Exercices sur les grammaires 26 septembre 2007

Corrigé

Exercice 1

1. La grammaire des nombres décimaux de la suite de Fibonacci inférieurs à 20 est un quadruplet $\langle \Sigma = \{1,2,3,4,5,6,7,8,9\}, N = \{S\}, P, S\rangle$ où P est l'ensemble de productions suivant.

$$P = \left\{ \begin{array}{rcl} S & = & 1|2|3|5|8|13 \end{array} \right\}$$

   Il n'existe pas de grammaire plus simple : la relation entre deux nombres de la suite de Fibonacci dépend de propriétés qui ne peuvent être encodées dans une grammaire (addition de deux entiers, ordre des éléments).

2. La grammaire des mots qui contiennent la chaîne *baab* est, par exemple, un quadruplet $\langle \Sigma = \{a,b\}, N = \{S,W\}, P, S\rangle$ où P est l'ensemble de productions suivant.

$$P = \left\{ \begin{array}{rcl} S & = & WbaabW \\ W & = & bW|aW| \end{array} \right\}$$

3. La définition d'une grammaire des mots dont le nombre de a et de b est égal est un peu plus subtile et peut se faire de différentes manières.

   Une solution est de garantir la propriété d'égalité du nombre de a et de b après chaque production. Dans ce cas, une grammaire est un quadruplet $\langle \Sigma = \{a,b\}, N = \{S\}, P, S\rangle$ ou P est, par exemple, un des ensembles de productions suivant.

$$P = \left\{ \begin{array}{rcl} S & = & aSb|bSa|abS|baS|Sab|Sba| \end{array} \right\} \quad (1)$$
$$P = \left\{ \begin{array}{rcl} S & = & aSb|bSa|SS| \end{array} \right\} \quad (2)$$
$$P = \left\{ \begin{array}{rcl} S & = & aSbS|bSaS| \end{array} \right\} \quad (3)$$

   Une autre solution est de construire la grammaire de telle sorte que, lorsqu'un a supplémentaire est ajouté au mot, on garantit qu'il sera

compensé ensuite par un b. Une telle grammaire peut, par exemple, être une quadruplet $\langle \Sigma = \{a, b\}, N = \{S, A, B\}, P, S\rangle$ où P est l'ensemble de productions suivant.

$$P = \left\{ \begin{array}{lcl} S & = & aA|bB| \\ A & = & bS|aAA \\ B & = & aS|bBB \end{array} \right\} \quad (4)$$

Intuitivement, on peut comprendre la production A comme celle des mots où un a manque. Dans ce cas, il faut accepter un b pour revenir dans l'étant S d'égalité entre lettres. Autrement, ce sont deux a qui manqueront, que l'on compensera en appelant deux fois la production A.

4. Pour les différentes grammaires proposées, le mot *abbaabab* peut se déduire de la façon suivante.

   (1) $abbaabab \overset{S=}{\longleftarrow} abbaaSbab \overset{S=aSb}{\longleftarrow} abbaSab \overset{S=baS}{\longleftarrow} abSab \overset{S=bSa}{\longleftarrow} aSb \overset{S=aSb}{\longleftarrow} S$

   (2) $abbaabab \overset{S=}{\longleftarrow} abbSaabab \overset{S=}{\longleftarrow} abbSaaSbab \overset{S=bSa}{\longleftarrow} abSaSbab \overset{S=aSb}{\longleftarrow} abSSab \overset{S=SS}{\longleftarrow} abSab \overset{S=bSa}{\longleftarrow} aSb \overset{S=aSb}{\longleftarrow} S$

   (3) $abbaabab \overset{S=}{\longleftarrow} abbaababS \overset{S=}{\longleftarrow} abbaabaSbS \overset{S=aSbS}{\longleftarrow} abbaabS \overset{S=}{\longleftarrow} abbaaSbS \overset{S=aSbS}{\longleftarrow} abbaS \overset{S=}{\longleftarrow} abbSaS \overset{S=bSaS}{\longleftarrow} abS \overset{S=}{\longleftarrow} aSbS \overset{S=aSbS}{\longleftarrow} S$

   (4) $abbaabab \overset{S=}{\longleftarrow} abbaababS \overset{A=bS}{\longleftarrow} abbaabaA \overset{S=aA}{\longleftarrow} abbaabS \overset{A=bS}{\longleftarrow} abbaaA \overset{S=aA}{\longleftarrow} abbaS \overset{B=aS}{\longleftarrow} abbB \overset{S=bB}{\longleftarrow} abS \overset{A=bS}{\longleftarrow} aA \overset{S=aA}{\longleftarrow} S$

Exercice 2

1. Le langage $L_{2.1}$ défini par la grammaire décrite dans la donnée est le langage des mots w sur $\Sigma = \{a, b\}$ tels que

   $$\forall w \,.\, \exists n, m \in \mathbb{N} \,.\, m > 0 \wedge w = a^n b^m a^n$$

   où a^n est la répétition n fois du caractère a.
   Ce langage n'est pas régulier, une propriété que l'exercice ne demandait pas de prouver.

Mais dans l'intérêt de votre culture générale, on le démontrera quand même, à l'aide du "pumping lemma" qui se définit de la manière suivante.
Un langage L n'est pas régulier si

$$\begin{cases} \forall p \in \mathbb{N} \,.\, p \geq 1 \\ \exists w \in L \,.\, |w| \geq p \\ \forall x, y, z \,.\, w = xyz \wedge |y| > 0 \wedge |xy| \leq p \\ \exists i \in \mathbb{N} \,.\, i \geq 0 \\ xy^i z \notin L \end{cases}$$

Démonstration. Considérons tout $p \in \mathbb{N} \geq 1$, on définit $w = a^p b a^p$, un mot de $L_{2.1}$ pour lequel la propriété $|w| \geq p$ est trivialement vraie.
La décomposition de w en xyz sous $|y| > 0 \wedge |xy| \leq p$ ne peut se faire que de deux façons (pour $k, j \in \mathbb{N}$) :

$$\begin{cases} x = a^k, y = a^{p-k}, z = ba^p & k < p & (1) \\ x = a^k, y = a^j, z = a^{p-k-j}ba^p & k + j < p, j > 0 & (2) \end{cases}$$

Considérons $i = 0$, on veut montrer que pour chacune des deux décompositions ci-dessus, $xy^i z = xy^0 z = xz \notin L_{2.1}$.

$$\begin{cases} a^k b a^p \notin L_{2.1} \text{ parce que } k < p & (1) \\ a^k a^{p-k-j} b a^p = a^{p-j} b a^p \notin L_{2.1} \text{ parce que } j > 0 & (2) \end{cases}$$

$L_{2.1}$ n'est donc pas régulier. □

2. (a) La grammaire des nombres décimaux de la suite de Fibonacci inférieurs à 20 définit un langage régulier.

   Ceci se démontre en présentant une grammaire de type EBNF non-récursif (ou un automate à états finis) équivalent à la grammaire originale. Dans ce cas, l'unique production est déjà au format EBNF non-récursif : le langage est donc régulier.

   (b) La grammaire des mots qui contiennent la chaîne *baab* définit également un langage régulier. La production EBNF non-récursive suivante l'accepte.

$$S \quad = \quad \{a|b\}baab\{a|b\}$$

   (c) La grammaire des mots dont le nombre de a et de b est égal ne définit pas un langage régulier.

Exercice 3

La grammaire des identifiants Scala est un quadruplet $\langle \Sigma, N, P, Ident \rangle$ où Σ est l'ensemble des caractères suivant (pour simplifier, on ne considérera pas les caractères accentués)

$$\Sigma = \{\texttt{a}, \ldots, \texttt{z}, \texttt{A}, \ldots, \texttt{Z}, \texttt{1}, \ldots, \texttt{9}, \texttt{0}, \texttt{\_}, \texttt{+}, \texttt{=}, \ldots\}$$

N est l'ensemble des non-terminaux suivant

$$N = \{Ident, Normal, Special, Letter, Digit\}$$

et P est l'ensemble de productions suivant.

$$P = \left\{ \begin{array}{rcl} Ident & = & Letter\{Normal\}[\_\{Special\}] \\ & | & \_\{Normal\}[\_\{Special\}] \\ & | & Special\{Special\} \\ Normal & = & Letter|Digit|\_ \\ Special & = & \texttt{+}|\texttt{=}|\cdots|\_ \\ Letter & = & \texttt{a}|\cdots|\texttt{z}|\texttt{A}|\cdots|\texttt{Z} \\ Digit & = & \texttt{1}|\texttt{2}|\texttt{3}|\texttt{4}|\texttt{5}|\texttt{6}|\texttt{7}|\texttt{8}|\texttt{9}|\texttt{0} \end{array} \right\}$$