Informatique théorique III
Série 7 solution

Prof. Nestmann, 2004

1. Grammaires quelconques

Pour chacun des langages suivants, donner une grammaire le générant. Indiquer le type des grammaires trouvées.

1. $L_0 = \{\text{airbag}, \text{mandragore}\}$, langage sur l'alphabet $\Sigma_0 = \{\text{a,b,}\ldots\text{,z}\}$.
2. $L_1 = \{a^n b^n \mid n \in \mathbb{N}\}$, langage sur l'alphabet $\Sigma_1 = \{a, b\}$.
3. $L_2 = \{a^n b^n c^n \mid n \in \mathbb{N}\}$, langage sur l'alphabet $\Sigma_2 = \{a, b, c\}$.
4. $L_3 = \left\{a^{(2^n)} \mid n \in \mathbb{N}\right\}$, langage sur l'alphabet $\Sigma_3 = \{a\}$.

Solution.

1. $G_0 = (\{S\}, \Sigma_0, P, S)$ avec P tel que :

$$S \to \text{airbag} \mid \text{mandragore}$$

   G_0 est de type 3.

2. $G_1 = (\{S\}, \Sigma_1, P, S)$ avec P tel que :

$$S \to aSb \mid \epsilon$$

   G_1 est de type 2 : c'est une grammaire non-contextuelle.

3. $G_2 = (\{S, A, B, C\}, \Sigma_2, P, S)$ avec P tel que :

$$\begin{aligned} S &\to aSBC \mid \epsilon \\ aB &\to ab \\ bB &\to bb \\ bC &\to bc \\ cC &\to cc \\ CB &\to BC \end{aligned}$$

   G_2 est de type 1.
   L'idée est de générer des chaînes commençant par des a et contenant autant de a que de symboles non terminaux B et C. La règle $CB \to BC$ permet de réordonner les non-terminaux de façon à placer les B avant les C. On propage ensuite l'écriture du mot de gauche vers la droite grâce aux règles $aB \to ab$, $bB \to bb$, $bC \to bc$ et $cC \to cc$. Noter que pour obtenir un mot terminal, on doit avoir échangé tous les B et C.

4. $G_3 = (\{A, S, [,]\}, \Sigma_3, P, S)$ avec P tel que :

$$\begin{aligned} S &\to [S] \mid a \\ a] &\to]aa \\ [\,] &\to \epsilon \end{aligned}$$

   G_3 est de type 0.

∎

2. Associativité et grammaires non-contextuelles

On considère comme alphabet $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -\}$.
Soit $G = (\{S, N\}, \Sigma, P, S)$ avec P tel que :

$$\begin{aligned} S &\to N - S \mid N \\ N &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{aligned}$$

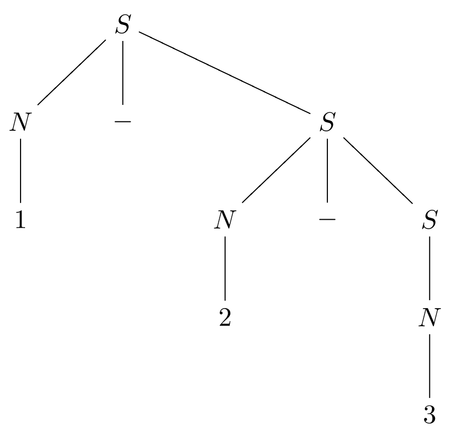
On interprète les symboles $0, \ldots, 9$ par les entiers correspondants. Étant donné un arbre d'analyse (t, ψ) de G, on définit l'interprétation $I(t, \psi)$ de l'arbre, qui calcule la valeur de l'expression arithmétique que celui-ci représente, par :

$$\begin{aligned} I((n), \psi) &\triangleq \psi(n) \\ I((n\ t), \psi) &\triangleq I(t, \psi) \\ I((n\ t_1 t_2 t_3), \psi) &\triangleq I(t_1, \psi) - I(t_3, \psi) \end{aligned}$$

1. Donner l'arbre (les arbres ?) de dérivation (t, ψ) de la chaîne $1 - 2 - 3$ dans G. Que vaut $I(t, \psi)$?
2. Comme vous avez du le remarquer, l'expression précédente ne s'évalue pas en ce que l'on souhaiterait — à savoir -4. Donner alors une grammaire G' tel que l'opérateur $-$ soit *associatif à gauche* et non à droite comme G le fait.
3. Donner l'arbre de dérivation (t', ψ') de la chaîne $1 - 2 - 3$ dans G' et vérifier que $I(t', \psi') = -4$.

Solution.

1. L'unique arbre de dérivation de $1 - 2 - 3$ dans G est :

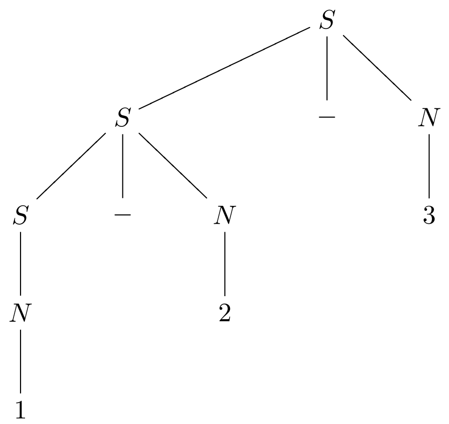


   L'interprétation de cet arbre est $1 - (2 - 3) = 1 - (-1) = 1 + 1 = 2$.

2. Soit $G' = (\{S, N\}, \Sigma, P', S)$ avec P' tel que :

$$\begin{aligned} S &\to S - N \mid N \\ N &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{aligned}$$

3. L'unique arbre de dérivation de $1-2-3$ dans G' est alors :



L'interprétation de cet arbre est $(1-2)-3 = -1-3 = -4$.

■

3. De l'ambiguïté

Soit $\Sigma = \{0, \ldots, 9, +, *\}$ un alphabet terminal et $G = (\{E, N\}, \Sigma, P, E)$ avec P tel que :

$$E \to N \mid E+E \mid E*E$$
$$N \to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

1. Démontrer l'ambiguïté de G en exhibant deux arbres d'analyse pour le mot $1{+}2{*}3$. Le mot $45{*}2$ appartient-il à $\mathrm{L}(G)$?
2. Dans ce qui suit, on confond les lettres $0, \ldots, 9$ avec les chiffres qui leur correspondent et les lettres $*$ et $+$ avec les opérations idoines sur les nombres naturels. Étant donné un arbre d'analyse (t, ψ) de G, on définit l'interprétation $I(t, \psi)$ de l'arbre, qui calcule la valeur de l'expression arithmétique que celui-ci représente, par :

$$I\big((n\ t_1\ t_2\ t_3), \psi\big) = I(t_1, \psi)\ \mathbf{op}\ I(t_3, \psi) \qquad \text{avec } \mathbf{op} = \psi\,(\mathrm{racine}(t_2))$$
$$I\big((n\ t), \psi\big) = I(t, \psi)$$
$$I\big((n), \psi\big) = \psi(n)$$

   N.B. la fonction ci-dessus est bien définie sur les arbres d'analyse de G car leurs nœuds n'ont que un ou trois sous-arbres directs (pourquoi ?).

   Calculer l'interprétation des deux arbres d'analyse du point précédent.
3. Changer la grammaire G en une grammaire $G' = (V, \Sigma, P', E)$ pour la rendre non-ambiguë et telle que l'ordre de priorité des opérateurs soit, du plus faible au plus fort, $+, *$ (c.-à.d, l'interprétation de $1{+}2{*}3$ devrait être 7).
4. Donner l'unique arbre d'analyse de $1{+}2{*}3$ dans G'.

   En définissant une fonction d'interprétation adéquate sur les arbres d'analyse de G' comme ci-dessus, l'unique interprétation de $1{+}2{*}3$ devrait être 7. Cependant, parfois, on aimerait pouvoir écrire des mots tels que $(1{+}2){*}3$, à interpréter par 9, les parenthèses servant à changer l'ordre de priorité des opérateurs.

Donner une grammaire $G'' = (V', \Sigma \cup \{),(\}, P'', E)$ qui garde l'ordre de priorité des opérateurs de la grammaire G' mais qui permet aussi d'introduire des parenthèses pour changer l'ordre de priorité des opérateurs.

5. Soit la grammaire $G''' = (\{E, N\}, \Sigma, P''', E)$ avec P''' tel que :

$$\begin{aligned} E &\to N \;\big|\; +EE \;\big|\; *EE \\ N &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{aligned}$$

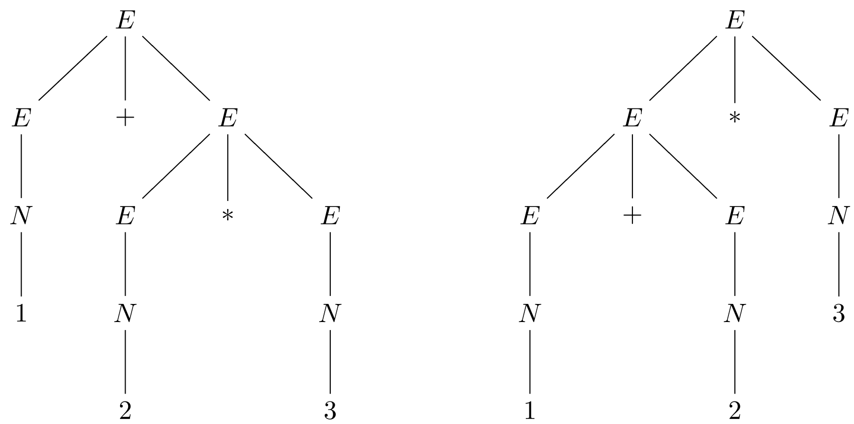
Cette grammaire génère le même genre d'expression que G mais en notation préfixe. Donner les arbres de dérivation de $+1{*}23$ et $*{+}123$ et démontrer que cette grammaire n'est pas ambiguë.

Pour cela utiliser le lemme suivant :

Lemme 3.1 *Soit G une grammaire non-contextuelle et $u\beta \Rightarrow^* w$ une dérivation dans G avec $u, w \in \Sigma^*$. Alors il existe $w' \in \Sigma^*$ tel que $w = u \cdot w'$.*

Solution.

1.



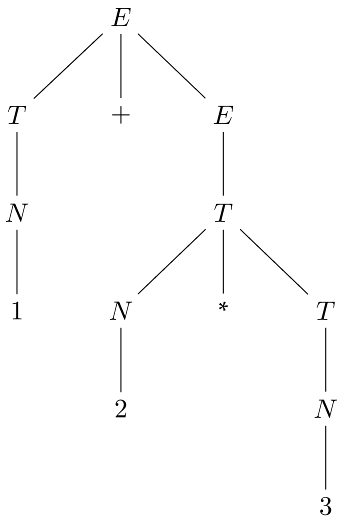
L'expression $45{*}2$ n'appartient pas à $\mathrm{L}(G)$ car il n'existe pas de d'arbre d'analyse pour ce mot.

2. En appliquant la définition, on obtient 7 pour le premier ci-desssus et 9 pour le second. Ceci montre que l'ambiguïté n'est pas une propriété désirable pour les grammaires décrivant les langages de programmation.

3. $G' = (\{E, T, F\}, \Sigma, P', E)$ avec P' tel que :

$$\begin{aligned} E &\to T + E \;\big|\; T \\ T &\to N * T \;\big|\; N \\ N &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \end{aligned}$$

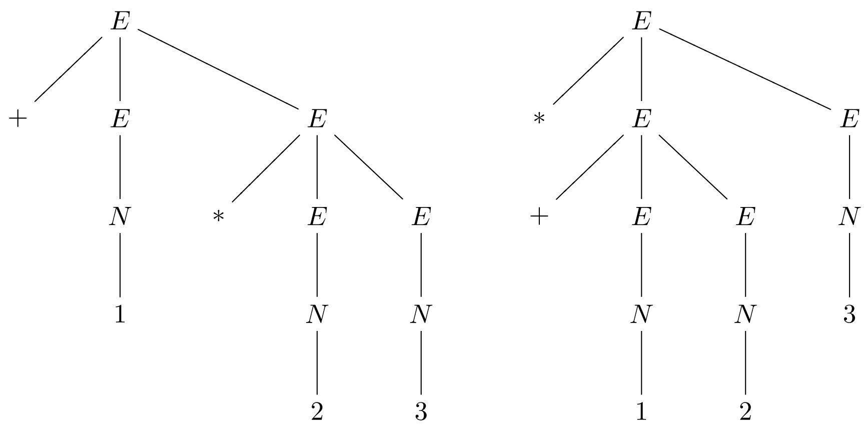
4. L'arbre unique pour 1+2∗3 est :



La grammaire $G'' = (\{E, T, F, N\}, \Sigma \cup \{), (\}, P'', E)$ avec P'' tel que :

$$
\begin{aligned}
E &\to T + E \mid T \\
T &\to F * T \mid F \\
T &\to (E) \mid N \\
N &\to 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9
\end{aligned}
$$

5. Les arbres de +1∗23 et ∗+123 sont respectivement.



Raisonnons par l'absurde. Supposons$^{(*)}$ que G''' est ambiguë. Cela signifie qu'il existe une chaîne $w \in \mathrm{L}(G''')$ qui admet deux dérivations distinctes les plus à gauche pour w :

$$
\begin{aligned}
E = \alpha_0 \Rightarrow \ldots \Rightarrow \alpha_n = w \\
E = \alpha'_0 \Rightarrow \ldots\ldots \Rightarrow \alpha'_m = w
\end{aligned}
$$

avec $n \leq m$. Les deux dérivations partant du même symbole initial E, il existe un pas de la dérivation où elle diffèrent. C'est à dire il existe $i \in [0, n-1]$ tel que $\alpha_i = \alpha'_i = \alpha$ et $\alpha_{i+1} \neq \alpha'_{i+1}$. Les dérivation étant des dérivation les plus à gauche et la grammaire non-contextuelle nous savons que $\alpha = uA\gamma$ avec $u \in \Sigma^*$, $A \in V$ et $\gamma \in (V \cup \Sigma)^*$ et que

$$\begin{aligned} uA\gamma = \alpha &\Rightarrow \alpha_{i+1} = u\beta\gamma && \text{avec } (A \to \beta) \in P \\ uA\gamma = \alpha &\Rightarrow \alpha'_{i+1} = u\beta'\gamma && \text{avec } (A \to \beta') \in P \end{aligned}$$

et $\beta \neq \beta'$. Il y a deux cas à considérer :

(a) $A = E$. Sachant que $\beta \neq \beta'$ nous avons par définition des productions les cas suivants :

 i. $\beta = N$ et $\beta' = +EE$.
 Nous avons $\alpha_{i+1} = uN\gamma$ et $\alpha_{i+2} = un\gamma$ avec $n \in \{0, \dots, 9\}$.
 Nous avons $\alpha'_{i+1} = u{+}EE\gamma$.
 En appliquant le lemme aux deux dérivations nous obtenons qu'il existe $w', w'' \in \Sigma^*$ avec

 $$\begin{aligned} w &= unw' \\ w &= u{+}w'' \end{aligned}$$

 une contradiction car $n \neq +$.

 ii. $\beta = +EE$ et $\beta' = *EE$.
 Nous avons $\alpha_{i+1} = u{+}EE\gamma$.
 Nous avons $\alpha'_{i+1} = u{*}EE\gamma$.
 En appliquant le lemme aux deux dérivations nous obtenons qu'il existe $w', w'' \in \Sigma^*$ avec

 $$\begin{aligned} w &= u{*}w' \\ w &= u{+}w'' \end{aligned}$$

 une contradiction car $+ \neq *$

 iii. Les autres cas sont soit symétriques soit similaires à ceux donnés ci-dessus. On tombe toujours sur une contradiction.

(b) $A = N$. Sachant que $\beta \neq \beta'$ et par définition des productions :
 nous avons $\alpha_{i+1} = un\gamma$ avec $n \in \{0, \dots, 9\}$ et,
 nous avons $\alpha'_{i+1} = un'\gamma$ avec $n' \in \{0, \dots, 9\}$ et $n' \neq n$.
 En appliquant le lemme aux deux dérivations nous obtenons qu'il existe $w', w'' \in \Sigma^*$ avec

 $$\begin{aligned} w &= unw' \\ w &= un'w'' \end{aligned}$$

 une contradiction car $n' \neq n$.

Dans tous les cas on obtient une contradiction, l'hypothèse $(*)$ est donc fausse et la grammaire est non-ambiguë.

■

4. De l'ambiguïté (bis)

Donner une grammaire G' ambiguë qui génère le même langage que la grammaire $G = (\{S, S'\}, \{), (\}, P, S)$ avec P tel que :

$$\begin{aligned} S &\to S'S \mid \epsilon \\ S' &\to (S) \end{aligned}$$

Démontrer l'ambiguïté de G'.

Solution. Les productions de G' sont (par exemple) : $S \rightarrow (S) \mid SS \mid \epsilon$
Cette grammaire est ambiguë car le mot $()()()$ admet les deux dérivations distinctes les plus à gauche suivantes :

$$
\begin{array}{ll}
S \Rightarrow SS \Rightarrow SSS \Rightarrow (S)SS & \Rightarrow ()SS \Rightarrow ()(S)S \Rightarrow ()()S \Rightarrow ()()(S) \Rightarrow ()()() \\
S \Rightarrow SS \Rightarrow (S)S \Rightarrow ()S & \Rightarrow ()SS \Rightarrow ()(S)S \Rightarrow ()()S \Rightarrow ()()(S) \Rightarrow ()()()
\end{array}
$$

■