Travaux Dirigés de Langages & XML - TD 6

1 Grammaires non-ambiguës

Exercice 1 *Construire la grammaire générant tous les palindromes sur l'alphabet* $\{0, 1\}$.

CORRIGÉ :

$$S \to 0 \mid 1 \mid \varepsilon \mid 0S0 \mid 1S1$$

Exercice 2 *Soit la grammaire :*

$$\begin{array}{lcl} S & \to & (L) \mid a \\ L & \to & L,S \mid S \end{array}$$

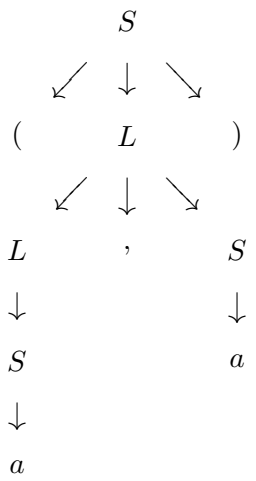
1. *Quels sont les symboles terminaux et non terminaux?*
2. *Donner les arbres de dérivation pour :*

$$(a,a) \qquad (a,(a,a)) \qquad (a,((a,a),(a,a)))$$

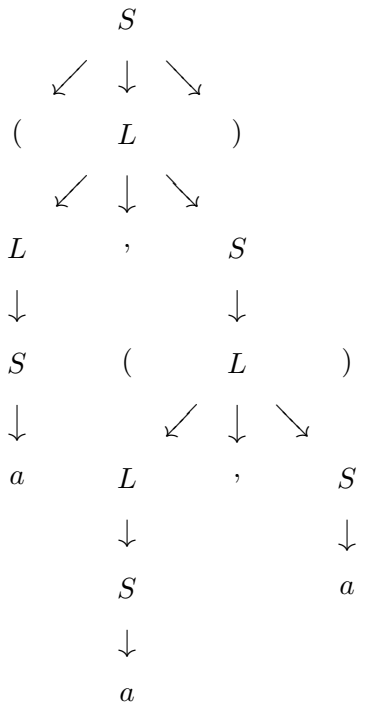
3. *Construire une dérivation à gauche et une dérivation à droite pour chacune des phrases de la question précédente.*

CORRIGÉ : L et S sont des variables, a, (,) et ',' sont des symboles terminaux.

- $S \Rightarrow (L) \Rightarrow (L,S) \Rightarrow (S,S) \Rightarrow (a,S) \Rightarrow (a,a)$.
  $S \Rightarrow (L) \Rightarrow (L,S) \Rightarrow (L,a) \Rightarrow (S,a) \Rightarrow (a,a)$.



- $S \Rightarrow (L) \Rightarrow (L,S) \Rightarrow (S,S) \Rightarrow (a,S) \Rightarrow (a,(L)) \Rightarrow (a,(L,S)) \Rightarrow (a,(S,S)) \Rightarrow (a,(a,S)) \Rightarrow (a,(a,a))$.
  $S \Rightarrow (L) \Rightarrow (L,S) \Rightarrow (L,(L)) \Rightarrow (L,(L,S)) \Rightarrow (L,(L,a)) \Rightarrow (L,(S,a)) \Rightarrow (L,(a,a)) \Rightarrow (S,(a,a)) \Rightarrow (a,(a,a))$.



–

$$\begin{aligned} S &\Rightarrow (L) \Rightarrow (L,S) \Rightarrow (S,S) \Rightarrow (a,S) \Rightarrow (a,(L)) \Rightarrow (a,(L,S)) \Rightarrow (a,(S,S)) \Rightarrow (a,((L),S)) \\ &\Rightarrow (a,((L,S),S)) \Rightarrow (a,((S,S),S)) \Rightarrow (a,((a,S),S)) \Rightarrow (a,((a,a),S)) \Rightarrow (a,((a,a),(L))) \\ &\Rightarrow (a,((a,a),(L,S))) \Rightarrow (a,((a,a),(S,S))) \Rightarrow (a,((a,a),(a,S))) \Rightarrow (a,((a,a),(a,a))). \end{aligned}$$

$$\begin{aligned} S &\Rightarrow (L) \Rightarrow (L,S) \Rightarrow (L,(L)) \Rightarrow (L,(L,S)) \Rightarrow (L,(L,(L))) \Rightarrow (L,(L,(L,S))) \\ &\Rightarrow (L,(L,(L,a))) \Rightarrow (L,(L,(S,a))) \Rightarrow (L,(L,(a,a))) \Rightarrow (L,(S,(a,a))) \\ &\Rightarrow (L,((L),(a,a))) \Rightarrow (L,((L,S),(a,a))) \Rightarrow (L,((L,a),(a,a))) \Rightarrow (L,((S,a),(a,a))) \\ &\Rightarrow (L,((a,a),(a,a))) \Rightarrow (S,((a,a),(a,a))) \Rightarrow (a,((a,a),(a,a))). \end{aligned}$$

Exercice 3 *Soit la grammaire :*

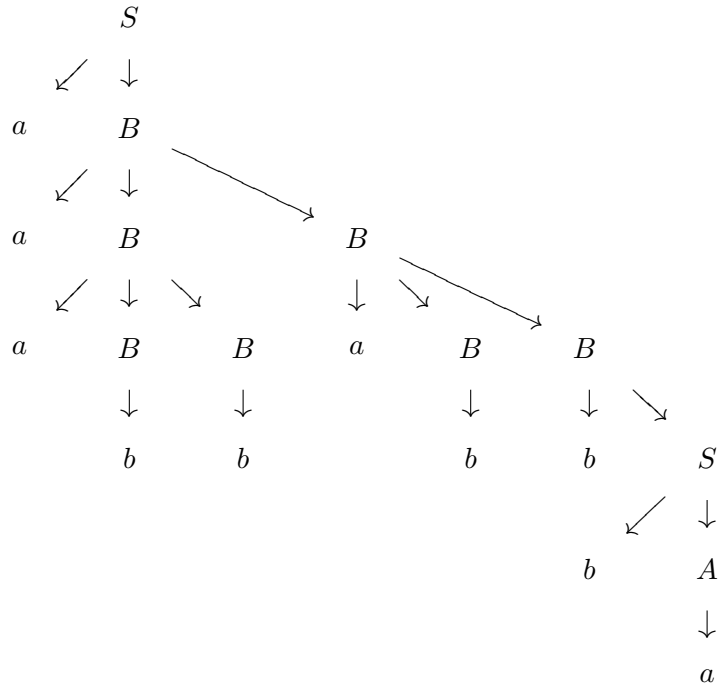
$$\begin{aligned} S &\rightarrow aB \mid bA \\ A &\rightarrow a \mid aS \mid bAA \\ B &\rightarrow b \mid bS \mid aBB \end{aligned}$$

1. *Trouver pour le mot* aaabbabbba *une dérivation à gauche, une dérivation à droite et un arbre de dérivation.*
2. *Montrer par récurrence sur $|w|$ que $L(G)$ est l'ensemble des mots de longueur non nulle qui contiennent autant de a que de b.*

Corrigé :

1. Il y a par exemple :
   – $S \Rightarrow aB \Rightarrow aaBB \Rightarrow aaaBBB \Rightarrow aaabBB \Rightarrow aaaabbB \Rightarrow aaabbaBB \Rightarrow aaabbabB \Rightarrow aaabbabbS \Rightarrow aaabbabbbA \Rightarrow aaabbabbba$

– $S \Rightarrow aB \Rightarrow aaBB \Rightarrow aaBbS \Rightarrow aaBbbA \Rightarrow aaBbba \Rightarrow aaaBBbba \Rightarrow aaaBbbba \Rightarrow aaabSbbba \Rightarrow aaabbAbbba \Rightarrow aaabbabbba$ (Ce n'est pas le même arbre de dérivation que la précédente.)



–

2. Soit L_0 l'ensemble des mots de longueur non nulle contenant autant de a que de b, L_a l'ensemble des mots ayant un a de plus que de b et L_b l'ensemble des mots ayant un b de plus que de a. Nous allons montrer que

$$\begin{cases} (S \stackrel{*}{\Rightarrow} w) & \iff & w \in L_0 \\ (A \stackrel{*}{\Rightarrow} w) & \iff & w \in L_a \\ (B \stackrel{*}{\Rightarrow} w) & \iff & w \in L_b. \end{cases}$$

Si $|w| = 1$, $(w \in L_a) \iff (w = a) \iff (A \stackrel{*}{\Rightarrow} a)$ et $(w \in L_b) \iff (w = b) \iff (B \stackrel{*}{\Rightarrow} b)$.
Si $|w| = 2$, si $w \in L_0$ alors $w = ab$ et $S \Rightarrow aB \Rightarrow ab$ ou bien $w = ba$ et $S \Rightarrow bA \Rightarrow ba$; de même si $S \stackrel{*}{\Rightarrow} w$, on a forcément $w = ab$ ou $w = ba$ et donc $w \in L_0$.
Supposons maintenant $S \stackrel{*}{\Rightarrow} w$. La dérivation commence soit par $S \Rightarrow aB \stackrel{*}{\Rightarrow} w$, soit par $S \Rightarrow bA \stackrel{*}{\Rightarrow} w$. Dans le premier cas, on a $B \stackrel{*}{\Rightarrow} w'$ avec $|w'| < |w|$ et donc $w' \in L_b$ ce qui donne $aw' = w \in L_0$; dans le deuxième cas, on a $A \stackrel{*}{\Rightarrow} w'$ avec $|w'| < |w|$ et donc $w' \in L_a$ ce qui donne $bw' = w \in L_0$. Les cas $A \stackrel{*}{\Rightarrow} w$ et $B \stackrel{*}{\Rightarrow} w$ se traitent de la même manière.
Dans l'autre sens, supposons $w \in L_0$. w est de longueur non nulle, il s'écrit donc $w = aw'$ (resp. $w = bw'$). Dans le premier cas, $w' \in L_b$ (resp. L_a) et donc $B \stackrel{*}{\Rightarrow} w'$ (resp. $A \stackrel{*}{\Rightarrow} w'$). On a alors une dérivation $S \Rightarrow aB \stackrel{*}{\Rightarrow} aw' = w$ (resp. $S \Rightarrow bA \stackrel{*}{\Rightarrow} bw' = w$). Les cas $w \in L_a$ et $w \in L_b$ se traitent pareillement.

Exercice 4 *Quel est le langage reconnu par la grammaire :*

$$\begin{array}{rcl} S & \rightarrow & AB \mid C \\ A & \rightarrow & aAb \mid ab \\ B & \rightarrow & cBd \mid cd \end{array}$$

$$\begin{aligned} C &\rightarrow aCd \mid aDd \\ D &\rightarrow bDc \mid bc \end{aligned}$$

CORRIGÉ : Si on part avec la règle $S \rightarrow AB$, on génère d'abord autant de a que de b, puis autant de c que de d ; le langage est donc $\{a^n b^n c^m d^m\}$.

Si on pars avec la règle $S \rightarrow C$, on génère d'abord autant de a que de d, puis autant de b que de c ; le langage est donc $\{a^n b^m c^m d^n\}$.

Finalement, le langage reconnu est $\{a^n b^n c^m d^m\} \cup \{a^n b^m c^m d^n\}$ pour n,$m \geq 1$.

Exercice 5 *Montrer que la grammaire*

$$S \rightarrow (S)S \mid \varepsilon$$

génère tous les mots de parenthèses équilibrés et « corrects ».

CORRIGÉ :

1. Montrons tout d'abord que tous les mots générés sont équilibrés et corrects (abrévié en corrects). Il suffit de faire une récurrence sur la longueur de la dérivation à gauche qui aboutit à un mot.
   - Au rang 1, $S \Rightarrow \varepsilon$, qui est correct.
   - Soit une dérivation à gauche de n étapes $S \Rightarrow (S)S \stackrel{*}{\Rightarrow} (x)S \stackrel{*}{\Rightarrow} (x)y$. x et y dérivent de S en moins de n étapes, donc ils sont corrects et $(x)y$ l'est également.
2. Montrons maintenant que tous les mots corrects sont générés. Nous allons faire une récurrence sur la longueur du mot.
   - Le seul mot correct de longueur 0 est ε et il est généré.
   - Supposons que tous les mots corrects de longueur inférieure à $2n$ sont générés. Soit un mot de longueur $2n$ correct. Il commence forcément par (. Soit (x) son plus petit préfixe ayant autant de (que de). Le mot s'écrit alors $(x)y$ où x et y sont corrects et de longueur inférieure à $2n$. Donc ils sont tous les deux générés. Autrement dit, $S \stackrel{*}{\Rightarrow} x$ et $S \stackrel{*}{\Rightarrow} y$. Mais alors : $S \Rightarrow (S)S \stackrel{*}{\Rightarrow} (x)S \stackrel{*}{\Rightarrow} (x)y$.

2 Grammaires ambiguës

Exercice 6

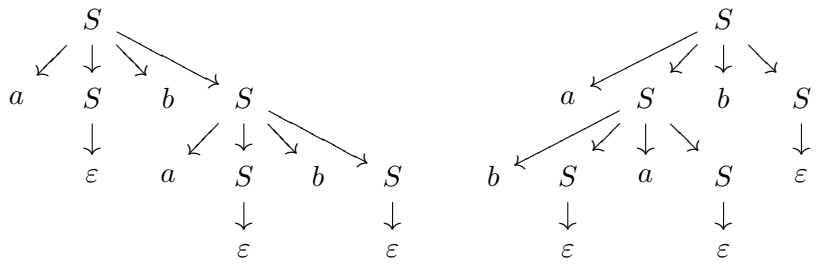
$$S \rightarrow aSbS \mid bSaS \mid \varepsilon$$

Montrer que cette grammaire est ambiguë en générant deux dérivations à gauche pour abab*. Donner deux dérivations à droite de* abab*. Quels sont les arbres de dérivation correspondants ?*

CORRIGÉ : Les dérivations à gauche sont : $S \Rightarrow aSbS \Rightarrow abS \Rightarrow abaSbS \Rightarrow ababS \Rightarrow abab$
et $S \Rightarrow aSbS \Rightarrow abSaSbS \Rightarrow abaSbS \Rightarrow ababS \Rightarrow abab$.

Les dérivations à droite sont : $S \Rightarrow aSbS \Rightarrow aSbaSbS \Rightarrow aSbaSb \Rightarrow aSbab \Rightarrow abab$
et $S \Rightarrow aSbS \Rightarrow aSb \Rightarrow abSaSb \Rightarrow abSab \Rightarrow abab$.

Les arbres sont :



Exercice 7 *Soit la grammaire*

$$R \to R'|'R \mid RR \mid R^* \mid (R) \mid a \mid b$$

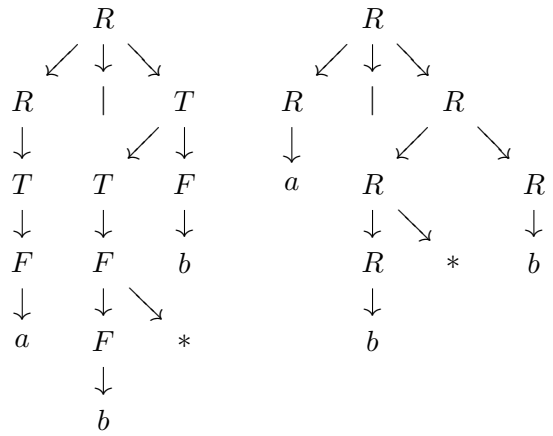
1. *Montrer que cette grammaire génère les expressions régulières sur l'alphabet* $\{a, b\}$.
2. *Montrer que cette grammaire est ambiguë.*
3. *Construire une grammaire équivalente non-ambiguë avec les priorités classiques (∗ puis . puis |) et l'associativité à gauche.*
4. *Construire un arbre de dérivation pour* `a|b*b` *dans les deux grammaires.*

Corrigé :

– Elle est ambiguë car, par exemple, on a $a|b^*$ qui peut être généré de deux façons par une dérivation à gauche : $R \Rightarrow R|R \Rightarrow a|R \Rightarrow a|R^* \Rightarrow a|b^*$ et $R \Rightarrow R^* \Rightarrow R|R^* \Rightarrow a|R^* \Rightarrow a|b^*$.

–

$$\begin{aligned} R &\to R'|'T \mid T \\ T &\to TF \mid F \\ F &\to F^* \mid a \mid b \mid (R) \end{aligned}$$



Exercice 8 *Reprenons la grammaire de l'exercice 4 :*

$$\begin{aligned} S &\to AB \mid C \\ A &\to aAb \mid ab \\ B &\to cBd \mid cd \\ C &\to aCd \mid aDd \\ D &\to bDc \mid bc \end{aligned}$$

Montrer que cette grammaire est ambiguë.

CORRIGÉ : Le langage lui-même est « ambigu » : si $n = m$, les mots $a^n b^n c^m d^m$ et $a^n b^m c^m d^n$ sont identiques.

Ainsi, on a les deux dérivations suivantes :

$$S \to AB \to abB \to abcd$$
$$S \to C \to aDd \to abcd$$

Exercice 9 *On considère la grammaire suivant :*

$$\begin{aligned} S &\to aS \mid bS \mid bA \\ A &\to bA \mid b \end{aligned}$$

1. *Montrer que cette grammaire est ambiguë en produisant* tous *les arbres de dérivation que l'on peut obtenir pour le mot aabbbb.*
2. *Un moyen que nous pourrions utiliser (non vu en cours) pour désambiguïser une grammaire consiste à assigner des probabilités aux règles de production de la grammaire de telle manière que*

$$\sum_{\alpha \in (V \cup T)^+} p(A \to \alpha) = 1$$

avec $(A \to \alpha) \in P$.
On peut alors déterminer la probabilité d'une dérivation en calculant le produit des probabilités assignées aux règles de production utilisées pendant la dérivation. On peut par exemple attribuer les probabilités suivantes aux règles de production de notre grammaire :

$$\begin{aligned} S &\to aS\ (0{,}8) \mid bS\ (0{,}01) \mid bA\ (0{,}19) \\ A &\to bA\ (0{,}9) \mid b\ (0{,}1) \end{aligned}$$

Parmi les dérivations obtenues à la question précédente, déterminer celle qui est désormais la plus probable.

CORRIGÉ :

1. On a les trois dérivations suivantes :

$$S \to aS \to aaS \to aabS \to aabbS \to aabbbA \to aabbbb$$
$$S \to aS \to aaS \to aabS \to aabbA \to aabbbA \to aabbbb$$
$$S \to aS \to aaS \to aabA \to aabbA \to aabbbA \to aabbbb$$

2. Ce qui nous donne les probabilités suivantes :

$$S \overset{0,8}{\to} aS \overset{0,8}{\to} aaS \overset{0,01}{\to} aabS \overset{0,01}{\to} aabbS \overset{0,19}{\to} aabbbA \overset{0,1}{\to} aabbbb$$
$$S \overset{0,8}{\to} aS \overset{0,8}{\to} aaS \overset{0,01}{\to} aabS \overset{0,19}{\to} aabbA \overset{0,9}{\to} aabbbA \overset{0,1}{\to} aabbbb$$
$$S \overset{0,8}{\to} aS \overset{0,8}{\to} aaS \overset{0,19}{\to} aabA \overset{0,9}{\to} aabbA \overset{0,9}{\to} aabbbA \overset{0,1}{\to} aabbbb$$

C'est clairement la dernière dérivation qui est la plus probable.

3 Simplification de grammaires

Exercice 10 *Supprimer (en expliquant la procédure) les symboles inutiles de la grammaire :*

$$\begin{aligned} S &\rightarrow aAAB \mid CC \mid cA \\ A &\rightarrow aA \mid a \\ C &\rightarrow cC \end{aligned}$$

Corrigé : À la première étape, on garde A qui dérive un terminal (a) puis S qui dérive une phrase formée d'un terminal et d'une variable déjà conservée (cA). Cela nous donne la grammaire :

$$\begin{aligned} S &\rightarrow cA \\ A &\rightarrow aA \mid a \end{aligned}$$

La deuxième étape ne permet pas d'éliminer d'autres symboles car A dérive de S. La grammaire ci-dessus est donc une grammaire équivalente sans symboles inutiles.

Exercice 11 *Supprimer (en expliquant la procédure) les ε-productions de la grammaire :*

$$\begin{aligned} S &\rightarrow AB \\ A &\rightarrow aA \mid \varepsilon \\ B &\rightarrow b \mid \varepsilon \end{aligned}$$

Corrigé : A et B sont annulables et donc S également. On obtient donc la grammaire:

$$\begin{aligned} S &\rightarrow AB \mid A \mid B \\ A &\rightarrow aA \mid a \\ B &\rightarrow b \end{aligned}$$

On peut rajouter la règle $S \rightarrow \varepsilon$ si on veut garder la possibilité de générer ε.

Exercice 12 *Supprimer (en expliquant la procédure) les règles unitaires de la grammaire :*

$$\begin{aligned} S &\rightarrow ABc \\ A &\rightarrow B \\ B &\rightarrow B \mid b \mid c \end{aligned}$$

Corrigé : Les règles $S \rightarrow ABc$ et $B \rightarrow b|c$ sont non-unitaires, il faut les garder. Comme $A \stackrel{*}{\Rightarrow} B$ et qu'on a la règle non-unitaire $B \rightarrow b|c$, il faut rajouter la règle $A \rightarrow b|c$. On obtient :

$$\begin{aligned} S &\rightarrow ABc \\ A &\rightarrow b \mid c \\ B &\rightarrow b \mid c \end{aligned}$$

Exercice 13 *Transformer la grammaire suivante en une grammaire équivalente (sans ε) sans symboles inutiles, sans ε-productions et sans règles unitaires.*

$$\begin{aligned} S &\rightarrow AB \mid CA \\ A &\rightarrow a \mid b \mid \varepsilon \\ B &\rightarrow BC \mid DB \\ C &\rightarrow E \mid \varepsilon \\ D &\rightarrow a \mid d \\ E &\rightarrow aB \mid c \mid d \mid \varepsilon \end{aligned}$$

Corrigé :

1. Éliminons d'abord les symboles inutiles. À la première étape, seul B ne permet pas de dériver un terminal ou une phrase composée de terminaux et de variables dérivant une phrase terminale. On peut donc l'éliminer. On obtient donc :

$$\begin{aligned} S &\rightarrow CA \\ A &\rightarrow a \mid b \mid \varepsilon \\ C &\rightarrow E \mid \varepsilon \\ D &\rightarrow a \mid d \\ E &\rightarrow c \mid d \mid \varepsilon \end{aligned}$$

   Pour la deuxième étape, S dérive CA et C dérive E, il faut donc garder A, C, E et S comme variables. Voici notre grammaire sans symboles inutiles :

$$\begin{aligned} S &\rightarrow CA \\ A &\rightarrow a \mid b \mid \varepsilon \\ C &\rightarrow E \mid \varepsilon \\ E &\rightarrow c \mid d \mid \varepsilon \end{aligned}$$

2. A, C, E et S sont annulables, on obtient donc une grammaire équivalente (ne générant pas ε), sans ε-productions :

$$\begin{aligned} S &\rightarrow CA \mid C \mid A \\ A &\rightarrow a \mid b \\ C &\rightarrow E \\ E &\rightarrow c \mid d \end{aligned}$$

3. Nous avons trois règles unitaires : $S \rightarrow C$, $S \rightarrow A$ et $C \rightarrow E$. Par ailleurs on peut dériver $S \overset{*}{\Rightarrow} A$, $S \overset{*}{\Rightarrow} C$, $S \overset{*}{\Rightarrow} E$ et $C \overset{*}{\Rightarrow} E$, donc on obtient la grammaire sans règles unitaires :

$$\begin{aligned} S &\rightarrow CA \mid a \mid b \mid c \mid d \\ A &\rightarrow a \mid b \\ C &\rightarrow c \mid d \\ E &\rightarrow c \mid d \end{aligned}$$

4. On termine en éliminant à nouveau les symboles devenus inutiles : E ne dérive pas de S, donc on obtient la grammaire suivante.

$$\begin{aligned} S &\rightarrow CA \mid a \mid b \mid c \mid d \\ A &\rightarrow a \mid b \\ C &\rightarrow c \mid d \end{aligned}$$

Exercice 14 *On considère la (célèbre) grammaire des expressions arithmétiques bien formées, où n désignera par commodité tout entier positif (traité ici comme un terminal).*

$$\begin{aligned} S &\rightarrow T+S \mid S+T \mid T \\ T &\rightarrow F*T \mid T*F \mid F \\ F &\rightarrow n \mid (S) \end{aligned}$$

Mettre cette grammaire sous forme normale de Chomsky.
Utiliser l'algorithme CYK (Cocke, Younger et Kasami) pour déterminer si le mot $n+(n)$ est généré par cette grammaire.

Corrigé : Pour mettre sous forme normale de Chomsky, on procède d'abord à l'élimination des ϵ-transitions, des productions unités et des symboles inutiles.

La grammaire ne comporte pas de ϵ-transitions. L'élimination des productions unités nous donne la grammaire :

$$\begin{aligned} S &\rightarrow T+S \mid S+T \mid F*T \mid T*F \mid n \mid (S) \\ T &\rightarrow F*T \mid T*F \mid n \mid (S) \\ F &\rightarrow n \mid (S) \end{aligned}$$

La grammaire ne comporte pas de symboles inutiles.

Pour la mise sous forme normale de Chomsky, on introduit d'abord de nouvelles variables pour tous les symboles terminaux qui apparaissent dans les parties droites des règles de longueur plus grande que 2.

$$\begin{aligned} S &\rightarrow TO_1S \mid SO_1T \mid FO_2T \mid TO_2F \mid n \mid P_1SP_2 \\ T &\rightarrow FO_2T \mid TO_2F \mid n \mid P_1SP_2 \\ S &\rightarrow n \mid P_1SP_2 \\ O_1 &\rightarrow + \\ O_2 &\rightarrow * \\ P_1 &\rightarrow (\\ P_2 &\rightarrow) \end{aligned}$$

Puis on scinde de manière à obtenir des règles avec deux symboles, en introduisant de nouveaux symboles (X_i)

$$\begin{aligned} S &\rightarrow TX_1 \mid SX_2 \mid FX_3 \mid TX_4 \mid n \mid P_1X_5 \\ T &\rightarrow FX_3 \mid TX_4 \mid n \mid P_1X_5 \\ F &\rightarrow n \mid P_1X_5 \\ O_1 &\rightarrow + \\ O_2 &\rightarrow * \\ P_1 &\rightarrow (\\ P_2 &\rightarrow) \end{aligned}$$

$$
\begin{aligned}
X_1 &\rightarrow O_1 S \\
X_2 &\rightarrow O_1 T \\
X_3 &\rightarrow O_2 T \\
X_4 &\rightarrow O_2 F \\
X_5 &\rightarrow S P_2
\end{aligned}
$$

On applique ensuite l'algo CYK, et on obtient le tableau suivant:

$\{S\}$				
$\emptyset$	$\{X_1, X_2\}$			
$\emptyset$	$\emptyset$	$\{S, T, F\}$		
$\emptyset$	$\emptyset$	$\emptyset$	$\{X_5\}$	
$\{S, T, F\}$	O_1	P_1	$\{S, T, F\}$	P_2
n	$+$	$($	n	$)$

Donc le mot est bien généré par cette grammaire.