L}_2$, $\mathcal{L}_1 \cap \mathcal{L}_2$ et $\mathcal{L}_1^2$.

Exercice 2

Sur l'alphabet A = {0, 1}, on considère les langages L_1 et L_2 définis par

$$\mathcal{L}_1 = \{01^n / n \in \mathbb{N}\}$$
$$\mathcal{L}_2 = \{0^n1 / n \in \mathbb{N}\}$$

Définir les langages $\mathcal{L}_1\mathcal{L}_2$, $\mathcal{L}_1 \cap \mathcal{L}_2$ et $\mathcal{L}_1^2$.

Correction :

- $\mathcal{L}_1\mathcal{L}_2 = \{01^n0^m1 / n \in \mathbb{N}, m \in \mathbb{N}\}$
- $\mathcal{L}_1 \cap \mathcal{L}_2 = \{01\}$
- $\mathcal{L}_1^2 = \{01^n01^m / n \in \mathbb{N}, m \in \mathbb{N}\}$

Exercice 3

Soient les langages $L_1 = \{a, ab, ba\}$, $L_2 = \{\varepsilon, b, ab\}$ et $L_3 = \{a^n.b^n \,/\, n \geq 0\}$.

Définir les langages suivants :

a) $L_1.L_2$; b) $L_2.L_1$; c) $L_1.L_3$; d) $L_1.\{\varepsilon\}$; e) $\{\varepsilon\}.L_1$; f) $L_1.\emptyset$;

g) $\emptyset.L_1$; h) $L_1.L_1$; i) $L_2.L_2$; j) $L_3.L_3$.

Grammaire

- Un langage peut être décrit par un certain nombre de règles.
- Le but était de donner une description précise des règles permettant de construire les phrases correctes d'une langue.

Exemple de la grammaire française :

❑ Le vocabulaire est défini par l'ensemble :
T = { *le, la, fille, jouet, regarde* }
❑ Les catégories syntaxiques sont :
la phrase, notée ***PH***
le groupe nominal, noté ***GN***
le verbe, noté ***V***
le déterminant, noté ***D***
le nom, noté ***N***

❑ Les règles : combiner des éléments du vocabulaire et des catégories syntaxiques pour construire des catégories syntaxiques :

PH → GN V GN
GN → D N
D → le
D → la
N → fille
N → jouet
V → regarde

où le symbole → est une abréviation de "peut être composé de".
❑ la catégorie syntaxique de départ est la phrase PH.

Grammaire

La phrase "**la fille regarde le jouet**" est une phrase correcte pour la grammaire envisagée, comme le montre l'analyse suivante :

PH → ***GN V GN***

→ ***D N V GN***

→ *la* ***N V GN***

→ *la fille* ***V GN***

→ *la fille regarde* ***GN***

→ *la fille regarde* ***D N***

→ *la fille regarde le* ***N***

→ *la fille regarde le jouet*

« *le fille regarde la jouet* »

« *le jouet regarde la fille* »

- **Phrases syntaxiquement correctes**

Définition (Grammaire)

Une grammaire est un quadruplet ***G = (T, N, S, R)***

- ***T*** est le vocabulaire terminal (l'alphabet sur lequel est défini le langage)

- ***N*** est le vocabulaire non terminal (l'ensemble des symboles qui n'apparaissent pas dans les mots générés, mais qui sont utilisés au cours de la génération). Un symbole non terminal désigne une "catégorie syntaxique".

- ***R*** est un ensemble de règles dites de réécriture ou de production de la forme :

$$u1 \rightarrow u2, \text{ avec } u1 \in (N \cup T)^{+} \text{ et } u2 \in (N \cup T)^{*}$$

➢ S ∊ N est le symbole de départ ou axiome. C'est à partir de ce symbole non terminal que l'on commencera la génération de mots au moyen des règles de la grammaire.

Terminologie

- Une suite de symboles terminaux et non terminaux (un élément de (N U T)*) est appelée **une forme**.

- Une règle u1 → u telle que u ∊ T^* est appelée **une règle terminale**.

Notation

- Lorsque plusieurs règles de grammaire ont une même forme en partie gauche, on pourra "**factoriser**" ces différentes règles en séparant les parties droites par **des traits verticaux**.
- Par exemple:

l'ensemble de règles **S →ab, S →aSb, S →c** pourra s'écrire **S → ab|aSb|c**.

- Le langage défini, ou généré, par une grammaire est l'ensemble des mots qui peuvent être obtenus à partir du symbole de départ par application des règles de la grammaire.
- Plus formellement, on introduit les notions de dérivation entre formes.

- Enfin, on **définit le langage généré par une grammaire comme étant l'ensemble des mots pouvant être dérivés depuis l'axiome.**

Dérivation

- **Dérivation simple** :

  Si $x \rightarrow w$ est une règle d'une grammaire, alors en remplaçant x par w dans une expression composée de terminaux et de non-terminaux on obtient une dérivation

  Exemple : Mot **E*(E)**, et la règle **E→ E+E** alors E*(E) → E*(E+E) est une **dérivation**

- **Enchaînement de dérivations**

  Si $u_0 \rightarrow u_1 \ldots \rightarrow u_n$ on écrit : $u_0 \xrightarrow{*} u_n$

Langage généré par une grammaire

Le langage généré par une grammaire ***G = (T, N, S, R)*** est l'ensemble des mots sur ***T*** qui peuvent être dérivés à partir de ***S***

$$\mathcal{L}(G) = \{v \in T^* / S \overset{+}{\Rightarrow} v\}$$

Remarques :

- Une grammaire définit un seul langage.
- Par contre, un même langage peut être engendré par plusieurs grammaires différentes.

Langage généré par une grammaire

Exercice : On considère la grammaire $G = (T, N, Ph, R)$ où

$$
\begin{aligned}
T = & \; \{ \quad un, une, le, la, enfant, garcon, fille, cerise, haricot, cueille, mange \; \} \\
N = & \; \{ \quad Ph, Gn, Gv, Df, Dm, Nf, Nm, V \; \} \\
R = & \; \{ \quad Ph \to Gn\ Gv \\
& \quad\quad Gn \to Df\ Nf \mid Dm\ Nm \\
& \quad\quad Gv \to V\ Gn \\
& \quad\quad Df \to une \mid la \\
& \quad\quad Dm \to un \mid le \\
& \quad\quad Nf \to fille \mid cerise \\
& \quad\quad Nm \to enfant \mid garcon \mid haricot \\
& \quad\quad V \to cueille \mid mange \; \}
\end{aligned}
$$

- La phrase *"une cerise cueille un enfant"* appartient-elle au langage $\mathcal{L}(G)$?
- Déterminer le nombre de phrases du langage décrit par G.

Langage généré par une grammaire

- Pour montrer qu'une phrase appartient au langage, on construit une dérivation de l'axiome ***Ph*** jusqu'à la phrase.

- On souligne à chaque fois le symbole non terminal qui est remplacé par la dérivation.

$$
\begin{aligned}
\underline{Ph} &\Rightarrow \underline{Gn}\ Gv \Rightarrow Df\ Nf\ \underline{Gv} \Rightarrow Df\ Nf\ V\ \underline{Gn} \Rightarrow \underline{Df}\ Nf\ V\ Dm\ Nm \\
&\Rightarrow \text{une}\ \underline{Nf}\ V\ Dm\ Nm \Rightarrow \text{une cerise}\ \underline{V}\ Dm\ Nm \Rightarrow \text{une cerise cueille}\ \underline{Dm}\ Nm \\
&\Rightarrow \text{une cerise cueille un}\ \underline{Nm} \Rightarrow \text{une cerise cueille un enfant}
\end{aligned}
$$

Langage généré par une grammaire

Exercice : On considère la grammaire $G = (T, N, S, R)$ où

$$
\begin{array}{rl}
T = & \{ \quad b, c \ \} \\
N = & \{ \quad S \ \} \\
R = & \{ \quad S \to bS \ \mid \ cc \ \}
\end{array}
$$

Déterminer $\mathcal{L}(G)$.

Correction : $\mathcal{L}(G) = \{b^n cc / n \in \mathbb{N}\}$

En effet, partant de l'axiome S, toute dérivation commencera nécessairement par appliquer 0, 1 ou plusieurs fois la première règle puis se terminera en appliquant la deuxième règle. On représentera cela en écrivant le schéma de dérivation suivant :

$$S \xrightarrow{n \text{ fois } (1)} b^n S \xrightarrow{(2)} b^n cc \quad \text{avec } n \in \mathbb{N}$$

Langage généré par une grammaire

Exercice : On considère la grammaire $G = (T, N, S, R)$ où

$$\begin{aligned} T &= \{ \quad 0, 1 \quad \} \\ N &= \{ \quad S \quad \} \\ R &= \{ \quad S \to 0S \mid 1S \mid 0 \quad \} \end{aligned}$$

Déterminer $\mathcal{L}(G)$.

Correction : $\mathcal{L}(G) = \{u0/u \in \{0,1\}^*\}$

En effet, partant de l'axiome S, toute dérivation commencera nécessairement par appliquer 0, 1 ou plusieurs fois la première ou la deuxième règle puis se terminera en appliquant la troisième règle. On représentera cela en écrivant le schéma de dérivation suivant :

$$S \xRightarrow{n \text{ fois } (1 \text{ ou } 2)} uS \xRightarrow{(3)} u0 \quad \text{avec } n \in \mathbb{N}, u \in \{0,1\}^* \text{ et } |u| = n$$

Langage généré par une grammaire

Exercice : Construire une grammaire pour le langage $\mathcal{L} = \{ab^n a / n \in \mathbb{N}\}$.

Correction : On définit la grammaire $G = (T, N, S, R)$ où

$$\begin{array}{rl} T = & \{ \quad a, b \ \} \\ N = & \{ \quad S, U \ \} \\ R = & \{ \quad S \rightarrow aUa \\ & \quad\ \ U \rightarrow bU \ \mid \ \epsilon \ \} \end{array}$$

Langage généré par une grammaire

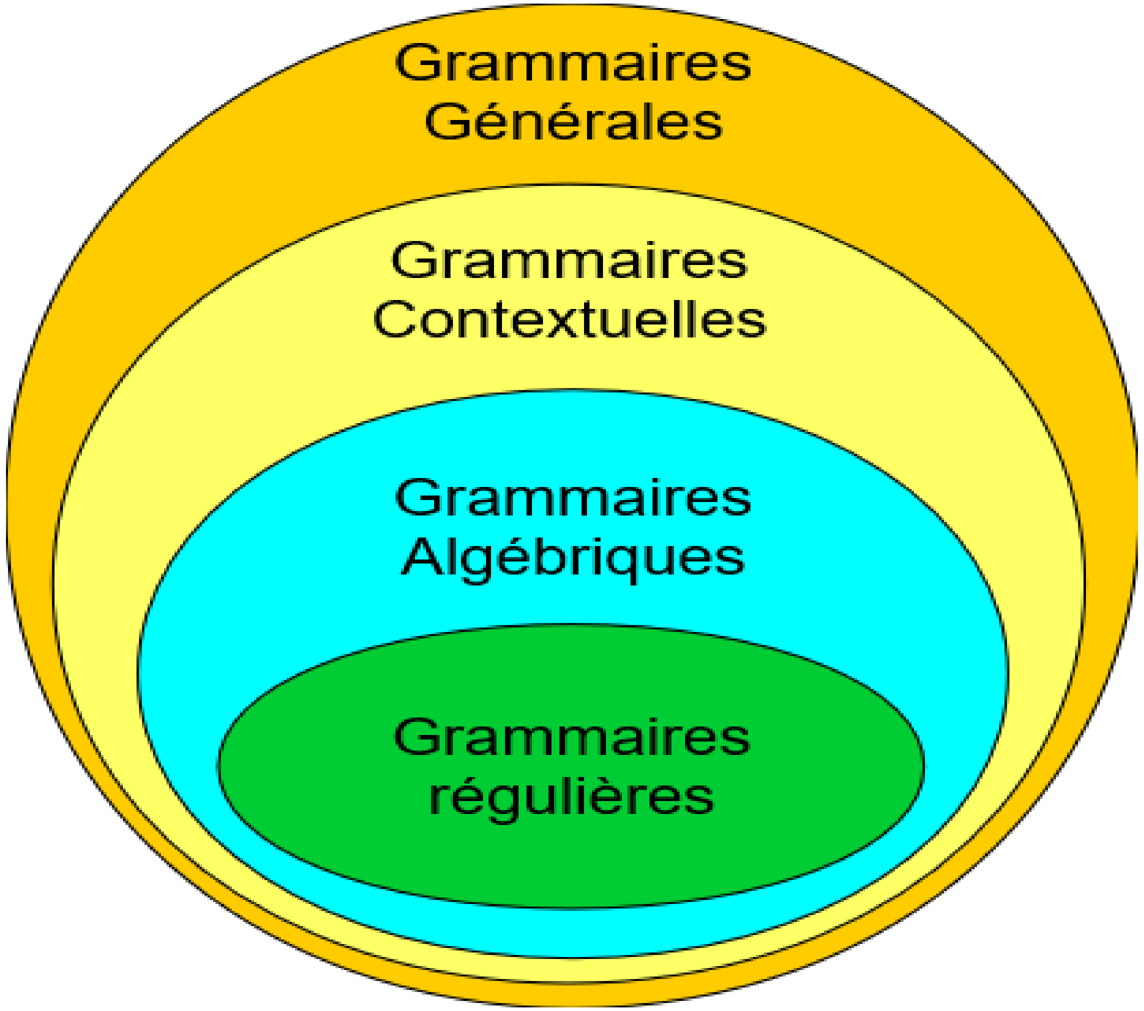
Exercice : Construire une grammaire pour le langage $\mathcal{L} = \{0^{2n}1^n / n \geq 0\}$.

Correction : On définit la grammaire $G = (T, N, S, R)$ où

$$\begin{array}{rl} T = & \{ \quad 0, 1 \ \} \\ N = & \{ \quad S \ \} \\ R = & \{ \quad S \to 00S1 \ \mid \ \epsilon \ \} \end{array}$$

Types de grammaires

En introduisant des critères plus ou moins restrictifs sur la forme des règles de grammaire, on obtient des classes de grammaires hiérarchisées, ordonnées par inclusion. La classification des grammaires, définie en 1957 par Noam CHOMSKY, distingue les quatre classes suivantes :



Types de grammaires

1

Type 0 : Pas de restriction sur les règles

2

Type 1 : Grammaires sensibles au contexte ou **contextuelles**.
Les règles de R sont de la forme :

$$uAv \to uwv \text{ avec } A \in N,\ u, v \in (N \cup T)^* \text{ et } w \in (N \cup T)^+$$

Autrement dit, le symbole non terminal ***A*** est remplacé par w si on a les contextes ***u*** à gauche et ***v*** à droite.

3

Type 2: Grammaires hors-contexte. Les règles de R sont de la forme :

$$A \to w \text{ avec } A \in N \text{ et } w \in (N \cup T)^*$$

Autrement dit, le membre de gauche de chaque règle est constitué d'un seul symbole non terminal.

4

Type 3 : Grammaires régulières
<u>à droite</u>. Les règles de R sont de la forme $A \to aB$ ou $A \to a$ avec $A, B \in N$ et $a \in T$
<u>à gauche</u>. Les règles de R sont de la forme $A \to Ba$ ou $A \to a$ avec $A, B \in N$ et $a \in T$

Types de grammaires

Autrement dit:

- le membre de gauche de chaque règle est constitué d'un seul symbole non terminal
- le membre de droite est constitué d'un symbole terminal et éventuellement d'un symbole non terminal.
- Pour les grammaires régulières à droite, le symbole non terminal doit toujours se trouver à droite du symbole terminal
- Pour les grammaires régulières à gauche il doit se trouver à gauche.

4

Type 3 : Grammaires régulières

<u>à droite</u>. Les règles de R sont de la forme $A \to aB$ ou $A \to a$ avec$A, B \in N$ et $a \in T$

<u>à gauche</u>. Les règles de R sont de la forme $A \to Ba$ ou $A \to a$ avec$A, B \in N$ et $a \in T$

Types de grammaires

A chaque type de grammaire est associé un type de langage :

- les grammaires de type 3 génèrent les langages réguliers,
- les grammaires de type 2 génèrent les langages hors-contexte,
- les grammaires de type 1 génèrent les langages contextuels,
- les grammaires de type 0 permettent de générer tous les langages “décidables”, autrement dit, tous les langages qui peuvent être reconnus en un temps fini par une machine. Les langages qui ne peuvent pas être générés par une grammaire de type 0 sont dits “indécidables”.

Types de grammaires

A chaque type de grammaire est associé un type d'automate qui permet de reconnaître les langages de sa classe (c'est-à-dire de déterminer si un mot donné appartient au langage) :

- **les langages réguliers** sont reconnus par des **automates finis**
- **les langages hors-contexte** sont reconnus par des **automates à pile**
- **les langages contextuelles** sont reconnus par des **automate linéairement borné**
- **les autres langages**, décrits par des grammaires de type 0, sont reconnus par des machines de Turing. Ainsi, la machine de Turing peut être considérée comme le modèle de machine le plus puissant qu'il soit, dans la mesure où tout langage (ou plus généralement, tout problème) qui ne peut pas être traité par une machine de Turing, ne pourra pas être traité par une autre machine.

Types de grammaires

Exercice : On considère la grammaire $G = (T, N, S, R)$ où

$$\begin{array}{rlll} T = & \{ & a, b, c, d & \} \\ N = & \{ & S, U & \} \\ R = & \{ & S \to aU \mid c & \\ & & U \to Sb \mid d & \} \end{array}$$

Donner le type de G et déterminer $\mathcal{L}(G)$.

Correction :

- G est hors-contexte (car la partie gauche de chaque règle est un symbole non terminal).
- G n'est pas régulière car la règle (1) est régulière à droite tandis que la règle (3) est régulière à gauche.
- Notons qu'il n'existe pas de grammaire régulière permettant de générer *L(G)*.

Types de grammaires

Exercice : On considère la grammaire $G = (T, N, S, R)$ où

$$\begin{array}{rll} T = & \{ & a, b, c, d \ \} \\ N = & \{ & S, U \ \} \\ R = & \{ & S \to aU \mid c \\ & & U \to Sb \mid d \ \} \end{array}$$

Donner le type de G et déterminer $\mathcal{L}(G)$.

Correction :

$$\mathcal{L}(G) = \{a^n cb^n, a^{n+1}db^n / n \in \mathbb{N}\}$$

En effet, partant de l'axiome S, toute dérivation partant de S suivra nécessairement un des deux schémas suivants :

$$S \;\; {}^{n}\!\xRightarrow{\text{fois (1 suivie de 3)}} \;\; a^n S b^n \;\; \xRightarrow{(2)} \;\; a^n c b^n$$

$$S \;\; {}^{n}\!\xRightarrow{\text{fois (1 suivie de 3)}} \;\; a^n S b^n \;\; \xRightarrow{(1)} \;\; a^n a U b^n \;\; \xRightarrow{(4)} \;\; a^n a d b^n$$

avec $n \in \mathbb{N}$

Types de grammaires

Exercice : On considère le langage L des mots sur {a, b, c} qui contiennent au moins une fois la chaine bac. Définir formellement *L* et construire une grammaire hors-contexte décrivant *L*.

Correction : $\mathcal{L} = \{u.bac.v/u \in \{a,b,c\}^*, v \in \{a,b,c\}^*\}$

On définit la grammaire hors-contexte $G = (T, N, S, R)$ où

$$\begin{array}{rcl} T = & \{ & a, b, c \ \} \\ N = & \{ & S, U \ \} \\ R = & \{ & S \rightarrow UbacU \\ & & U \rightarrow Ua \mid Ub \mid Uc \mid \epsilon \ \} \end{array}$$

Types de grammaires

Exercice : On considère le langage *L* des mots sur {0, 1} qui représentent des entiers pairs non signés en base 2 (les mots de ce langage se terminent tous par 0 et ne commencent pas par 0, sauf pour l'entier nul). Définir formellement *L* et construire une grammaire régulière décrivant *L*.

Correction: $\mathcal{L} = \{0, 1u0/u \in \{0,1\}^*\}$

On définit la grammaire régulière à droite G = (T,N, S,R) où

$$
\begin{aligned}
T = \quad & \{ \quad a, b, c \ \} \\
N = \quad & \{ \quad S, U \ \} \\
R = \quad & \{ \quad S \to 0 \ | \ 1U \\
& \qquad U \to 1U \ | \ 0U \ | \ 0 \ \}
\end{aligned}
$$

On définit la grammaire régulière à gauche G = (T,N, S,R) où

$$
\begin{aligned}
T = \quad & \{ \quad a, b, c \ \} \\
N = \quad & \{ \quad S, U \ \} \\
R = \quad & \{ \quad S \to 0 \ | \ U0 \\
& \qquad U \to U1 \ | \ U0 \ | \ 1 \ \}
\end{aligned}
$$

Exercice 1

On considère la grammaire G = (T, N, S, R) où
$T = \{ b, c \}$
$N = \{ S \}$
$R = \{ S \rightarrow bS \mid cc \}$

1. Quel est le type de G.
2. Déterminer L(G).

Exercice 1

On considère la grammaire G = (T, N, S, R) où
T = { b, c }
N = { S }
R = { $S \rightarrow bS \mid cc$ }

1. Quel est le type de G.
2. Déterminer L(G).

G est une grammaire régulière à droite.

2. L(G) = $\{b^n cc / n \in N\}$
En effet, partant de l'axiome S, toute dérivation commencera nécessairement par appliquer 0, 1 ou plusieurs fois la première règle puis se terminera en appliquant la deuxième règle. On représentera cela en écrivant le schéma de dérivation suivant :

S=>cc
S=>bS =>bcc
S=>bS=>bbS=>bbcc n fois
S=>bS=>bbS=>bbbS=>bbbbS=>...............=>b^n S=> b^n cc

Exercice 2

Soit la grammaire **G = ({a, b, c} , {S, A} , S , P)**
où P contient les règles suivantes :

$$S \rightarrow aS \mid bA \; ; \; A \rightarrow cA \mid \varepsilon$$

1) Déterminer si les mots w1 = abac, w2 = aabccc, w3 = cabbac et w4 = ab sont générés par G.

2) Trouver le langage généré par G (qu’on note L(G))

Exercice 3

Pour chacun des langages suivants, donner des exemples de mots contenus dans chacun des langages, et des grammaires qui engendrent chacun des langages $(L_i)_{i=1,..,5}$:

1) $L_1 = \{ w \in \{a, b, c\}^* \,/\, w$ commence par la lettre 'a' $\}$;
2) $L_2 = \{ w \in \{a, b, c\}^* \,/\, w$ se termine par la lettre 'a' $\}$;
3) $L_3 = \{ w \in \{a, b, c\}^* \,/\, w$ contient au moins une occurrence de la lettre 'a' $\}$;
4) $L_4 = \{ w \in \{a, b, c\}^* \,/\, w$ contient au moins deux occurrences la lettre 'a' $\}$;
5) $L_5 = \{ w \in \{a, b, c\}^* \,/\, w$ contient au moins deux occurrences consécutives de la lettre 'a' $\}$.