Feuille d'exercices n°13 - Grammaires non contextuelles - partie 1

Notions abordées
- exemple de grammaires non contextuelles
- suite de dérivations, arbre de dérivation
- langage d'une grammaire non contextuelle, preuve par induction
- exemple de langages non contextuels : langage de Dyck, mots de Łukasiewicz

Exercice 1 : Exemples de dérivations

Soit $\Sigma = \{a, b\}$, $\mathcal{V} = \{S, D, T, X\}$, et $\mathcal{G}$ la grammaire non contextuelle de symbole initial S, et contenant les règles de production ci-contre[a].

$$\begin{aligned} S &\to XSX \mid D \\ D &\to aTb \mid bTa \\ T &\to XTX \mid X \mid \varepsilon \\ X &\to a \mid b \end{aligned}$$

Q. 1 Parmi les dérivations suivantes, lesquelles sont vraies ?

$(a)\ T \Rightarrow aba \qquad (b)\ T \overset{\star}{\Rightarrow} aba \qquad (c)\ T \Rightarrow T \qquad (d)\ T \overset{\star}{\Rightarrow} T$

Q. 2 Parmi les dérivations suivantes, lesquelles sont vraies ?

$(a)\ XXX \overset{\star}{\Rightarrow} aba \qquad (c)\ T \overset{\star}{\Rightarrow} XX \qquad (e)\ D \overset{\star}{\Rightarrow} \varepsilon$
$(b)\ X \overset{\star}{\Rightarrow} aba \qquad (d)\ T \overset{\star}{\Rightarrow} XXX$

Q. 3 Donner 3 mots de $\mathcal{L}(\mathcal{G})$. Justifier leur appartenance à $\mathcal{L}(\mathcal{G})$ par une dérivation.

Q. 4 Donner 3 mots de $\Sigma^\star$ qui ne sont pas dans $\mathcal{L}(\mathcal{G})$.

Q. 5 Donner une description en français de $\mathcal{L}(\mathcal{G})$. En première lecture de cet énoncé, on ne demande pas de preuve du résultat avancé.

Exercice 2 : Arbres de dérivations

Soit $\Sigma = \{a, b, c, \ldots, z, +, (,)\}$, $\mathcal{V} = \{E, S, P, F, T\}$, et la grammaire $\mathcal{G}$ de symbole initial E et de règles de production ci-contre.

$$\begin{aligned} S &\to S + P \mid P \\ P &\to PF \mid F \\ F &\to (S) \mid T \\ T &\to a \mid b \mid c \mid \ldots \mid z \end{aligned}$$

Q. 1 Donner un arbre de dérivation de $ab + bc$.

Q. 2 Donner un arbre de dérivation de $b + (a + b)c + a(bc)$.

Q. 3 Que représente cette grammaire, *i.e.* que réprésentent les mots engendrés par $\mathcal{G}$?

[a]. Lorsque plusieurs règles de production ont le même membre gauche, on peut les représenter factorisées en séparant les différents membres droits par $|$. Par exemple, on note $A \to w|w'$ pour les deux règles $A \to w$ et $A \to w'$.

Exercice 3 : Construction de grammaires

Q. 1 Soit $\Sigma = \{a, b\}$. Donner une grammaire non contextuelle $\mathcal{G}_i$ de langage L_i ci-dessous.
- $L_0 = \{w \in \Sigma^\star \mid |w| \geqslant 3\}$;
- $L_1 = \{w \in \Sigma^\star \mid |w|_a \geqslant 3\}$;
- $L_2 = \{w \in \Sigma^+ \mid w$ commence et finit par la même lettre$\}$;
- $L_3 = \emptyset$;

Q. 2 Soit $\Sigma = \{a, b\}$. Donner une grammaire non contextuelle $\mathcal{G}_i$ de langage L_i ci-dessous♣.
- $L_4 = \{w \in \Sigma^\star \mid |w| \equiv 1[2]\}$;
- $L_5 = \{w \in \Sigma^\star \mid |w| \equiv 1[2]$ et sa lettre du milieu est $a\}$;
- $L_6 = \{w \in \Sigma^\star \mid w$ est un palindrome$\}$;
- $L_7 = L_6^C$;

Q. 3 Donner une grammaire non contextuelle $\mathcal{G}_i$ de langage L_i ci-dessous.
- $L_8 = \{a^n b^n \mid n \in \mathbb{N}\}$;
- $L_9 = \{a^n b^m \mid (n, m) \in \mathbb{N}^2,\ m \geqslant n\}$;
- $L_{10} = \{a^n b^m \mid (n, m) \in \mathbb{N}^2,\ m \geqslant 2n\}$;

Q. 4 Soit $\Sigma = \{a, b\}$. Donner une grammaire non contextuelle $\mathcal{G}_i$ de langage L_i ci-dessous.
- $L_{11} = \{w \in \Sigma^\star \mid |w|_a \geqslant |w|_b\}$;
- $L_{12} = L_{11}^C$;

Q. 5 Soit $\Sigma = \{a, b, \#\}$. Donner une grammaire non contextuelle $\mathcal{G}_{13}$ de langage L_{13} ci-dessous.
- $L_{13} = \{w\#x \mid (w, x) \in (\{a, b\}^\star)^2, \overline{w}^R$ est un sous-mot de x où $\overline{w}^R$ désigne le miroir de $w\}$.

Exercice 4 : Langage de Dyck

Le langage des mots de Dyck est le langage des mots sur $\Sigma = \{(,)\}$ bien parenthésés, c'est-à-dire le plus petit langage $L \subseteq \Sigma^\star$ tel que :
- $\varepsilon \in L$;
- $\forall u \in L,\ (u) \in L$;
- $\forall (u, v) \in L \times L,\ u \cdot v \in L$.

Q. 1 Donner une grammaire non contextuelle $\mathcal{G}$ reconnaissant le langage de Dyck.

Q. 2 Montrer que les mots u de $\mathcal{L}(\mathcal{G})$ sont tels que :
- $|u|_(= |u|_)$
- pour tout préfixe v de u, $|v|_(\geqslant |v|_)$

Q. 3 Montrer que les mots $u \in \Sigma^\star$ vérifiant ces deux conditions sont bien parenthésés.

Q. 4 En déduire une fonction OCaml testant si un mot est dans le langage de Dyck.

Q. 5 Démontrer que ce langage n'est pas régulier.

♣. Lorsque la définition d'un langage fait intervenir le complémentaire, celui-ci s'entend par rapport à $\Sigma^\star$.

Exercice 5 : Raisonner par induction sur une grammaire

Soit $\Sigma = \{a, b\}$, $\mathcal{V} = \{S\}$ et la grammaire $\mathcal{G}$ contenant les règles de production :

$$S \rightarrow aS \mid Sb \mid a \mid b \mid \varepsilon$$

Q. 1 Montrer que ba n'est sous-mot d'aucun mot de $\mathcal{L}(\mathcal{G})$.

Q. 2 Donner $\mathcal{L}(\mathcal{G})$.

Q. 3 Donner une grammaire reconnaissant $\mathcal{L}(\mathcal{G})$ et assurant que chaque mot est obtenu par une unique dérivation depuis le symbole initial.

Exercice 6 : Ambiguïté

Soit $\Sigma = \{a, b\}$, $\mathcal{V} = \{S\}$, et la grammaire $\mathcal{G}$ de symbole initial S et de règles de production :

$$S \rightarrow aS \mid aSbS \mid \varepsilon$$

Q. 1 Montrer que $\mathcal{G}$ est ambiguë.

Q. 2 Montrer que $\mathcal{L}(G) = \{v \in \Sigma^\star \mid \text{pour tout } u \text{ préfixe de } v, |u|_a \geqslant |u|_b\}$.

Q. 3 Donner une grammaire non ambiguë reconnaissant $\mathcal{L}(G)$.

Exercice 7 : Mots de Łukasiewicz [Centrale 2008]

On considère l'ensemble de deux symboles $\Sigma = \{\square, \bigcirc\}$, Σ contient donc uniquement le symbole "carré" : $\square$ et le symbole "rond" : $\bigcirc$. On s'intéresse dans cet exercice à un sous-ensemble de $\Sigma^\star$, nommé **mots de Łukasiewicz**. On rappelle qu'étant donné un mot $w = w_1 w_2 \dots w_n \in \Sigma^\star$, la notation $|w|$ désigne la longueur de w, n ici. On considère la fonction $\mathsf{va} : \Sigma \rightarrow \{-1, 1\}$ telle que $\mathsf{va}(\square) = -1$ et $\mathsf{va}(\bigcirc) = 1$. À un mot $w = w_1 w_2 \dots w_n \in \Sigma^\star$ et à un entier $i \in [\![0, n]\!]$ on associe la valeur : $\overline{w}^i = \sum_{k=1}^{i} \mathsf{va}(w_i)$, en prenant comme convention que $\sum_{k=1}^{0} \mathsf{va}(w_i) = 0$. Par exemple : $\overline{\bigcirc\bigcirc\square\square}^4 = \mathsf{va}(\bigcirc) + \mathsf{va}(\bigcirc) + \mathsf{va}(\square) + \mathsf{va}(\square) = 1 + 1 - 1 - 1 = 0$. et $\overline{\varepsilon}^0 = 0$.
On définit alors l'ensemble des **mots de Łukasiewicz** comme étant l'ensemble des mots $w \in \Sigma^\star$ tels que

$$\forall i \in [\![0, |w| - 1]\!], \overline{w}^i \geqslant 0$$
$$\text{et } \overline{w}^{|w|} = -1$$

Par exemple : $w = \bigcirc\bigcirc\square\square\square \in \mathcal{L}$, en effet les valeurs des $\overline{w}^i$ sont :

i	0	1	2	3	4	5
$\overline{w}^i$	0	1	2	1	0	-1

Q. 1 Donner l'ensemble des éléments de $\mathcal{L}$ de longueur 1, 2, 3.

Q. 2 Donner (en justifiant votre résultat) l'ensemble des éléments de $\mathcal{L}$ de longueur paire.

On suppose définis en OCaml les types et fonctions ci-contre. Le type `sym` est utilisé pour représenter l'ensemble Σ : le symbole $\bigcirc$ est représenté par le constructeur `R` (pour "rond") et $\square$ est représenté par le constructeur `C` (pour "carré"). Le type `word` est utilisé pour représenter les mots de $\Sigma^\star$. La fonction `va` implémente la fonction va définie ci-dessus.

```
type sym = | R | C
type word = sym list
let va (s: sym) : int =
  match s with
   | R ->  1
   | C -> -1
```

Q. 3 L'ensemble des mots de Łukasiewicz est-il régulier ?

Q. 4 Définir une fonction `est_luka : word -> bool` permettant de tester si un mot est de Łukasiewicz, *i.e.* permettant de tester, pour $w \in \Sigma^\star$, si $w \in \mathcal{L}$. On proposera une implémentation ayant une complexité pire cas en $\mathcal{O}(n)$.

Q. 5 On appelle préfixe strict d'un mot $w \in \Sigma^\star$, un mot $u \in \Sigma^\star$ tel qu'il existe un mot $v \in \Sigma^\star \setminus \{\varepsilon\}$ tel que $w = u{\cdot}v$. Montrer qu'un préfixe strict d'un mot de $\mathcal{L}$ n'est pas un mot de $\mathcal{L}$.

Q. 6 Montrer que si $u \in \mathcal{L}$ et $v \in \mathcal{L}$ alors $\circ \cdot u \cdot v \in \mathcal{L}$.

Q. 7 Montrer que si $w \in \mathcal{L}$ et $|w| \neq 1$ alors il existe un unique couple $(u, v) \in \mathcal{L} \times \mathcal{L}$ tel que $w = \circ \cdot u \cdot v$.

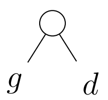
Q. 8 Montrer qu'il existe une grammaire contextuelle non ambiguë reconnaissant l'ensemble des mots de Łukasiewicz.

Q. 9 Donner une fonction `decompose: word -> word * word` prenant en argument un mot de $\mathcal{L}$ de taille différente de 1 (et de 0 car $\varepsilon \notin \mathcal{L}$) et calculant la décomposition de la **Q. 7**.

Exemples :
```
decompose [R; R; C; C; C] = ([R; C; C], [C])
decompose [R; R; C; C; R; C; C] = ([R; C; C], [R; C; C])
```

On définit maintenant l'ensemble des arbres binaires $\mathbb{B}$ **sans étiquette** par les deux règles de construction :

```
type btree =
  | N of btree * btree          (* Nœud interne *)
  | E                           (* Arbre vide *)
```

Autrement énoncé :
- $\square \in \mathbb{B}$
- Si $g \in \mathbb{B}$ et $d \in \mathbb{B}$ alors 

$\in \mathbb{B}$

Q. 10 Donner une bijection naturelle de $\mathbb{B}$ dans $\mathcal{L}$, induite par la **Q. 8**

Q. 11 Définir en OCaml deux fonctions `to_word : btree -> word` et `from_word: word -> btree` telles que : $\forall w \in \mathcal{L}$, `(to_word (from_word w))` $= w$ et $\forall b \in \mathbb{B}$, `(from_word (to_word b))` $= b$.

Exemples :
```
from_word [R; R; C; C; R; C; C] = N (N (E, E), N (E, E))
to_word (N (N (E, E), N (E, E))) = [R; R; C; C; R; C; C]
```

Q. 12 Démontrer que $\forall w \in \mathcal{L}, |w| \geqslant 3 \Rightarrow \exists (u, v) \in \Sigma^\star, w = u \cdot \circ\square\square \cdot v$.