Chapitre 9 : Grammaires non contextuelles

January 29, 2025

1 Grammaire non contextuelle (ou hors contexte)

1.1 Vocabulaire

Définition 9.1 - *grammaire au sens général, grammaire de type 0*

Une *grammaire* est défini par un quadruplet (Σ, V, P, S) où :

- Σ est un alphabet fini de *symboles terminaux*, dit aussi *alphabet terminal*
- V est un alphabet fini de *symboles non terminaux* (ou *variables*), dit aussi *alphabet non terminal*
- $P \subset (\Sigma \cup V)^* \times (\Sigma \cup V)^*$ est un ensemble de *règles de production*.
  Une règle de production $(w_1, w_2) \in P$, notée $w_1 \to w_2$ est un couple de mots écrits avec des symboles terminaux et non terminaux.
- $S \in V$ est un symbole non terminal avec un statut particulier de *symbole initial* (ou *axiome*, *variable initiale*)

Une grammaire sans propriété particulière est dite *type 0*.

Remarque 9.2 - *grammaires*

On note usuellement par des majuscules les symboles non terminaux, et en minuscule les terminaux.

Exemple 9.3 - *de grammaire de type 0*

Pour $\Sigma = \{\texttt{a}\}$, $S = \texttt{S}$, $V = \{S, \texttt{D}, \texttt{F}, \texttt{X}, \texttt{Y}, \texttt{Z}\}$ et $P = \{\texttt{S} \to \texttt{DXaD}$, $\texttt{Xa} \to \texttt{aaX}$, $\texttt{XF} \to \texttt{YF}$, $\texttt{aY} \to \texttt{Ya}$, $\texttt{DY} \to \texttt{DX}$, $\texttt{XZ} \to \texttt{Z}$, $\texttt{aZ} \to \texttt{Za}$, $\texttt{DZ} \to \epsilon\}$, $G = (\Sigma, V, P, S)$ est une grammaire de type 0.

Définition 9.4 - *dérivabilité immédiate*

Soit $G = (\Sigma, V, P, S)$ une grammaire. Soit α et β deux mots de $(\Sigma \cup V)^*$. On dit que α *se dérive immédiatement en* β lorsqu'il existe $(\alpha_1, \beta_1) \in P$ tel que :

$$\exists (u, v) \in \left((\Sigma \cup V)^*\right)^2, \begin{cases} \alpha = u\alpha_1 v \\ \beta = u\beta_1 v \end{cases}$$

Le cas échéant, on note $\alpha \Rightarrow \beta$. On parle de *dérivation immédiate*. Moralement, la règle de production (α_1, β_1) remplace le facteur α_1 par le facteur β_1.

Définition 9.5 - *clôture reflexive et transitive*

On note $\Rightarrow^*$ la *clôture reflexive et transitive* de la relation $\Rightarrow$ de dérivabilité immédiate.

$\Rightarrow^*$ est définie comme la plus petite relation au sens de l'inclusion tel que :

- $\forall \alpha \in (\Sigma \cup V)^*, \alpha \Rightarrow^* \alpha$
- $\forall (\alpha, \beta) \in \left((\Sigma \cup V)^*\right)^2, (\alpha \Rightarrow \beta) \implies (\alpha \Rightarrow^* \beta)$
- $\forall (\alpha, \beta, \gamma) \in \left((\Sigma \cup V)^*\right)^3, (\alpha \Rightarrow^* \beta \text{ et } \beta \Rightarrow^* \gamma) \implies (\alpha \Rightarrow^* \gamma)$

Autrement dit, $\alpha \Rightarrow^* \beta$ lorsqu'il existe $(\alpha = \alpha_0, \ldots, \alpha_k = \beta)$ une suite de mots dans $(\Sigma \cup V)^*$ telle que :

$$\forall i \in [\![0, k-1]\!], \alpha_i \Rightarrow \alpha_{i+1}$$

Définition 9.6 - *clôture reflexive et transitive de* $\Rightarrow$

Soit $G = (\Sigma, V, P, S)$ une grammaire. Soit α et β deux mots de $(\Sigma \cup V)^*$. On note $\Rightarrow^*$ *la clôture reflexive et transitive de la relation* $\Rightarrow$. Ainsi $\alpha \Rightarrow^* \beta$ lorsqu'il existe $(\alpha = \alpha_0, \ldots, \alpha_k = \beta)$ une suite de mots dans $(\Sigma \cup V)^*$ telle que :

$$\forall i \in [\![0, k-1]\!], \alpha_i \Rightarrow \alpha_{i+1}$$

Exemple 9.7 - *de dérivation*

Dans la grammaire précédemment introduite :
Pour $\Sigma = \{a\}$ et $V = \{S, D, F, X, Y, Z\}$ et $P = \{\texttt{S} \to \texttt{DXaD}, \texttt{Xa} \to \texttt{aaX}, \texttt{XF} \to \texttt{YF}, \texttt{aY} \to \texttt{Ya}, \texttt{DY} \to \texttt{DX}, \texttt{XZ} \to \texttt{Z}, \texttt{aZ} \to \texttt{Za}, \texttt{DZ} \to \epsilon\}$, $G = (\Sigma, V, P, S)$ est une grammaire de type 0.

$$\begin{aligned} S &\Rightarrow \texttt{DXaF} \\ &\Rightarrow \texttt{DaaXF} \\ &\Rightarrow \texttt{DaaYF} \\ &\Rightarrow \texttt{DaYaF} \\ &\Rightarrow \texttt{DYaaF} \\ &\Rightarrow \texttt{DXaaF} \\ &\Rightarrow \texttt{DaaXaF} \\ &\Rightarrow \texttt{DaaaaXF} \\ &\Rightarrow \texttt{DaaaaZ} \\ &\Rightarrow \texttt{DaaaZa} \\ &\Rightarrow \texttt{DaaZaa} \\ &\Rightarrow \texttt{DaZaaa} \\ &\Rightarrow \texttt{DZaaaa} \\ &\Rightarrow \texttt{aaaa} \end{aligned}$$

D'où $S \Rightarrow^* \texttt{aaaa}$

Définition 9.8 - *langage engendré, langage élargi engendré par une grammaire depuis un mot*

Soit $G = (\Sigma, V, P, S)$ une grammaire et $\alpha \in (\Sigma \cup V)^*$.

- le *langage engendré par G depuis α* est l'ensemble des mots de Σ^* que l'on peut obtenir par dérivation de α en utilisant les règles de production de G :

$$\mathcal{L}_G(\alpha) = \{\beta \in \Sigma^*, \alpha \Rightarrow^* \beta\}$$

- le *langage élargi engendré par G depuis α* est l'ensemble des mots de $(\Sigma \cup V)^*$ que l'on peut obtenir par dérivation de α en utilisant les règles de production de G :

$$\widehat{\mathcal{L}_G(\alpha)} = \{\beta \in (\Sigma \cup V)^*, \alpha \Rightarrow^* \beta\}$$

Définition 9.9 - *langage engendré par une grammaire*

Soit $G = (\Sigma, V, P, S)$ une grammaire et $\alpha \in (\Sigma \cup V)^*$. Le *langage engendré par G* désigne $\mathcal{L}_G(S)$ le langage engendré par G depuis son symbole initial S.

Exemple 9.10

Pour la grammaire de l'exemple, on pourrait montrer que $\mathcal{L}_G(S) = \{a^{2^n}, n \in \mathbb{N}^*\}$ On montre que ce langage n'est pas régulier (absurde + lemme de l'étoile tmtc)

Définition 9.11 - *langage de type 0*

On dit qu'un *langage est de type 0* s'il peut être engendré par une grammaire de type 0.

Théorème 9.12 - *de Chomsky (HP)*

Les langages de type 0 sont exactement les langages récursivement énumérables, c'est-à-dire les langages reconnaissables par une machine de Turing.

Définition 9.13 - *grammaire contextuelle (HP)*

Une grammaire $G = (\Sigma, V, P, S)$ de type 0 est appelée *grammaire contextuelle (ou de type 1 ou monotone)* lorsque :

$$\forall(\alpha, \beta) \in P \setminus \{(S, \epsilon)\},\ |\alpha| \leq |\beta|$$

Moralement, tous les facteurs "produits" sont plus long que les facteurs remplacés.

Exemple 9.14 - *de grammaire qui n'est pas "contextuelle"*

la grammaire exemple définie par $\Sigma = \{a\}$ et $V = \{S, D, F, X, Y, Z\}$ et $P = \{\texttt{S} \to \texttt{DXaD}, \texttt{Xa} \to \texttt{aaX}, \texttt{XF} \to \texttt{YF}, \texttt{aY} \to \texttt{Ya}, \texttt{DY} \to \texttt{DX}, \texttt{XZ} \to \texttt{Z}, \texttt{aZ} \to \texttt{Za}, \texttt{DZ} \to \epsilon\}$ puis $G = (\Sigma, V, P, S)$ n'est pas contextuelle :

$$(\texttt{DZ} \to \epsilon) \in P$$

mais :

$$|DZ| = 2 > |\epsilon| = 0$$

Remarque 9.15 - *indépendance des caractères "contextuel" et "non contextuel"*

Nous le reverrons, mais une grammaire est "contextuelle" indépendamment de son caractère "non contextuel".

Remarque 9.16

Sans l'autorisation d'avoir $(S, \epsilon) \in P$, les langages engendrés par des grammaires contextuelles ne peuvent pas contenir ϵ.
En effet, on a initialement S de longueur 1 et les règles de productions ne peuvent qu'augmenter la longueur du mot. 0 serait donc une longueur inexistente.

Définition 9.17 - *grammaire non contextuelle*

Une grammaire $G = (\Sigma, V, P, S)$ est dite *non contextuelle* (ou *hors contexte* ou encore *algébrique* ou *de type 2*) lorsque :

$$P \subset V \times (\Sigma \cup V)^*$$

Moralement, les règles de production ne permettent de remplacer que des symboles non terminaux (et pas des mots) : des majuscules.

Exemple 9.18 - *de règle de production non valide pour une grammaire "non contextuelle"*

`DZ` $\rightarrow \epsilon$ n'est pas une règle possible pour une grammaire hors contexte.

Exemple 9.19 - *de grammaire "non contextuelle"*

$G_1 = (\Sigma_1, V_1, P_1, S_1)$ définie par :

- $\Sigma_1 = \{a, b\}$
- $V_1 = \{S_1\}$
- $P_1 = \{S_1 \rightarrow aS_1b,\ S_1 \rightarrow \epsilon\}$

est une grammaire hors contexte.

Remarque 9.20

On dit "hors contexte" car les symboles non terminaux sont considérés seuls par les règles de production : on ne regarde pas les lettres autour.

Exemple 9.21 - *langage engendré par une grammaire hors contexte*

$G_1 = (\Sigma_1, V_1, P_1, S_1)$ définie par :

- $\Sigma_1 = \{a, b\}$
- $V_1 = \{S_1\}$
- $P_1 = \{S_1 \to aS_1b,\ S_1 \to \epsilon\}$

Alors $\mathcal{L}_{G_1}(S_1) = \{a^n b^n,\ n \in \mathbb{N}\}$. On le démontre (Démo 1). Par ailleurs $\widehat{\mathcal{L}_{G_1}(S_1)} = \{a^n S_1 b^n,\ n \in \mathbb{N}\} \cup \{a^n b^n,\ n \in \mathbb{N}\}$.

Exemple 9.22 - *Langage de Dyck*

Le langage de Dyck (bon parenthésage) est engendré par :

- $\Sigma = \{\texttt{(}, \texttt{)}\}$
- $S \to (S)S$
- $S \to \epsilon$

Remarque 9.23 - *grammaires "non contextuelles" uniquement définies par les règles de production*

En général, on donne une grammaire hors contexte uniquement avec les règles de production.
On ne précise le symbole initial que si ce n'est pas S.
Les symboles non terminaux sont exactement ceux que l'on rencontre à gauche des règles et les terminaux sont les autres qu'on rencontre.

Remarque 9.24 - *abréviation d'une famille de règles de production*

Pour noter rapidement un ensemble de règles de production utilisant le même symbole non terminal comme départ :

$$\{\texttt{X} \to \alpha_1,\ \ldots,\ \texttt{X} \to \alpha_k\} \subset P$$

avec $X \in V$, on note :

$$X \to \alpha_1 | \cdots | \alpha_k$$

Ce n'est qu'une notation, on a bien k règles de production derrière ça.

Exemple 9.25 - *exploitant ce raccourci*

G_1 s'écrit :

$$S_1 \to aS_1b | \epsilon$$

1.2 Langages réguliers et langages hors contexte

Un résultat à connaître et savoir redémontrer.

Proposition 9.26 - *régulier* $\implies$ *hors contexte*

Soit Σ un alphabet, Tout langage sur Σ régulier est hors contexte.

Voir Démo 2 (elle est incomplète)

Remarque 9.27 - *hors contexte* $\not\Rightarrow$ *régulier*

il existe des langages hors contexte qui ne sont pas réguliers. En effet on a montré que pour $G_1 = (\Sigma_1, V_1, P_1, S_1)$ définie par :

- $\Sigma_1 = \{a, b\}$
- $V_1 = \{S_1\}$
- $P_1 = \{S_1 \to aS_1b,\ S_1 \to \epsilon\}$

Alors $\mathcal{L}_{G_1}(S_1) = \{a^n b^n,\ n \in \mathbb{N}\}$.

Ce langage est donc hors contexte, mais n'est pas régulier.

Définition 9.28 - *grammaire hors contexte linéaire (HP)*

Une grammaire $G = (\Sigma, V, P, S)$ hors contexte est dite *linéaire* si :

$$P \subset V \times \left(\Sigma^* \cup \Sigma^* V \Sigma^*\right)$$

Moralement, on a toujours au plus un symbole non terminal (voir Fig.1)

Définition 9.29 - *grammaire hors contexte linéaire gauche (HP)*

Une grammaire $G = (\Sigma, V, P, S)$ hors contexte est dite *linéaire* si :

$$P \subset V \times \left(\Sigma^* \cup V \Sigma^*\right)$$

Moralement, Le facteur "produit" commence par un unique symbole non terminal ou n'en a aucun.

Définition 9.30 - *grammaire hors contexte linéaire droite (HP)*

Une grammaire $G = (\Sigma, V, P, S)$ hors contexte est dite *linéaire* si :

$$P \subset V \times \left(\Sigma^* \cup \Sigma^* V\right)$$

Moralement, Le facteur "produit" termine par un unique symbole non terminal ou n'en a aucun.

Remarque 9.31 - *chaine d'implications*

linéaire droit implqiue linéaire implique hors contexte linéaire gauche implique linéaire implique hors contexte.

Définition 9.32 - *langages linéaire, linéaire droit, linéaire gauche*

Un langage est dit linéaire (resp. simple, droit, gauche) lorsqu'il peut être engendré par une grammaire linéaire (resp. simple, linéaire droite).

Remarque 9.33 - *existence de langages linéaires non réguliers*

$$\mathtt{S} \to \mathtt{aSb} | \epsilon$$

est linéaire et engendre $\{a^n b^n,\ n \in \mathbb{N}\}$

Proposition 9.34 - *pour un langage, régulier $\iff$ linéaire droit $\iff$ linéaire gauche*

Soit L un langage. Les propriétés suivantes sont équivalentes.

1. L est régulier
2. L est linéaire droit
3. L est linéaire gauche

Remarque 9.35 - *sur la démo*

En montrant régulier implqiue linéaire droit, on montre à nouveau que régulier implqiue hors contexte, ceci constitue une autre démo que la propriété précédente.

Démo 3.
subsectionSystème d'équations d'une grammaire

Définition 9.36

Soit $G = (\Sigma, V, P, S)$ une grammaire hors contexte avec $V = \{S, X_1, \ldots, X_n\}$. Soit $L = (L_0, \ldots, L_n)$ un $(n+1)$-uplet de langages sur Σ. On définit par induction :

- $\epsilon(L) = \{\epsilon\}$ (base)
- $\forall a \in \Sigma,\ a(L) = \{a\}$ (base)
- $S(L) = L_0$ (base)
- $\forall i \in [\![1, n]\!],\ X_i(L) = L_i$ (base)
- $\forall (\alpha, \beta) \in \left((\Sigma \cup V)^*\right)^2,\ \alpha\beta(L) = \alpha(L) \cdot \beta(L)$ (compatibilité avec la concaténation)
- $\forall K \subset (\Sigma \cup V)^*,\ K(L) = \bigcup_{w \in K} w(L)$ (compatibilité avec l'union)

Le *système d'équations $\mathcal{S}(G)$ de la grammaire G* est le suivant :

$$\begin{cases} \forall i \in [\![1, n]\!],\ L_i = \bigcup_{(X_i, \alpha) \in P} \alpha(L) \\ L_0 = \bigcup_{(S, \alpha) \in P} \alpha(L) \end{cases}$$

Remarque 9.37

À chaque symbole on associe un langage L_i. On veut montrer que $\mathcal{L}_G = (\mathcal{L}_G(S), \mathcal{L}_G(X_1), \ldots, \mathcal{L}_G(X_n))$ est solution du système $\mathcal{S}(G)$

Exemple 9.38

Avec cette grammaire, on obtient le système suivant :

$$\begin{cases} L_0 = \{a\}L_1\{b\} \\ L_1 = \{c\}L_2\{d\} \\ L_2 = \{\epsilon\} \cup L_0 \end{cases}$$

Remarque 9.39 - *notation liées à un sytème d'équations d'une grammaire*

On note $\mathcal{L}_G = (\mathcal{L}_G(S), \mathcal{L}_G(X_1), \ldots, \mathcal{L}_G(X_n))$ le $(n+1)$-uplet de langages engendrés par les symboles non terminaux.

Proposition 9.40 - *lemme 1 concernant ce $(n+1)$-uplet*

$\forall \alpha \in (V \cup \Sigma)^*,\ \mathcal{L}_G(\alpha) = \alpha(\mathcal{L}_G)$

Démo 4

Proposition 9.41 - *Lemme 2 concernant ce ...*

Soit L un $(n+1)$-uplet de langages sur Σ solution de $\mathcal{S}(G)$.

$$\forall(\alpha,\beta) \in (\Sigma \cup V)^*,\ (\alpha \Rightarrow^* \beta) \implies (\beta(L) \subset \alpha(L))$$

demo 5

Théorème 9.42 - *utilisant les deux lemmes*

$\mathcal{L}_G = (\mathcal{L}_G(S), \mathcal{L}_G(X_1), \ldots, \mathcal{L}_G(X_n))$ est l'unique solution minimale pour l'inclusion composante par composannte de $\mathcal{S}(G)$.

Démo 6

Exemple 9.43 - *pratique*

En pratique on peut résoudre le système en utilisant le lemme d'Arden pour déterminer le langage engendré par une grammaire hors contexte pas trop complexe. D'après la théorie précédente, la solution sera l'unique solution minimale pour le système de grammaire.

$$\begin{cases} L_0 = L_0 L_1 \cup \{\epsilon\} \\ L_1 = \{a\} L_1 \{b\} \end{cases}$$

Voir Démo 7

Rappel

Théorème 9.44 - *lemme d'Arden (revisité mdr)*

Soit A et B deux langages sur un même alphabet Σ. Toute solution de $L = AL \cup B$ continet A^*B. En particulier A^*B est solution et elle est unique si $\epsilon \notin A$. On a un résultat analogue pour LA et AL !

Revoir la démo de ça et traiter les 4 cas.

Exemple 9.45 - *pénible*

Pour $S \to aSb|\epsilon$, On ne peut pas appliquer le lemme d'Arden, il faut faire une induction :

$$L = (A_1 L A_2) \cup B$$

Une solution minimale est $\{m_1 m_0 m_2, k \in \mathbb{N}, m_1 \in A_1^k, m_2 \in A_2^k, m_0 \in B\}$. La démo est celle du lemme d'Arden mais il ne couvre pas ce cas c'est tout.

Remarque 9.46

Les systèmes d'équation de grammaire et le lemme d'Arden sont des outils HORS PROGRAMME. À l'écrit, il faut le redémontrer une fois avant d'utiliser le lemme d'Arden ou faire une induction structurelle dans les cas particuliers.
Pour $\mathcal{S}(G)$, on justifie par une phrase en s'appuyant sur la première règle utilisée et on donne $\mathcal{S}(G)$.

1.3 1.4

Passe 2

1.4 Hiérarchie de Chomsky - HP, culture générale

Théorème 9.47 - *hiérarchie de Chomsky*

Il s'agit d'un théorème qui définit une "hiérarchie" sur les langages :

1. Tout langage régulier (rationnel, de type 3) est un langage linéaire (réciproque fausse)
2. Tout langage linéaire est un langage hors contexte (de type 2). La réciproque est fausse.
3. Tout langage hors contexte (de type 2) est un langage contextuel (de type 1). La réciproque est fausse
4. Tout langage contextuel (de type 1) est un langage récursif (réciproque fausse). C'est à dire les langages récurssifs (*i.e.* reconnus par une machine de Turing déterministe, sans calcul infini).
5. Tout langage récursif est récursivement énumérable (= de type 0 d'après le théorème de Chomsky) (reconnaissable par une machine de Turing).

Voir Fig.8

sectionArbre d'analyse

Définition 9.48 - *arbre d'analyse*

Un *arbre d'analyse* (ou *arbre d'analyse syntaxique*, ou *arbre de dérivation*) pour une grammaire hors contexte $G = (\Sigma, V, P, S)$ et un mot $w \in \mathcal{L}_G(S)$ est un arbre :

- dont les étiquettes sur les noeuds sont des symboles dans $\Sigma \cup V \cup \{\epsilon\}$:
  - $\Sigma \cup \{\epsilon\}$ pour les feuilles
  - V pour les noeuds internes (racine comprise)
- dont la racine est étiquettée par S
- pour tout noeud interne n d'étiquette $X \in V$, de fils d'étiquettes $x_1, \ldots, x_k$, on a $(X, x_1, \ldots x_k) \in P$

On appelle alors *frontière* de l'arbre d'analyse le mot obtenu en concaténant les symboles des feuilles de gauche à droite : c'est w.

Exemple 9.49 - *d'arbre d'analyse*

Pour $G : S \to aSb|$, Voir Fig.9

Remarque 9.50 - *propriété de l'arbre d'analyse*

L'arbre de l'arbre d'analyse est la preuve qu'un mot est engendré par une grammaire. Il rend compte de toutes les règles de production utilisées mais pas entièrement de l'ordre. On le verra au prochain cours.