Grammaires formelles et hiérarchie de Chomsky

1 Introduction

On utilise les grammaires formelles pour spécifier la syntaxe des langages.
Une grammaire est un quadruplet (N, T, S, R) où :

— T est un ensemble fini non vide de symboles appelé ensemble de symboles terminaux ou alphabet terminal ;

— N est un ensemble fini de symboles appelé ensemble de symboles non-terminaux ou alphabet non-terminal qui vérifie $T \cap N = \emptyset$;

— S est un symbole faisant partie de N appelé axiome ;

— R est l'ensemble fini des productions de la forme $\alpha \longrightarrow \beta$, $\alpha, \beta \in (N \cup T)^*$ et α contient au moins un non-terminal. Les productions sont encore appelées règles de réécriture du fait qu'elles signifient que la séquence de symboles α peut être remplacée par la séquence de symboles β.

On note en général

- $N = \{A, B, C, \ldots\}$ les non-terminaux
- $T = \{a, b, c, \ldots\}$ les terminaux
- $N \cup T = \{X, Y, Z, \ldots\}$ les symboles terminaux ou non-terminaux
- u, v, w, x, y, z les mots de T^*
- $\alpha, \beta, \gamma, \ldots$ les mots de $(N \cup T)^*$

Les règles $\begin{matrix} \alpha \longrightarrow \beta \\ \alpha \longrightarrow \gamma \end{matrix}$ sont notées plus simplement $\alpha \longrightarrow \beta|\gamma$

2 Dérivation

Un mot ou phrase ou forme syntaxique du langage engendré par la grammaire G est obtenu en partant de l'axiome et en appliquant une suite de productions jusqu'à ce que la chaîne obtenue par les réécritures successives ne contienne que des symboles terminaux. L'application d'une suite de productions est appelée dérivation. On note $\alpha \Rightarrow \beta$ la réécriture de α en β par l'application d'une production et $\alpha \overset{*}{\Rightarrow} \beta$ la réécriture de α en β par l'application d'une suite éventuellement vide de productions. La longueur (ou l'ordre ou le nombre d'étapes) de la dérivation $\alpha \overset{*}{\Rightarrow} \beta$ est le nombre de productions appliquées pour passer de la forme α à la forme β.

Le langage $L(G)$ engendré par une grammaire G est l'ensemble de toutes les phrases possibles, c'est-à-dire l'ensemble de toutes les formes syntaxiques constituées uniquement

de symboles terminaux obtenues par dérivation à partir de l'axiome.
Formellement, on écrit : $L(G) = \{w \in T^* \mid S \overset{*}{\Rightarrow} w\}$.

Deux grammaires G_1 et G_2 sont équivalentes si elles engendrent le même langage. Une grammaire engendre un unique langage. Un langage peut être engendré par plusieurs grammaires.

3 Classification des langages et hiérarchie de Chomsky

En 1957, Noam Chomsky alors qu'il travaillait sur une étude des langues naturelles, a établi la classification suivante des grammaires en fonction de la nature des productions :

- les grammaires de type 0 pour lequel il n'y a aucune restriction sur les productions. Les langages engendrés sont dits récursivement énumérables ;
- les grammaires de type 1 sont appelées grammaires sensibles au contexte. Leurs productions vérifient $|\alpha| \leq |\beta|$. Le cas particulier $S \longrightarrow \varepsilon$ est permis si S n'apparaît pas dans la partie de droite d'une production de G. Les grammaires de type 1 sont dites sensibles au contexte car elles peuvent contenir des productions telles que $aAb \longrightarrow a\beta b$ où le remplacement de A par β se fait uniquement dans le contexte précis où A est entouré par les symboles a et b. Les langages engendrés sont appelés langages sensibles au contexte ;
- les grammaires de type 2 appelées grammaires hors-contexte ou grammaires algébriques sont celles qui nous intéressent principalement car la plupart des langages de programmation sont engendrés par des grammaires qui font partie de cette catégorie. Leurs productions doivent être de la forme $A \longrightarrow \beta$ avec $A \in N$. Les langages engendrés sont appelés langages hors-contexte ou langages algébriques ;
- les grammaires de type 3 sont appelées grammaires rationnelles. Ces grammaires sont linéaires à droite : toutes les productions sont de la forme $A \longrightarrow wB \; ou A \longrightarrow w$
  ou bien linéaires à gauche : toutes les productions sont de la forme $A \longrightarrow Bw$ ou $A \longrightarrow w$.
  Les langages engendrés sont appelés langages réguliers ou langages rationnels.

Une grammaire de type 3 est évidemment une grammaire de type 2 qui est une grammaire de type 1 qui est aussi une grammaire de type 0. De ces quatre types, les plus importants dans le cadre des techniques de compilation sont les types 2 et 3. En effet, une grammaire algébrique peut être utilisée pour spécifier la syntaxe des langages de programmation et les langages rationnels sont utilisés pour construire les analyseurs lexicaux qui permettent au compilateur de découper le programme analysé en une suite de mots du langage.

Un langage engendré par une grammaire de type i est dit langage de type i. Dans chaque type de langage, il existe une machine virtuelle qui permet de tester si un mot donné est dans le langage.

4 Résumé

type	**restriction sur les productions**	**Machines à états**	**Temps de recherche**	**langage grammaire**
0	aucune	Machine de Turing	∞	récursivement énumérable
1	$\|\beta\| \geq \|\alpha\|$	Machine de Turing bornée	k^n	sensible au contexte
2	$A \longrightarrow \alpha$	Automate à pile	n^3	indépendant du contexte (context-free) ou algébrique
3	$A \longrightarrow a$ ou $A \longrightarrow aB$	Automate	n	rationnel
	$A \longrightarrow a$ ou $A \longrightarrow Ba$	Automate	n	rationnel

(Le problème de trouver si un mot est engendré par une grammaire de type 0 n'est pas décidable).