2.0 Rappels sur les grammaires syntagmatiques

Grammaire formelle Pour définir une grammaire formelle, il faut définir
- un alphabet *terminal* (X),
- un alphabet *non terminal* disjoint (V), comprennant l'*axiome* $S \in V$,
- un ensemble de couples de $(X \cup V)^*V(X \cup V)^* \times (X \cup V)^*$, les *règles de production*.

Principe de réécriture Les grammaires formelles constituent un cas particulier de *système de réécriture* (ensemble de règles de réécriture).

Une *règle de réécriture* de la forme $r \to r'$ s'applique à l'objet syntaxique t si celui-ci contient une instance du membre gauche r, c'est-à-dire un sous-objet que l'on peut identifier à r. L'objet t se réécrit alors en un nouvel objet t', obtenu en remplaçant l'instance de r par l'instance du membre droit r' correspondante. Notation : $t \to t'$.

Dérivation Pour les grammaires formelles, on définit ainsi la réécriture (ou dérivation) immédiate (pour une règle $r : A \longrightarrow u$) : $f \xrightarrow{r} g$ ssi $\exists v, w$ t.q. $f = vAw$ et $g = vuw$

La *dérivation* est définie comme la fermeture de la dérivation immédiate :
$f \xrightarrow{\mathcal{G}*} g$ ssi $f = g$, ou
$\exists f_1 = f, f_2, ..., f_n = g,\ \exists r_i \in P$ t.q. $f_{i-1} \xrightarrow{r_i} f_i$

Proto-mot, langage engendré Un mot sur $(X \cup V)^*$ obtenu au cours d'une dérivation et contenant au moins un symbole non terminal est appelé *proto-mot*. L'ensemble des mots (sur X^*) engendrés à partir de l'axiome constitue le *langage engendré* par la grammaire.

Dérivations multiples En général, la suite de réécritures immédiates qui conduit de l'axiome à un mot donné u n'est pas unique. On distingue parmi les suites possibles la *dérivation gauche*, définie de telle sorte que le non-terminal réécrit à chaque étape est le premier non-terminal du proto-mot. Une grammaire telle que pour un mot donné il existe plusieurs dérivations gauches distinctes est une grammaire *ambigüe*.

Pouvoir expressif On peut distinguer des sous-classes de grammaires formelles, sur la base de la forme des règles de production. Ces classes de grammaires définissent à leur tour des classes de langages formels. Chomsky a proposé une hiérarchie de classes de grammaires/langages, dont les classes sont incluses les unes dans les autres.

Langages	type	règles	machine
récursivement énumérables	0	*pas de contraintes*	Machine de Turing
contextuels	1	$\alpha A\beta \to \alpha\gamma\beta$, avec $A \in V, \alpha, \beta, \gamma \in (X \cup V)^*, \gamma \neq \varepsilon$	Automates à pile linéairement bornée
algébriques	2	$A \to \gamma$, avec $A \in V, \gamma \in (X \cup V)^*$	Automates à pile
rationnels	3	$A \to Ba$, $A \to a$, avec $A, B \in V, a \in X$	Automates à nombre fini d'états

Arbre de dérivation Pour les **grammaires algébriques**, il existe on peut représenter l'ensemble des dérivations équivalentes par un *arbre de dérivation*.

$$\mathcal{G}_0 = \begin{array}{lcl} S & \rightarrow & TZ \\ T & \rightarrow & 0T1C \\ & \mid & \varepsilon \\ C1 & \rightarrow & 1C \\ CZ & \rightarrow & Z2 \\ 1Z & \rightarrow & 1 \end{array} \qquad \begin{array}{c} \underline{\mathtt{S}} \\ \underline{\mathtt{T}}\mathtt{Z} \\ \mathtt{0}\underline{\mathtt{T}}\mathtt{1CZ} \\ \mathtt{00}\underline{\mathtt{T}}\mathtt{1C1CZ} \\ \mathtt{00}\varepsilon\mathtt{1}\underline{\mathtt{C1}}\mathtt{CZ} \\ \mathtt{0011C}\underline{\mathtt{CZ}} \\ \mathtt{0011}\underline{\mathtt{CZ}}\mathtt{2} \\ \mathtt{001}\underline{\mathtt{1Z}}\mathtt{22} \\ \mathtt{001122} \end{array} \qquad \begin{array}{c} \underline{\mathtt{S}} \\ \underline{\mathtt{T}}\mathtt{Z} \\ \mathtt{0T1}\underline{\mathtt{CZ}} \\ \mathtt{0}\underline{\mathtt{T}}\mathtt{1Z2} \\ \mathtt{00}\underline{\mathtt{T}}\mathtt{1C1Z2} \\ \mathtt{00}\varepsilon\mathtt{1}\underline{\mathtt{C1}}\mathtt{Z2} \\ \mathtt{0011}\underline{\mathtt{CZ}}\mathtt{2} \\ \mathtt{001}\underline{\mathtt{1Z}}\mathtt{22} \\ \mathtt{001122} \end{array} \qquad L_{G_0} = 0^i1^i2^i$$

$$\mathcal{G}_1 = \left\langle \{jean, dort\}, \{Np, SN, SV, V, S\}, S, \left\{ \begin{array}{l} S \rightarrow SN\ SV \\ SN \rightarrow Np \\ SV \rightarrow V \\ Np \rightarrow jean \\ V \rightarrow dort \end{array} \right\} \right\rangle\}$$

$\mathcal{G}_2 = \langle\{(,)\}, \{S\}, S, \{S \longrightarrow \varepsilon \mid (S)S\}\rangle$

$G_3 = E \rightarrow E + T \mid T, T \rightarrow T \times F \mid F, F \rightarrow (E) \mid a$

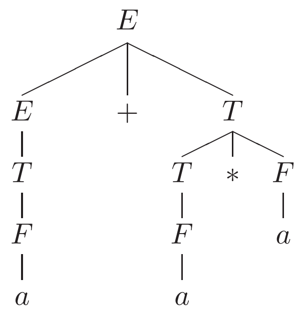
Voici un exemple de dérivation du mot $a + a * a$ par la grammaire $\mathcal{G}_2$ (8 étapes) :

$E \Rightarrow E{+}T \Rightarrow E{+}T{*}F \Rightarrow T{+}T{*}F \Rightarrow T{+}F{*}F \Rightarrow T{+}a{*}F \Rightarrow F{+}a{*}F \Rightarrow a{+}a{*}F \Rightarrow a{+}a{*}a$

Dérivation gauche :

$E \Rightarrow E + T \Rightarrow T + T \Rightarrow F + T \Rightarrow a + T \Rightarrow a + T * F \Rightarrow a + F * F \Rightarrow a + a * F \Rightarrow a + a * a$

Factorisation par un arbre de dérivation :



$G_2' = \langle\{E\}, \{a, +, \times\}, E, \{E \rightarrow E + E \mid E \times E \mid a\}\rangle$

Deux arbres $\neq$ pour $a + a * a$: