Théorie des Langages – Feuille n° 2
LANGAGES ET GRAMMAIRES
CORRECTION

Exercice 1 Soient les langages L_1, L_2, L_3 et L_4 suivants construits sur l'alphabet $\Sigma = \{a, b, c\}$. Montrer que ces 4 langages sont tous deux à deux différents.

— $L_1 = (a+b)^*ca^*b^*$
— $L_2 = a^*b^*c(a+b)^*$
— $L_3 = (a^* + b^*)ca^*b^*$
— $L_4 = \{(a+b)^n ca^m b^n,\ n, m \in \mathbb{N}\}$

	L_1	L_2	L_3	L_4	
cba	$\notin$	$\in$	$\notin$	$\notin$	$L_2 \neq L_1, L_3, L_4$
abc	$\in$	$\in$	$\notin$	$\notin$	$L_1 \neq L_3, L_4$
bca	$\in$	$\in$	$\in$	$\notin$	$L_3 \neq L_4$

Ces 4 langages sont bien différents.

Exercice 2 - Soient les langages L_1, L_2 et L_3 construits sur l'alphabet $X = \{a, b\}$. On rappelle que $(a+b) = \{a\} \cup \{b\}$.

$$L_1 = \{a^n b(a+b)^n, n \in \mathbb{N}\}$$
$$L_2 = \{(a+b)^n ba^n, n \in \mathbb{N}\}$$
$$L_3 = \{(a+b)^n b(a+b)^n, n \in \mathbb{N}\}$$

1. Montrez que les langages L_1, L_2 et L_3 ne sont pas égaux

	L_1	L_2	L_3
abb	$\in$	$\notin$	$\in$
bba	$\notin$	$\in$	$\in$

Ces trois langages sont bien différents.

2. Soit $L_4 = \{(a+b)^m ba^n, m, n \in \mathbb{N}\}$. Montrez que $L_2 \neq L_4$

$abaa \in L_4$, mais $abaa \notin L_2$.

3. Donnez les grammaires qui engendrent L_2 et L_4

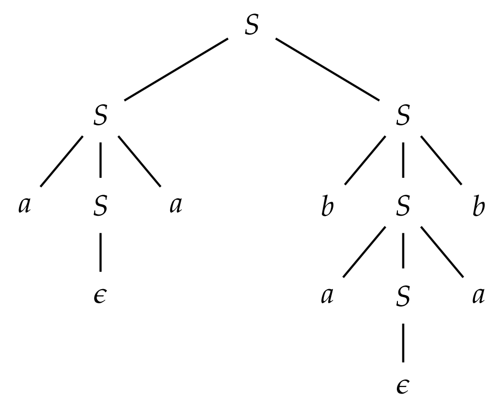
Soit $G_2 = \langle V, \Sigma, P, S\rangle$, telle que $\mathcal{L}(G_2) = L_2$, avec
— $\Sigma = \{a, b\}$
— $V \setminus \Sigma = \{S\}$
— Axiome S
— 3 règles de production : $S \rightarrow aSa|bSa|b$

Soit $G_4 = \langle V, \Sigma, P, S\rangle$, telle que $\mathcal{L}(G_4) = L_4$, avec
— $\Sigma = \{a, b\}$
— $V \setminus \Sigma = \{S\}$
— Axiome S
— 4 règles de production : $S \rightarrow aS|bS|Sa|b$

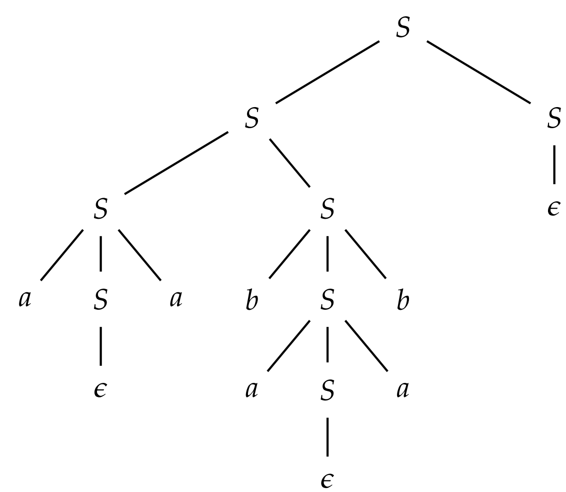
Exercice 3 - Soit la grammaire $G = \langle V, \Sigma, P, S \rangle$, avec $V = \{a, b, S\}$, $\Sigma = \{a, b\}$ et $P = \{S \rightarrow aSa; S \rightarrow bSb; S \rightarrow \epsilon\}$.

1. Soit $G' = \langle V, \Sigma, P', S \rangle$, avec $P' = P \cup \{S \rightarrow SS\}$. Montrez que $aabaab \in \mathcal{L}(G')$. Montrez ensuite que G' est ambigüe.

$w = aabaab \in \mathcal{L}(G')$.



Autre méthode : $S \mapsto SS \mapsto aSaS \mapsto a\epsilon aS \mapsto aabSb \mapsto aabaSab \mapsto aaba\epsilon ab \mapsto aabaab$
G' ambigue : il y a deux arbres de dérivation pour le mot $aabaab$:



2. Quel est le langage engendré par G ? Démontrez

Démontrons que $\mathcal{L}(G)$ est l'ensemble des palindromes de longueur paire sur l'alphabet $\{a, b\}$.

(1). Montrons que tout mot w de $\mathcal{L}(G)$ est un palindrome de longueur paire sur $\{a, b\}$.
D'abord, notons que les mots de $\mathcal{L}(G)$ sont forcément de longueur paire : chaque application d'une règle ajoute soit 2 lettres (2 a ou 2 b), soit aucune lettre (ϵ) au mot w ainsi formé.
Nous allons montrer **(1)** par récurrence sur la taille n de $w \in \mathcal{L}(G)$.
Base : $n = 0$. Si $|w| = 0$, alors $w = \epsilon$, qui est bien un palindrome de longueur paire sur $\{a, b\}$.
Hypothèse de récurrence : On suppose que tout mot de $\mathcal{L}(G)$ de longueur $2n$ est un palindrome de longueur paire.

Soit $w \in \mathcal{L}(G)$ tel que $|w| = 2(n+1)$. Montrons que w est un palindrome.
Comme w est engendré par G, il existe une dérivation pour w qui se termine de la façon suivante :

$$\ldots \mapsto \alpha' S \beta' \mapsto \alpha S \beta \mapsto \alpha \epsilon \beta \mapsto \alpha\beta = w$$

Avec :
- soit $\alpha = \alpha' a$ et $\beta = a\beta'$,
- soit $\alpha = \alpha' b$ et $\beta = b\beta'$

A partir de $\alpha' S \beta'$, il est possible de dériver le mot $\alpha'\beta'$ en appliquant la règle $S \mapsto \epsilon$. Donc, $\alpha'\beta' \in \mathcal{L}(G)$, et $w = \alpha\beta = \begin{cases} \alpha' aa\beta' \\ \text{ou} \\ \alpha' bb\beta' \end{cases}$
Comme $|w| = 2n+2$, $|\alpha'\beta'| = 2n$. D'après l'hypothèse de récurrence, $\alpha'\beta'$ est un palindrome de longueur paire. w est donc également un palindrome de longueur paire.

(2) Montrons que tout palindrome de longueur paire sur $\{a, b\}$ appartient à $\mathcal{L}(G)$.
Soit w un palindrome de longueur paire sur $\{a, b\}$. w est donc de la forme : $x_1x_2x_3 \ldots x_nx_n \ldots x_3x_2x_1$, avec $\forall i, x_i \in \{a, b\}$.
Numérotons les règles de la grammaire G comme suit :
- (a) $S \to aSa$
- (b) $S \to bSb$
- (0) $S \to \epsilon$

Par exemple, la séquence $(a)(a)(b)(0)$ permet de dériver le mot $aabbaa$.
La séquence $(x_1)(x_2)(x_3)\ldots(x_n)(0)$ permet de dériver le mot w. w appartient donc bien à $\mathcal{L}(G)$

3. Pourquoi G n'est pas ambigüe ?

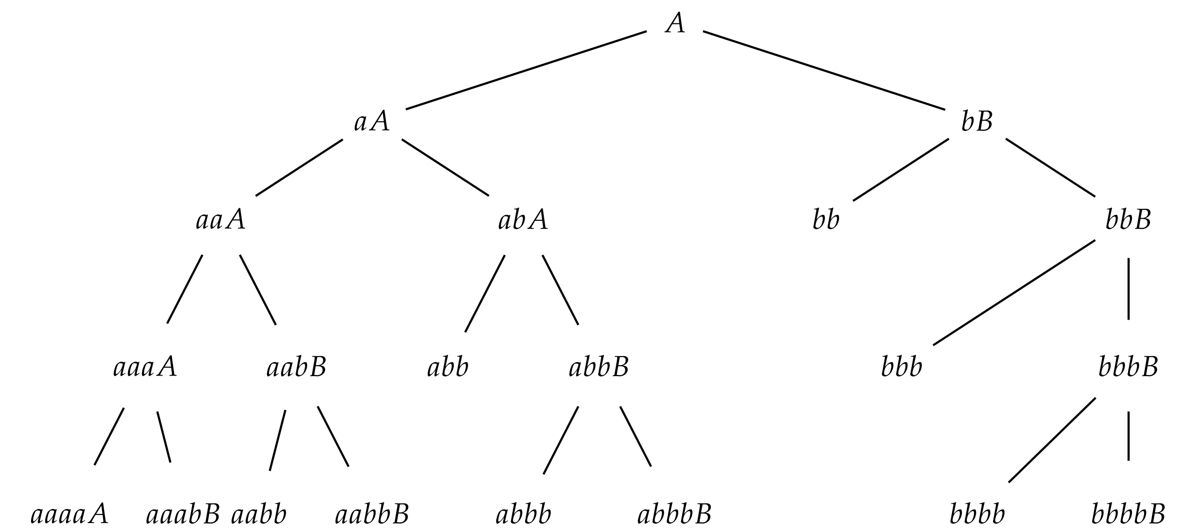
Soit $w = x_1x_2x_3 \ldots x_nx_n \ldots x_3x_2x_1$, avec $\forall i, x_i \in \{a, b\}$, un palindrome de longueur paire sur $\{a, b\}$. Il faut appliquer les règles $(x_1)(x_2)(x_3)\ldots(x_n)(0)$, dans cet ordre, pour obtenir w.
Il n'y a qu'une seule dérivation possible pour obtenir w. G n'est donc pas ambigüe.

Exercice 4 - Soit la grammaire $G = \langle V, \Sigma, P, A\rangle$, avec $V = \{a, b, A, B\}$, $\Sigma = \{a, b\}$ et $P = \{A \rightarrow aA|bB; B \rightarrow b|bB\}$.

1. De quel type est la grammaire G ?

   > G est une grammaire régulière car elle est uniquement composée de règles régulières à droite.

2. Construisez l'arbre de dérivation de profondeur 4 de G.

   > Format un peu différent du cours mais plus lisible pour "lister" les mots générés par une grammaire.

3. A l'aide des règles de production et des mots dérivés de l'arbre de dérivation, déduisez quel est le langage généré par G (écriture en compréhension) ?

   > $L(G) = \{a^n b^2 b^m \mid n \in \mathbb{N}, m \in \mathbb{N}, n \leq m\}$

Exercice 5 - Soit la grammaire $G = \langle V, \Sigma, P, S\rangle$, avec $V = \{\text{if,then,else}, a, b, S\}$, $\Sigma = \{\text{if,then,else}, a, b\}$ et $P = \{S \rightarrow \text{if } b \text{ then } S \text{ else } S\,;\ S \rightarrow \text{if } b \text{ then } S\,;\ S \rightarrow a\}$.

1. Démontrez que cette grammaire est ambigüe

   > $w =$ if b then if b then a else a $\in \mathcal{L}(G)$.
   > 2 dérivations possibles :
   >
   > (a) $S \mapsto$ if b then S else $S \mapsto$ if b then if b then S else $S \mapsto$ if b then if b then a else $S \mapsto$ if b then if b then a else a
   >
   > (b) $S \mapsto$ if b then $S \mapsto$ if b then if b then S else $S \mapsto$ if b then if b then a else $S \mapsto$ if b then if b then a else a

2. Proposez une solution pour lever l'ambiguïté

   > Plusieurs solutions possibles. Par exemple, rajouter des parenthèses :
   > $G = \langle V, \Sigma, P, S\rangle$, avec $V = \{\text{if,then,else,(,)}, a, b, S\}$, $\Sigma = \{\text{if,then,else,(,)}, a, b\}$ et $P = \{S \rightarrow \text{if } b \text{ then } (S) \text{ else } S\,;\ S \rightarrow \text{if } b \text{ then } (S)\,;\ S \rightarrow a\}$.

Exercice 6 - Donner, pour chacun des langages suivants, une grammaire qui l'engendre (précisez le type de cette grammaire).

1. $L_1 = \{0^{2n} \mid n \geq 0\}$
2. $L_2 = \{0^n 1^n \mid n \geq 0\}$
3. $L_3 = \{a^n b^{2n} \mid n \geq 0\}$
4. $L_4 = \{a^m b^n a^n b^m \mid n, m \geq 1\}$
5. $L_5 = \{w \in \{a, b\}^+ \mid \ |w| \text{ est un multiple de } 3\ \}$
6. $L_6 = \{0^i 1^j \mid i \neq j, n, m \geq 0\}$

(L_1) Il est engendré par $G_1 = \langle \{0, S\}, \{0\}, P_1, S\rangle$ avec $P_1 : S \to 00S|\epsilon$

(L_2) Il est engendré par $G_2 = \langle \{0, 1, S\}, \{0, 1\}, P_2, S\rangle$ avec $P_2 : S \to 0S1|\epsilon$

(L_3) Il est engendré par $G_3 = \langle \{a, b, S\}, \{a, b\}, P_3, S\rangle$ avec $P_3 : S \to aSbb|\epsilon$

(L_4) Il est engendré par $G_4 = \langle \{a, b, S, A\}, \{a, b\}, P_4, S\rangle$
avec $P_4 : S \to aSb|aAb; A \to bAa|bAb|\epsilon$

(L_5) Il est engendré par $G_5 = \langle \{a, b, S, A\}, \{a, b\}, P_5, S\rangle$
avec $P_5 : S \to AAAS|AAA; A \to a|b$

(L_6) Peut être décomposé comme $L_6 = \{0^i 1^j \mid i > j\} \cup \{0^i 1^j \mid i < j\}$.
Il est engendré par $G_6 = \langle \{0, 1, S, S_0, S_1\}, \{0, 1\}, P_6, S\rangle$
avec $P_6 : S \to S_0|S_1; S_0 \to 0S_01|0S_0|0; S_1 \to 0S_11|S_11|1;$

Exercice 7 - Ecrire une grammaire permettant de générer les identificateurs d'un langage de programmation (comme Python ou C). On considérera qu'un identificateur est valide s'il commence par une lettre alphabétique (uniquement minuscule) qui peut éventuellement être suivie d'une ou plusieurs lettres alphabétiques (minuscule ou majuscule) et/ou chiffres.

Pour générer ces identificateurs on utilisera la grammaire $G_1 = \langle V, \Sigma, P, \langle ID_1\rangle\rangle$ où
$\Sigma = \{A, \ldots, Z, a, \ldots, z, 0, \ldots, 9\}$,
$V \backslash \Sigma = \{\langle ID_1\rangle, \langle ID_2\rangle, \langle ID_3\rangle, \langle Lettre\rangle, \langle LettreMin\rangle, \langle LettreMaj\rangle, \langle Chiffre\rangle\}$ et
P :

$$\begin{aligned}
\langle ID_1\rangle &\to \langle LettreMin\rangle\langle ID_2\rangle \\
\langle ID_2\rangle &\to \langle ID_3\rangle\langle ID_2\rangle \mid \epsilon \\
\langle ID_3\rangle &\to \langle Lettre\rangle \mid \langle Chiffre\rangle \\
\langle Lettre\rangle &\to \langle LettreMin\rangle \mid \langle LettreMaj\rangle \\
\langle LettreMin\rangle &\to a \mid b \mid \ldots \mid z \\
\langle LettreMaj\rangle &\to A \mid B \mid \ldots \mid Z \\
\langle Chiffre\rangle &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9
\end{aligned}$$

Exercice 8

1. Ecrire une grammaire hors-contexte G_{HC} permettant de décrire tous les mots d'un langage sur $\Sigma = \{a, b\}$ qui contiennent les sous-chaînes aa ET bb.

$G_{HC} = \langle V, \Sigma, P, S \rangle$ où $\Sigma = \{a, b\}$, $V \backslash \Sigma = \{S, S_1, S_2, S_3\}$ et
P :

$$\begin{aligned}
S &\rightarrow aS \mid bS \mid aaS_1 \mid bbS_2 \\
S_1 &\rightarrow aS_1 \mid bS_1 \mid bbS_3 \\
S_2 &\rightarrow aS_2 \mid bS_2 \mid aaS_3 \\
S_3 &\rightarrow aS_3 \mid bS_3 \mid \epsilon
\end{aligned}$$

2. En vous basant sur la grammaire hors-contexte définie à la question précédente, construisez une grammaire régulière G_R équivalente.

$G_R = \langle V, \Sigma, P, S \rangle$ où $\Sigma = \{a, b\}$, $V \backslash \Sigma = \{S, S_1, S_2, S_3, A, A', B, B'\}$ et
P :

$$\begin{aligned}
S &\rightarrow aS \mid bS \mid aA \mid bB \\
A &\rightarrow aS_1 \\
S_1 &\rightarrow aS_1 \mid bS_1 \mid bA' \\
A' &\rightarrow bS_3 \\
B &\rightarrow bS_2 \\
S_2 &\rightarrow aS_2 \mid bS_2 \mid aB' \\
B' &\rightarrow aS_3 \\
S_3 &\rightarrow aS_3 \mid bS_3 \mid \epsilon
\end{aligned}$$

G_R est bien un grammaire régulière car toutes ces règles de production sont régulières à droite.