SI 3 — **Année 2007-2008**
Informatique théorique — **Langages Formels et Automates**

Grammaires

Corrigé partiel de la feuille de travaux dirigés n^{o}7

1 avril 2008

1. Numérotons les règles :

$$\begin{array}{ll} S \to_1 aB & A \to_5 bAA \\ S \to_2 bA & B \to_6 b \\ A \to_3 a & B \to_7 bS \\ A \to_4 aS & B \to_8 aBB \end{array}$$

a) On a trois dérivations gauches. Toutes trois commencent par $S \to_1 aB \to_8 aaBB \to_8 aaaBBB$ mais diffèrent ensuite

$$aaaBBB \to_6 aaabBB \to_7 aaabbSB \to_1 aaabbaBB$$
$$aaaBBB \to_7 aaabSBB \to_2 aaabbABB \to_3 aaabbaBB$$
$$aaaBBB \to_6 aaabBB \to_6 aaabbB \to_8 aaabbaBB$$

puis se terminent de la même manière :$aaabbaBB \to_6 aaabbabB \to_7 aaabbabbS \to_2 aaabbabbbA \to_3 aaabbabbba$

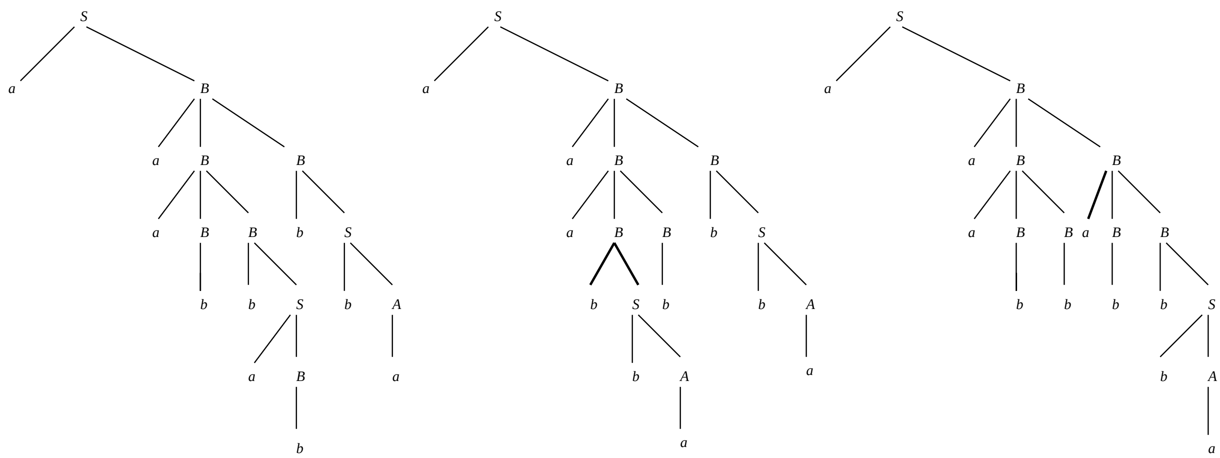
b) Comme pour la dérivation gauche, on a trois dérivations droites qui commencent de la même manière$S \to_1 aB \to_8 aaBB$ mais différent ensuite

$aaBB \to_7 aaBbS \to_2 aaBbbA \to_3 aaBbba \to_8 aaaBBbba \to_7 aaaBbSbba \to_1 aaaBbaBbba \to_6 aaaBbabbba \to_6 aaabbabbba$

$aaBB \to_7 aaBbS \to_2 aaBbbA \to_3 aaBbba \to_8 aaaBBbba \to_6 aaaBbbba \to_7 aaabSbbba \to_2 aaabbAbbba \to_3 aaabbabbba$

$aaBB \to_8 aaBaBB \to_7 aaBaBbS \to_2 aaBaBbbA \to_3 aaBaBbba \to_6 aaBabbba \to_8 aaaBBabbba \to_6 aaaBbabbba \to_6 aaabbabbba$

c) Un arbre syntaxique qui engendre le mot $aaabbabbba$:



Un parcours en profondeur gauche de cet arbre donne une dérivation gauche associée, idem pour la droite. Ce mot a en fait trois arbres syntaxiques, ce qui implique que la grammaire est ambiguë.

Remarque : On observe que :

- la grammaire donnée engendre l'ensemble des mots ayant autant de a que de b,

- la variable A engendre l'ensemble des mots ayant exactement 1 a de plus que de b,
- la variable B engendre l'ensemble des mots ayant exactement 1 b de plus que de a.

2.

a) On peut partir du fait que $L_1 = \{a^n b^n | n > 0\}c^+$. Il suffit alors de trouver une variable A qui engendre $\{a^n b^n | n > 0\}$, et une variable C qui engendre c^+ pour engendrer L_1 par une variable S avec la règle : $S \to AC$.
Or on peut engendrer le langage (rationnel) c^+ via les 2 règles suivantes : $C \to c \mid cC$ et le langage $\{a^n b^n | n > 0\}$ via les 2 règles suivantes : $A \to ab \mid aAb$. On obtient :
$N = \{S, A, C\}; T = \{a, b, c\}; S$ et $P = \{S \to AC, A \to ab \mid aAb, C \to c \mid cC\}$ qui engendre le langage algébrique L_1.
b) L'idée est de dériver des mots dont le début est identique à la fin, puis à un moment donné d'engendrer une différence. A partir de ce moment, la seule contrainte à respecter est la parité de la longueur. On obtient donc :
$N = \{S, T, U\}; T = \{a, b\}; S$ et $P = \{S \to aSa \mid bSb \mid aTb \mid bTa, T \to \varepsilon \mid aU \mid bU, U \to aT \mid bT\}$
c) Le langage L_3 est l'union de 2 langages, $L_{31} = \{a^i b^j \mid i \neq j\}c^+$ et $L_{32} = a^+\{b^i c^j \mid i \neq j\}$. Soient les variables S_{31} et S_{32} qui engendrent respectivement L_{31} et L_{32}, et la variable S_3 qui permet d'engendrer le langage union L_3 via la règle : $S_3 \to S_{31} \mid S_{32}$.
$N = \{S_3, S_{31}, S_{32}, T_{31}, T_{32}, A, B, C\}, T = \{a, b, c\}, S_3$

$$P \left\{ \begin{array}{ll} S_3 \to S_{31} \mid S_{32} & T_{32} \to bT_{32}c \mid bB \mid cC \\ S_{31} \to T_{31}C & A \to aA \mid \varepsilon \\ T_{31} \to aT_{31}b \mid aA \mid bB & B \to bB \mid \varepsilon \\ S_{32} \to AT_{32} & C \to cC \mid \varepsilon \end{array} \right.$$

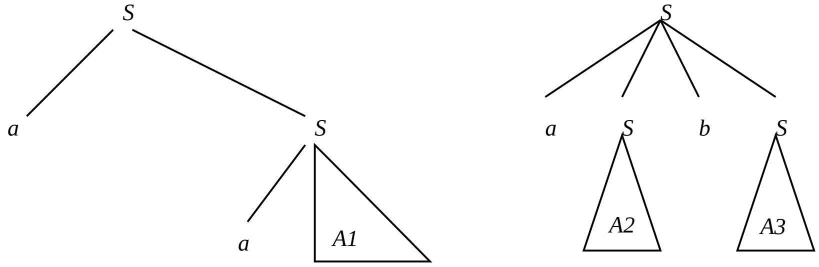
d) On peut utiliser le langage et la grammaire de l'exercice 1. Il suffit de l'adapter, pour engendrer à la place des mots contenant autant de a que de b les mots contenant soit plus de a que de b, soit plus de b que de a. Ainsi A (resp. B) engendre les mots ayants un excédent de a (resp. b) et E engendre les mots équilibrés. On obtient : $N = \{S, A, B, E, C, D\}, T = \{a, b\}, S,$ $P \left\{ \begin{array}{ll} S \to A \mid B & E \to aD \mid bC \mid \varepsilon \\ A \to aE \mid aA \mid EA & C \to a \mid aE \mid bCC \\ B \to bE \mid bB \mid EB & D \to b \mid bE \mid aDD \end{array} \right.$
e) Il suffit d'utiliser les règles qui permettent de créer des expressions rationnelles (comme dans la grammaire ETF). $N = \{E, T, F\}, T = \{a, b, c, *, +, (,), e\}$ où e représente ε, E $P \left\{ \begin{array}{l} E \to TE \mid T* \mid T \\ T \to F + T \mid F \\ F \to (E) \mid a \mid b \mid c \mid e \end{array} \right.$

3. Notons L_S le langage engendré par S et L le langage des mots w vérifiant la propriété $P(w)$:

$P(w)$: tout préfixe de w a au moins autant de a que de b.

Il s'agit de montrer que $L_S = L$.

- On montre d'abord que tout mot w de L_S vérifie $P(w)$, c.a.d. $L_S \subseteq L$, par induction sur la hauteur des arbres syntaxiques. Pour les mots de L_S obtenus avec un arbre syntaxique de hauteur 1 (i.e. ε) ou de hauteur 2 (i.e. a ou ab) la propriété $P(w)$ est vraie. Soit n un entier, supposons $P(w)$ vraie pour tous les mots de L_S obtenus avec un arbre syntaxique de hauteur inférieure ou égale à n. Soit w un mot de L_S obtenu avec un arbre syntaxique de hauteur $n+1$. Les arbres pouvant donner w sont de l'un des 2 types suivants :



où $A1$ est un arbre syntaxique de hauteur n, et $A2$, $A3$ sont 2 arbres syntaxiques dont au moins 1 est de hauteur n. Par hypothèse d'induction, les mots obtenus à partir de ces arbres vérifient $P(w)$. Il est alors immédiat de vérifier que w vérifie aussi $P(w)$. Ainsi tous les mots de L_S vérifient $P(w)$ et donc $L_S \subseteq L$.

- Réciproquement, on montre que $L \subseteq L_S$. Les mots de L sont exactement les mots w tels que pour tout préfixe w' de w on a : $|w'|_a - |w'|_b \geq 0$. Remarquons que tout mot de L commence par a et qu'il y a 2 types de mots dans L :

  1. les mots w tels que pour tout préfixe w' de w on a : $|w'|_a - |w'|_b > 0$ - ces mots s'écrivent am, où m est un mot de L
  2. les mots w tels que pour au moins 1 préfixe w' on a : $|w'|_a - |w'|_b = 0$. Soit v le plus court des préfixes w'. $w = vv'$, où :
     - $v \in L$ (tout préfixe d'un mot de L est aussi un mot de L) qui se termine par b (car $|v|_a - |v|_b = 0$), et donc $v = aub$ où $u \in L$ (car v étant le plus petit mot du type 2., au est du type 1.)
     - $v' \in L$, car $|v|_a - |v|_b = 0$ et que vv' (i.e. w) est un mot de L.

  On raisonne alors par induction sur la longueur des mots de L. Les mots de L de longueur 0 ou 1 sont bien engendrés par la grammaire. Soit n un entier, supposons que tous les mots de L de longueur inférieure ou égale à n sont obtenus par la grammaire. Soit w un mot de longueur $n + 1$:

  1. soit il est du type 1., i.e. $w = am$ où m est un mot de L de longueur n et donc engendré par la grammaire, d'où en appliquant $S \to aS$, puis en dérivant S pour obtenir m, on obtient w à partir de S.
  2. soit il est du type 2., i.e. $w = aubv$ où $u, v \in L$ sont de longueur inférieure à n et donc engendrés par la grammaire, en appliquant $S \to aSbS$, puis en dérivant le premier S pour obtenir u et le deuxième S pour obtenir v, on obtient w à partir de S.

  Donc tous les mots de L sont engendrés par la grammaire et $L \subseteq L_S$.

D'où le résultat.