Chapitre 9

Grammaires non contextuelles

Hugo Salou MPI*

Dernière mise à jour le 31 mars 2023

Table des matières

Ce chapitre se rattache à la branche de l'informatique des langages formels. On l'a étudié au chapitre 1 avec les automates, et au chapitre 4 avec les machines. Pour le moment, nous avons 5 classes de langages : (1) les langages finis, (2) les langages locaux, (3) les langages réguliers, (4) les langages décidables en temps polynomial, (5) les langages décidables. L'objectif de ce chapitre se situe entre les points (3) et (4).

Intéressons-nous à un langage particulier, le langage des programmes OCaml avec une syntaxe valide. Ce langage est il régulier ? Non. On peut considérer l'expression

$$e_n = \text{« } \underbrace{((\cdots(}_{n}0\underbrace{)\cdots))}_{n} \text{ »}$$

qui est une expression OCaml valide. Par application du lemme de l'étoile, il n'est pas reconnaissable par un automate à N états en considérant e_{N+1}.

L'ensemble $\mathcal{B}$ des mots bien parenthésés est le plus petit ensemble tel que

— $\varepsilon \in \mathcal{B}$
— si $u \in \mathcal{B}$ et $v \in \mathcal{B}$, alors $u \cdot v \in \mathcal{B}$
— si $u \in \mathcal{B}$, alors $(\cdot u \cdot) \in \mathcal{B}$.

Une telle définition de langage est appelée une grammaire. Un autre exemple de langage est la grammaire de la langue française :

— phrase : sujet + verbe + complément,
— complément : COD + complément,
— complément : CCL + complément,
— complément : ε.

Avec cette définition [1], on peut reconnaître des phrases simples comme

$$\text{« } \underbrace{\text{Matthieu}}_{\text{sujet}}\ \underbrace{\text{aime}}_{\text{verbe}}\ \underbrace{\text{les trains}}_{\text{complément}}\text{. »}$$

1 Définition, vocabulaire, propriétés

1.1 Grammaires non contextuelles

Définition : On se munit d'un alphabet Σ qu'on appelle *terminaux*. On se munit d'un ensemble de symboles $\mathcal{V}$ qu'on appelle *non-terminaux*. On suppose $\mathcal{V} \cap \Sigma$.

Définition : On appelle *règle de production* la donnée

— d'un symbole $V \in \mathcal{V}$,
— d'un mot $w_1 w_2 \dots w_n$ sur l'alphabet $\mathcal{V} \cup \Sigma$,

que l'on note $V \to w_1 w_2 \dots w_n$.

Définition (Grammaire non contextuelle) **:** Une *grammaire non contextuelle* est la donnée de

— un alphabet de non-terminaux $\mathcal{V}$,
— un alphabet de terminaux Σ, [2]
— un ensemble fini de règles de production P,
— un symbole initial $S \in \mathcal{V}$,

1. On néglige les règles manquantes à cette définition de la grammaire française.

que l'on note $(\mathcal{V}, \Sigma, P, S)$.

Interlude. Avec cette définition, on espère pouvoir appliquer ces règles pour définir des mots valides. Par exemple,

$$\mathrm{B} \overset{?}{\rightsquigarrow} \mathrm{BB} \overset{?}{\rightsquigarrow} (\mathrm{B})\mathrm{B} \overset{?}{\rightsquigarrow} (\mathrm{B})(\mathrm{B}) \overset{?}{\rightsquigarrow} (\mathrm{BB})(\mathrm{B}) \overset{?}{\rightsquigarrow} (\mathrm{BB})() \overset{?}{\rightsquigarrow} (\mathrm{B})() \overset{?}{\rightsquigarrow} ((\mathrm{B}))() \overset{?}{\rightsquigarrow} (())().$$

C'est l'objectif de la sous-section suivante.

1.2 Dérivation

Définition (Dérivation immédiate) **:** Soit $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$ une grammaire. Soient u et v deux mots de $(\Sigma \cup \mathcal{V})^\star$. On dit que v *dérive immédiatement* de u dans la grammaire $\mathcal{G}$, que l'on note $u \Rightarrow v$, si

— il existe x et y deux mots de $(\Sigma \cup \mathcal{V})^\star$

— il existe $(V \to w_1 w_2 \dots w_n) \in P$

tels que $u = x \cdot V \cdot y$ et $v = x \cdot w_1 w_2 \dots w_n \cdot y$.

Lemme (de composition) **:** Soit $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$ une grammaire. Soient $u, v \in (\Sigma \cup \mathcal{V})^\star$ tels que $u \Rightarrow v$. Soit $w \in (\Sigma \cup \mathcal{V})^\star$. On a $u \cdot w \Rightarrow v \cdot w$ et $w \cdot u \Rightarrow w \cdot v$.

■

On propose trois définitions différentes mais équivalentes pour la dérivation $\overset{\star}{\Rightarrow}$.

Définition : Étant donné une grammaire $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$, on étend $\Rightarrow$ en sa cloture réflexive et transitive, que l'on note $\overset{\star}{\Rightarrow}$, appelé *dérivation*.

Définition : On définit $\overset{\star}{\Rightarrow}$ comme $u \overset{\star}{\Rightarrow} v \overset{\text{def.}}{\iff} \exists n \in \mathbb{N},\ \exists (u_0, u_1, \dots, u_n) \in \big((\mathcal{V} \cup \Sigma)^\star\big)^{n+1}$ tels que $u_0 = u$, $u_n = v$ et $\forall i \in [\![0, n-1]\!],\ u_i \Rightarrow u_{i+1}$.

Définition : Soit la suite $(\Rightarrow^n)_{n \in \mathbb{N}}$ définie inductivement par

— $u \Rightarrow^0 v \overset{\text{def.}}{\iff} u = v$,

— $u \Rightarrow^{n+1} v \overset{\text{def.}}{\iff} u \Rightarrow^n v$ ou $\exists w \in (\Sigma \cup \mathcal{V})^\star,\ u \Rightarrow w$ et $w \Rightarrow^n v$.

On pose alors $\overset{\star}{\Rightarrow} = \bigcup_{n \in \mathbb{N}} \Rightarrow^n$.

Lemme (de composition*) **:** Soit $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$ une grammaire, et soient $u, v \in (\Sigma \cup \mathcal{V})^\star$ tels que $u \Rightarrow^p v$, pour $p \in \mathbb{N}$. Et, soient $(w, t) \in (\Sigma \cup \mathcal{V})^2$ tels que $w \Rightarrow^q t$, pour $q \in \mathbb{N}$. Alors, $uw \Rightarrow^{p+q} vt$.

2. avec "terminaux/non-terminaux" vient l'implication que $\mathcal{V} \cap \Sigma = \varnothing$

■

Définition : Soit $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$ une grammaire. Son langage est défini par

$$\mathcal{L}(\mathcal{G}) = \{u \in \Sigma^\star \mid S \overset{\star}{\Rightarrow} u\}.$$

Interlude : grammaires contextuelles. Dans une grammaire contextuelle, une règle de production peut-être de la forme $b\mathrm{B} \to a\mathrm{B}a\mathrm{B}$, où $\Sigma = \{a, b\}$. Ces règles dépendent du contexte, et non juste des symboles.

Lemme (de décomposition) **:** Étant donnée une grammaire non contextuelle $(\mathcal{V}, \Sigma, P, S)$, soit $w = w_1 \dots w_n$ un mot de n lettres tel qu'il existe $p \in \mathbb{N}$ et $v \in (\Sigma \cup \mathcal{V})^\star$ tel que $w \Rightarrow^p v$ alors il existe $\tilde{v}_1, \tilde{v}_2, \dots, \tilde{v}_n \in (\Sigma \cup \mathcal{V})^\star$ et $p_1, \dots, p_n \in \mathbb{N}$ tels que $w_1 \Rightarrow^{p_1} \tilde{v}_1$, $w_2 \Rightarrow^{p_2} \tilde{v}_2$, …, $w_n \Rightarrow^{p_n} \tilde{v}_n$, et $v = \tilde{v}_1 \cdots \tilde{v}_n$, et $\sum_{i=1}^{n} p_i = p$.

■

1.3 Preuves par induction

Propriété (principe d'induction) **:** Étant donnée une grammaire $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$, et un non-terminal $V \in \mathcal{V}$, on note localement $V_\downarrow = \{w \in \Sigma^\star \mid V \overset{\star}{\Rightarrow} w\}$.[3] Soit alors un ensemble de propriétés $\mathcal{P}_V$ pour $V \in \mathcal{V}$, tel que, $\forall (V \to w_1 w_2 \dots w_n) \in P$, $\forall (\hat{w}_1, \hat{w}_2, \dots, \hat{w}_m) \in (\Sigma^\star)^m$

$$\left(\forall i \in [\![1, m]\!], \begin{cases} w_i \in \Sigma & \implies \hat{w}_i = w_i \\ w_i \in \mathcal{V} & \implies \hat{w}_i \text{ vérifie } \mathcal{P}_{w_i} \end{cases}\right) \implies \hat{w}_i \dots \hat{w}_n \text{ vérifie } \mathcal{P}_V,$$

alors pour tout $V \in \mathcal{V}$, $V_\downarrow$ vérifie $\mathcal{P}_V$.

1.4 Définitions équivalentes

Comment représenter une dérivation en machine ? On considère les règles de production $\mathrm{X} \to \mathrm{XX}$ et $\mathrm{X} \to a$. On peut, par exemple, représenter une dérivation par un ensemble de possibilités :

$$\mathrm{X} \Rightarrow \mathrm{XX} \Rightarrow \{a\mathrm{X}, \mathrm{X}a, \mathrm{XXX}\} \Rightarrow \cdots.$$

Mais, avec une telle définition, il y a explosions du nombre de possibilités.

Définition (Dérivation immédiatement gauche (resp. droite)) **:** Étant donnée une grammaire $\mathcal{G} = (\mathcal{V}, \Sigma, P, I)$, et étant donnés deux mots u et v de $(\Sigma \cup \mathcal{V})^\star$, on dit que u *dérive immédiatement à gauche* (resp. droite) de u dès lors qu'il existe $x \in \Sigma^\star$, $y \in (\Sigma \cup \mathcal{V})^\star$ (resp. $x \in (\Sigma \cup \mathcal{V})^\star$ et $y \in \Sigma^\star$), et $(V \to w_1 \dots w_n) \in P$ tels que $u = xVy$ et $v = x \cdot w_1 w_2 \dots w_n \cdot y$. On note alors $u \Rightarrow_{\mathrm{g}} v$ (resp. $u \Rightarrow_{\mathrm{d}} v$). On définit, de la même

3. Avec cette définition, $\mathcal{L}(\mathcal{G}) = S_\downarrow$.

manière que pour la dérivation simple, $\overset{\star}{\Rightarrow}_{\mathrm{g}}$ et $\overset{\star}{\Rightarrow}_{\mathrm{d}}$.

Définition (Arbre de dérivation) **:** Étant donnée une grammaire $\mathcal{G} = (\mathcal{V}, \Sigma, P, S)$, on appelle *arbre de dérivation* un arbre dont les nœuds sont étiquetés par des éléments de $\{\varepsilon\} \cup \mathcal{V} \cup \Sigma$ et tel que

- tout nœud interne (ayant des fils) est étiquetés par un élément de $\mathcal{V}$,
- la racine est étiquetée par S,
- tout nœud interne ayant une étiquette V et ayant des fils $T_1, \dots, T_n$ où $n \neq 0$ dont les racines sont étiquetées par $w_1, \dots, w_n$ avec $(V \to w_1 \dots w_n) \in P$.
- tout nœud interne d'étiquette V ayant pour unique fils l'arbre feuille réduit à ε et tel que $(V \to \varepsilon) \in P$.

Dans la suite du chapitre, on fixe $\mathcal{G} = (\mathcal{V}, \Sigma, I, P)$ une grammaire.

Définition (« production » d'un arbre) **:** On définit inductivement la fonction prod de l'ensemble des arbres de la grammaire $\mathcal{G}$ vers $(\Sigma \cup \mathcal{V})^\star$, comme

- $\mathrm{prod}\big(\mathrm{Leaf}(x)\big) = x$,
- $\mathrm{prod}\big(\mathrm{Node}(\_, [T_1, \dots, T_n])\big) = \mathrm{prod}(T_1) \cdot \mathrm{prod}(T_2) \cdot \dots \cdot \mathrm{prod}(T_n)$.

Remarque (Notation) :
On dit qu'un arbre de dérivation T est « *clos* » lorsque $\mathrm{prod}(T) \in \Sigma^\star$.

Propriété : Étant donné $w \in \Sigma^\star$, il est équivalent de dire que

(1) $w \in \mathcal{L}(\mathcal{G})$,
(2) $S \overset{\star}{\Rightarrow}_{\mathrm{g}} w$
(3) $S \overset{\star}{\Rightarrow}_{\mathrm{d}} w$
(4) il existe un arbre de dérivation T dont w est le produit.

■

Définition : Une grammaire est dite *ambigüe* s'il existe un mot w de son langage admettant au moins deux arbres de dérivations.

Interlude : langages algébriques. Les langages reconnus par des grammaires non-contextuelles sont des solutions d'équations polynômiales, où la multiplication correspond à la concaténation et l'addition correspond à l'union. D'où le terme langage *algébrique*.

Définition : Deux grammaires $\mathcal{G}_1$ et $\mathcal{G}_2$ sont dites *faiblement équivalentes* dès lors que $\mathcal{L}(\mathcal{G}_1) = \mathcal{L}(\mathcal{G}_2)$.

2 La hiérarchie de Chomsky

Le terme « *hiérarchie de Chomsky* » n'est pas au programme. Dans cette partie, on traite des inclusions avec les autres familles des langages au programme.

2.1 Avec les langages réguliers

Propriété : Soient $\mathcal{G}_1$ et $\mathcal{G}_2$ des grammaires non contextuelles. Alors,

1. $\mathcal{L}(\mathcal{G}_1) \cup \mathcal{L}(\mathcal{G}_2)$ est reconnu par une grammaire non-contextuelle ;
2. $\mathcal{L}(\mathcal{G}_1) \cdot \mathcal{L}(\mathcal{G}_2)$ est reconnu par une grammaire non-contextuelle ;
3. $(\mathcal{L}(\mathcal{G}_1))^\star$ est reconnu par une grammaire non-contextuelle.

■

Théorème : Tout langage régulier est reconnu par une grammaire non contextuelle.

■

Remarque :
L'inclusion précédente est stricte. En effet le langage $\{a^n \cdot b^n \mid n \in \mathbb{N}\}$ est reconnu par une grammaire non contextuelle mais n'est pas un langage régulier.

Remarque (Digression) :

« Tout langage est-il le langage d'une grammaire non contextuelle ? »

On procède par un argument de taille d'ensembles. Soit $\mathcal{G} = (\mathcal{V}, \Sigma, P, I)$ une grammaire. On considère $\Sigma = \{0, 1\}$. On pose $|\mathcal{G}| = |\Sigma| + |\mathcal{V}| + \sum_{(V \to w_1 \dots w_n) \in P} (n + 1) + 1$. On considère $\mathbb{G}_n = \{\mathcal{G} \mid |\mathcal{G}| = n\}$. L'ensemble $\mathbb{G} = \bigcup_{n \in \mathbb{N}} \mathbb{G}_n$ est dénombrable comme union dénombrable d'ensembles finis. Montrons qu'il existe une bijection entre $\wp(\{0,1\}^\star)$ et $[0, 1[$. À tout $x \in [0, 1[$, on peut poser $x = 0, x_1 x_2 \dots x_n \dots$. D'où,

$$[0, 1[\xleftrightarrow[\text{encodage binaire}]{\text{bijection}} (\mathbb{N} \to \{0, 1\}) \xleftrightarrow[\mathbb{1}]{\text{bijection}} \wp(\mathbb{N}) \xleftrightarrow[\text{encodage binaire}]{\text{bijection}} \wp(\{0, 1\}^\star).$$

2.2 Lien avec les langages décidables

Propriété : Les langages des grammaires non contextuelles sont des langages décidables. □

3. Deux grammaires sont fortement équivalentes si elles produisent les mêmes arbres de dérivation.

La preuve de cette propriété est dans le TD 15. (On peut montrer, par programmation dynamique, que l'appartenance d'un mot à une grammaire non contextuelle est calculable en temps polynômial, en $\mathcal{O}(n^3)$.)