Langage de Dyck $\mathfrak{D}_n$

$$\Sigma = \{a_1, \overline{a_1}, \ldots, a_n, \overline{a_n}\},$$

$G = (V, \Sigma, P, S)$ avec $V = \{S, T\}$ et les productions de P sont

$$\begin{array}{lcl} S & \rightarrow & ST \mid \varepsilon \\ T & \rightarrow & a_1\ S\ \overline{a_1} \mid \cdots \mid a_n\ S\ \overline{a_n}\ . \end{array}$$

$$\begin{array}{lcl} S & \rightarrow & ST \mid \varepsilon \\ T & \rightarrow & (\ S\ ) \mid [\ S\ ]. \end{array}$$

$$\underline{S} \Rightarrow S\underline{T} \Rightarrow \underline{S}(S) \Rightarrow (\underline{S}) \Rightarrow (\,),$$

$$\begin{aligned} \underline{S} &\Rightarrow S\underline{T} \Rightarrow \underline{S}(S) \Rightarrow ST(\underline{S}) \Rightarrow ST(\,) \\ &\Rightarrow S\underline{T}(\,) \Rightarrow S[S](\,) \Rightarrow S[\underline{S}](\,) \Rightarrow \underline{S}[\,](\,) \Rightarrow [\,](\,), \end{aligned}$$

$$\begin{aligned} \underline{S} &\Rightarrow S\underline{T} \Rightarrow S(\underline{S}) \Rightarrow S(S\underline{T}) \Rightarrow S(S(\underline{S})) \\ &\Rightarrow S(S(\underline{S}T)) \Rightarrow S(S(S\underline{T}T)) \Rightarrow^* ((()())). \end{aligned}$$

Remarque

w appartient à $\mathfrak{D}_1$ si et seulement si les deux conditions suivantes sont satisfaites

- $|w|_a = |w|_{\overline{a}}$
- pour tout préfixe u de w, $|u|_a \geq |u|_{\overline{a}}$.

PROPOSITION

$$\{\text{langages réguliers}\} \subsetneq \{\text{langages algébriques}\}$$

Rappel : L'ensemble des langages réguliers sur Σ est la plus petite famille de langages contenant $\emptyset$, $\{\sigma\}$, $\sigma \in \Sigma$, et stable pour l'union, la concaténation et l'étoile de Kleene.

Définition

Une grammaire hors contexte $G = (V, \Sigma, P, S)$ est *régulière* (*à gauche*) si toute production de G possède une des trois formes suivantes :

- $A \to a$
- $A \to Ba$
- $A \to \varepsilon$

où A, B appartiennent à V et a à Σ.

Proposition (admis)

Un langage est régulier si et seulement si il est généré par une grammaire régulière à gauche (resp. à droite).

Grammaire de type 0

Une grammaire $G = (V, \Sigma, P, S)$ de type **0**, ou *grammaire non restrictive*, est la forme la plus générale de grammaire.
Une production de la forme $u \to v$ précise qu'une occurence du mot $u \neq \varepsilon$ peut être remplacée par v, avec $u, v \in (V \cup \Sigma)^*$.

L est généré par une grammaire non restrictive si et seulement si L est récursivement énumérable (i.e., accepté par une machine de Turing).

EXEMPLE

La grammaire non restrictive $G = (V, \Sigma, P, S)$ telle que $V = \{S, A, C\}$, $\Sigma = \{a, b, c\}$ et dont les règles sont données par

$$\begin{array}{rcl} S & \to & aAbc \mid \varepsilon \\ A & \to & aAbC \mid \varepsilon \\ Cb & \to & bC \\ Cc & \to & cc \end{array}$$

génère le langage $\{a^n b^n c^n \mid n \in \mathbb{N}\}$. En effet,

$$\begin{array}{rcl} S & \Rightarrow & aAbc \Rightarrow aaAbCbc \Rightarrow^* a(a)^i A(bC)^i bc \\ & \Rightarrow^* & a(a)^i (bC)^i bc = (a)^{i+1} b(Cb)^i c \\ & \Rightarrow^* & (a)^{i+1}(b)^{i+1} C^i c \Rightarrow^* (a)^{i+1}(b)^{i+1}(c)^{i+1}. \end{array}$$

CONTEXT-SENSITIVE LANGUAGE

Une grammaire non restrictive $G = (V, \Sigma, P, S)$ est de type **1**, si toutes les productions $u \to v$ de G satisfont

- $u, v \in (V \cup \Sigma)^* \setminus \{\varepsilon\}$
- $|u| \leq |v|$.

Avec cette dernière condition, on parle de *grammaire non contractante* ou *monotone* car la longueur des mots produits croît à chaque application d'une nouvelle règle. On autorise de plus une unique règle de la forme $S \to \varepsilon$.

Une définition équivalente de grammaire dépendant du contexte $G = (V, \Sigma, P, S)$ est de spécifier les productions de P sous la forme

$$\alpha N \beta \to \alpha v \beta$$

où $\alpha, \beta \in (V \cup \Sigma)^*$, $N \in V$, $v \in (V \cup \Sigma)^* \setminus \{\varepsilon\}$.

Langage $\{a^n b^n c^n \mid n \geq 0\}$

$$\begin{array}{rcl} S & \to & aAbc \mid abc \\ A & \to & aAbC \mid abC \\ Cb & \to & bC \\ Cc & \to & cc. \end{array}$$

Grammaire monotone dépendant du contexte.

Les langages générés par une grammaire dépendant du contexte sont exactement les langages décidés par les machines de Turing dont la mémoire disponible est bornée de manière linéaire par la taille des données.

Hiérarchie de Chomsky

	générateur	langage	accepteur
0	grammaire non restrictive	récursivement énumérable	machine de Turing
1	grammaire dépendant du contexte	dépendant du contexte	machine de Turing à mémoire linéaire
2	grammaire hors contexte	hors contexte	automates à pile
3	expression régulière	régulier	AFD

Définitions

les grammaires régulières sont des cas particuliers de grammaire linéaire dont les seconds membres des productions appartiennent tous à $\Sigma^* \cup \Sigma^* V \Sigma^*$.

$$\boxed{\text{Reg} \subset \text{Lin} \subset \text{Alg} \subset \text{DP} \subset \text{RE}}$$

RÉDUCTIONS ET FORMES NORMALES

Elimination des règles $A \to \varepsilon$
Les "ε-productions" font grossir inutilement la longueur des mots intermédiaires produits.

EXEMPLE

$$\begin{array}{lcl} S & \to & SaB \mid aB \\ B & \to & bB \mid \varepsilon. \end{array}$$

La dérivation à gauche générant le mot *aaa* génère trois B qui seront chacun éliminés par l'application de la règle $B \to \varepsilon$. Ainsi, on a

$$S \Rightarrow SaB \Rightarrow SaBaB \Rightarrow aBaBaB \Rightarrow aaBaB \Rightarrow aaaB \Rightarrow aaa.$$

Réductions et formes normales

On appelle variable effaçable toute variable A telle que

$$A \Rightarrow^* \varepsilon.$$

Si une grammaire ne contient aucune variable effaçable, alors $u \Rightarrow v$ entraîne $|u| \leq |v|$. On est dès lors en présence d'une grammaire monotone.

Détecter les variables effaçables

$$E_0 = \{A \in V \mid A \to \varepsilon \in P\}.$$

Si $E_0 = \emptyset$, l'algorithme s'achève et la grammaire ne possède aucune variable effaçable. Sinon, pour $i \geq 0$, on définit

$$E_{i+1} = E_i \cup \{A \in V \mid \exists w \in E_i^* : A \to w \in P\}.$$

Puisque V est fini, la suite des E_i se stabilise.
Condition d'arrêt : tester l'égalité $E_i = E_{i+1}$?

PROPOSITION

Soit $G = (V, \Sigma, P, S)$ une grammaire hors contexte. Il existe une grammaire $G' = (V', \Sigma', P', S')$ que l'on peut construire effectivement telle que

- G et G' sont équivalentes,
- S' n'apparaît dans aucun second membre des productions de G',
- si ε appartient à $L(G) = L(G')$, alors la seule variable effaçable est S'. Sinon, G' ne contient aucune variable effaçable.

Idée de la preuve

- Chirurgie en introduisant si nécessaire S'
- Détecter l'ensemble E des variables effaçables
- Toute règle $A \to w_1 A_1 w_2 A_2 \cdots w_n A_n w_{n+1}$ avec $A \in V$, $A_1, \ldots, A_n \in E$, $w_1, \ldots, w_{n+1} \in ((V \cup \Sigma) \setminus E)^*$ est remplacée par les règles

$$A \to w_1 x_1 w_2 x_2 \cdots w_n x_n w_{n+1}$$

où chaque x_i peut prendre la valeur A_i ou ε.

- Supprimer (de façon récursive) les règles de la forme $A \to \varepsilon$.

EXEMPLE

$$
\begin{array}{rcl}
S & \rightarrow & ACA \\
A & \rightarrow & aAaD \mid B \mid C \\
B & \rightarrow & bB \mid b \\
C & \rightarrow & cS \mid \varepsilon \\
D & \rightarrow & \varepsilon.
\end{array}
$$

introduire une nouvelle variable S', les règles deviennent

$$
\begin{array}{rcl}
S' & \rightarrow & S \\
S & \rightarrow & ACA \\
A & \rightarrow & aAaD \mid B \mid C \\
B & \rightarrow & bB \mid b \\
C & \rightarrow & cS \mid \varepsilon \\
D & \rightarrow & \varepsilon.
\end{array}
$$

$$
\begin{array}{rcl}
S' & \to & S \\
S & \to & ACA \\
A & \to & aAaD \mid B \mid C \\
B & \to & bB \mid b \\
C & \to & cS \mid \varepsilon \\
D & \to & \varepsilon.
\end{array}
$$

Appliquons l'algorithme de recherche des variables effaçables. On trouve

$$E_0 = \{C, D\},\ E_1 = \{A, C, D\},\ E_2 = \{S, A, C, D\}$$

$$E_3 = \{S', S, A, C, D\} = E.$$

$$
\begin{array}{rcl}
S' & \to & S \mid \varepsilon \\
S & \to & ACA \mid CA \mid AA \mid AC \mid A \mid C \mid \varepsilon \\
A & \to & aAaD \mid B \mid C \mid aAa \mid aaD \mid aa \mid \varepsilon \\
B & \to & bB \mid b \\
C & \to & cS \mid c \mid \varepsilon \\
D & \to & \varepsilon.
\end{array}
$$

Il ne reste plus qu'à éliminer les ε-productions. En particulier, puisque D est le premier membre de l'unique règle $D \to \varepsilon$, on peut supprimer D de tous les seconds membres. On a

$$
\begin{array}{rcl}
S' & \to & S \\
S & \to & ACA \mid CA \mid AA \mid AC \mid A \mid C \\
A & \to & aAa \mid B \mid C \mid aa \\
B & \to & bB \mid b \\
C & \to & cS \mid c.
\end{array}
$$

Enfin, il suffit d'ajouter la règle $S' \to \varepsilon$.