A.2 Transformation de grammaires

1. Soit l'alphabet $X = \{+, =, a\}$. (1) Donner une grammaire algébrique pour le langage L dont chaque mot représente une addition correcte de deux suites de caractères a. Par exemple L contient le mot $aa + aaaa = aaaaaa$. (2) Donner un automate à pile qui reconnaît le même langage.
2. Soit le langage L_1 sur le vocabulaire $V = \{$*l'*, *homme*, *ours*, *qui*, *a*, *vu*$\}$ formé de l'ensemble des phrases finies de la forme *l'homme qui a vu l'ours*, *l'homme qui a vu l'homme qui a vu l'ours*, *l'homme qui a vu l'homme qui a vu . . . qui a vu l'ours*.

   (a) Donner une grammaire algébrique (*context-free*) engendrant L_1.
   (b) Donner une grammaire régulière engendrant L_1.

   Soit L_2 le langage engendré par la grammaire $\mathcal{G}_2$:

   $$\begin{array}{lcl} \text{S} & \rightarrow & \text{NP Rel} \\ \text{NP} & \rightarrow & \text{l'homme} \\ & \mid & \text{l'ours} \\ \text{Rel} & \rightarrow & \text{qui a vu NP Rel} \\ & \mid & \text{qui a vu l'ours} \end{array}$$

   (c) Mettre la grammaire $\mathcal{G}_2$ sous forme normale de Chomsky.
   (d) Proposer une grammaire $\mathcal{G}_3$ qui engendre le langage L_3, sur-ensemble de L_2 mais dans lequel les symboles *ours* et *homme* sont **strictement** interchangeables.
   (e) Décrire informellement les différences entre les langages L_1 et L_2.
   (f) Donner une grammaire algébrique du langage $L_2 \setminus L_1$.
3. Mettre sous forme normale de Chomsky la grammaire définie par les règles de production suivantes
   $$\begin{array}{l} S \rightarrow AB \mid aS \mid a \\ A \rightarrow Ab \mid \varepsilon \\ B \rightarrow AS \end{array}$$
4. (a) Montrer que la grammaire suivante est ambiguë :
   $S \rightarrow TU; T \rightarrow ST \mid a; U \rightarrow US \mid b$

   (b) La grammaire suivante est-elle ambiguë ? $S \rightarrow aSSb \mid ab$
5. Eliminer la récursivité gauche de la grammaire ETF.
6. Transformer en forme normale de Greibach la grammaire suivante (qui est déjà non récursive gauche) :
   $$\begin{array}{l} S \rightarrow Aa \mid b \\ A \rightarrow bdC \mid cC \\ C \rightarrow cC \mid adC \mid c \mid ad \end{array}$$
7. Appliquer l'algorithme de dé-récursivation (gauche) vu en cours à la grammaire suivante, grammaire de la liste.
   $S \rightarrow (\, L\,) \mid a$
   $L \rightarrow L, S \mid S$
8. Dé-récursiver la grammaire suivante
   $$\begin{array}{l} S \rightarrow aS \mid BS \mid \varepsilon \\ B \rightarrow bAb \mid SaS \\ A \rightarrow a \mid Sa \end{array}$$