TD 6

Exercice 1.
Donner des grammaires algébriques engendrant les langages suivants. Pour tout mot w on note w^R son miroir. C'est-à-dire que si $w = w_1 \dots w_n$ alors $w^R = w_n \dots w_1$.

1. $\{w \# w' \mid w, w' \in \{a, b\}^* \wedge w \neq w'\}$
2. $\Sigma^* - \{w\, w \mid w \in \Sigma^*\}$

   Indication. *Les mots de longueur paire qui ne sont pas des carrés peuvent s'écrire comme un produit de deux mots de longueur impaires dont les lettres du milieu diffèrent.*

Exercice 2. *If Then Else*
On considère la grammaire suivante :

$$Stmt \to \textbf{if b then } Stmt \mid \textbf{if b then } Stmt \textbf{ else } Stmt \mid \textbf{a}$$

1. Montrer que cette grammaire est ambiguë.
2. Proposer une grammaire non ambiguë pour le même langage.

Exercice 3. *Chomsky*

Définition. Une grammaire hors-contexte $G = (V, \Sigma, P, S)$ [1] est sous *forme normale de Chomsky* ssi toutes ses règles sont de la forme

(i) $A \longrightarrow B\,C$ avec $B, C \in V$

(ii) $A \longrightarrow a$ avec $a \in \Sigma$

(iii) $S \longrightarrow \varepsilon$

De plus, si $S \longrightarrow \varepsilon$ est une règle de P, alors $B, C \in V - \{S\}$ dans (i).

Le but de cet exercice est de prouver le théorème suivant :

Théorème. *Pour toute grammaire hors-contexte $G = (V, \Sigma, P, S)$, il existe une grammaire $G' = (V', \Sigma, P', S)$ telle que $\mathscr{L}(G') = \mathscr{L}(G)$ et G' est sous forme normale de Chomsky.*

Considerons les cinq transformations suivants :

- **START** : On introduit S_0 nouvelle variable de départ et la règle $S_0 \to S$.
- **TERM** : Pour chaque lettre $a \in \Sigma$ terminale, on ajoute une variable V_a avec la règle $V_a \longrightarrow a$. Puis, si dans une règle avec au moins 2 membres droits, un membre droit a est un terminal, on le remplace par V_a.
- **BIN** : On remplace toute règle $A \to B_1 \dots B_k$ avec $k \geq 3$ par $A \to B_1 V_1, V_1 \to B_2 V_2, \dots, V_{k-2} \to B_{k-1} B_k$ avec $V_1, \dots, V_{k-2}$ qui sont $k - 2$ nouvelles variables.
- **DEL** : On élimine les ε-règles : Il faut d'abord trouver quelles sont les variables *annulables*, c'est à dire les variables qui peuvent engendrer le mot vide. On peut construire cet ensemble par induction : les variables qui ont une règle $A \longrightarrow \varepsilon$ son annulables, puis les variables qui ont une règle $A \longrightarrow A_1 A_2, \dots A_n$ avec $A_1, A_2, \dots A_n$ annulables sont elles aussi annulables. Une fois que l'on a cet ensemble de variables annulables, on regarde chaque règle de notre grammaire, et s'il y a une variable annulable à droite, on ajoute la même règle mais sans cette variable. Par exemple, si on a la règle $A \longrightarrow A_1 A_2$ et que A_2 est annulable, alors on ajoute la règle $A \longrightarrow A_1$. Notez que cette construction peut introduire de nouvelles ε-règles. Une fois que l'on a fait ça, on supprime toutes les règles qui mènent à ε sauf pour l'axiome S_0.

1. Rappel : $V \cap \Sigma = \varnothing$

† **UNIT** : Enfin, on élimine les règles unitaires c'est-à-dire du type $A \longrightarrow B$ avec $B \in V$. Première étape, on considère le graphe de ces règles unitaires (donc on a une arrête de A vers B ssi la règle $A \longrightarrow B$ existe) et on regarde les composantes fortement connexes. Pour chacune d'entre elles, on choisit un représentant et on remplace toutes les autres variables de cette composante par ce représentant dans notre grammaire (aussi bien dans les membres gauches que dans les membres droits). Puis, on élimine les règles de la forme $A \longrightarrow A$.

Ensuite, ptant que l'on a une règle de la forme $A \longrightarrow B$, on supprime cette règle et on la remplace par $A \longrightarrow \alpha$ pour chaque $B \longrightarrow \alpha$ que l'on a encore dans la grammaire. On fait ceci jusqu'à ce qu'il n'y ait plus de règles unitaires.

1. Montrez que cet algorithme est correct : Il finit, ne change pas le langage engendré et renvoie une grammaire en FNC
2. Quel est le langage reconnu par la grammaire :

$$\begin{aligned} S &\longrightarrow X\,C \mid \varepsilon \\ X &\longrightarrow a\,X\,b\,b \mid \varepsilon \\ C &\longrightarrow c\,C \mid \varepsilon \end{aligned}$$

3. Faites tourner l'algorithme sur cette grammaire (on pourra se passer de la première étape ici)

Exercice 4. *Intersection*

1. Montrer que l'intersection d'un langage algébrique avec un langage rationnel sur le même alphabet est algébrique.
2. Est-ce vrai pour l'intersection de deux langages algébriques ?

Exercice 5. *Ogden*

Montrer que les langages suivants ne sont pas algébriques.

1. $L = \{0^n\,1^n\,0^n\,1^n \mid n \in \mathbb{N}\}$
2. $L = \{0^n \,\#\, 0^{2n} \,\#\, 0^{3n} \mid n \in \mathbb{N}\}$
3. $L = \{w \in \{a,b,c\}^* \mid |w|_a = |w|_b = |w|_c\}$
4. $L = \{t_1 \# t_2 \# \cdots \# t_k \mid k \geq 2 \wedge (\forall i \in [\![1\,;\,k]\!] \qquad t_i \in \{a,b\}^*) \wedge \big(\exists i,j \in [\![1\,;\,k]\!] \qquad i \neq j \wedge t_i = t_j\big)\}$
5. $L = \{a^i\,b^j \mid i,j \in \mathbb{N} \wedge j \mid i\}$