Langages formels

L3 Informatique / Math-Info ENS Paris-Saclay
2025-2026

Transparents élaborés par Paul Gastin et Stefan Schwoon

Rappel: Grammaires

Définition : $G = \langle \Sigma, V, P, S \rangle$

- Σ alphabet des *terminaux* (ensemble fini)
- V alphabet des *variables* (ensemble fini)
- P ensemble fini de *productions*
- S variable initiale

Hiérarchie de Chomsky (1956) :

- Grammaire de type 0 : $P \subseteq (\Sigma \cup V)^+ \times (\Sigma \cup V)^*$
- Grammaire de type 1 : $P \subseteq \{ \alpha \to \beta \mid \alpha, \beta \in (\Sigma \cup V)^+,\ |\alpha| \leq |\beta| \}$
- Grammaire de type 2 : $P \subseteq V \times (\Sigma \cup V)^+$
- Grammaire de type 3 : $P \subseteq V \times (\Sigma V \cup \Sigma)$

Exemple de type 2: $S \to aSb \mid ab$ engendre $\{ a^n b^n \mid n \geq 1 \}$.

Le cas du mot vide

Remarque : Exception $S \to \varepsilon$

Avec la définition précédente, les grammaires du type 0 sont les seules à générer le mot vide. Afin d'établir une meilleure correspondance entre grammaires et automates, on ajoute l'exception suivante pour les autres types :
La production $S \to \varepsilon$ est permise si S ne figure pas dans la partie droite d'une production.

Grammaire de type 2 faible : $P \subseteq V \times (\Sigma \cup V)^*$

Proposition : Elimination des ε

Soit $G = \langle \Sigma, V, P, S \rangle$ une grammaire de type 2 faible. Il existe une grammaire de type 2 qui engendre le même langage que G.

Du coup, nous allons utiliser la version faible de la définition.

Une grammaire de type 2 (ainsi que son langage engendré) s'appelle aussi *hors-contexte* ou *algébrique*.

Elimination des ε

Preuve

Soit $G' = \langle \Sigma, V \uplus \{S_0\}, P', S_0 \rangle$. On obtient P' en commençant par $P \cup \{S_0 \to S\}$ et en répétant la modification suivante :

- Tant que P' contient une production $A \to \varepsilon$, avec $A \neq S_0$, éliminer $A \to \varepsilon$ et, pour toute production $B \to \alpha A \beta$, ajouter $B \to \alpha\beta$.

Exemple : La grammaire faible $S \to aSb \mid \varepsilon$ engendre $\{ a^n b^n \mid n \geq 0 \}$.

Conversion en grammaire de type 2 :

- Ajouter $S_0 \to S$.
- Remplacer $S \to \varepsilon$ par $S_0 \to \varepsilon$ et $S \to ab$.

La grammaire résultante ($S_0 \to S \mid \varepsilon$, $S \to aSb \mid ab$) est bien équivalente.

Dérivations / ambigüité

Définition : Dérivations

- dérivation gauche $\rightarrow_g^*$: remplacer toujours la variable la plus à gauche
- dérivation droite $\rightarrow_d^*$: analogue

Définition : Ambigüité

- Une grammaire est ambigüe s'il existe deux arbres de dérivations (distincts) de même racine et de même frontière.
- Un langage algébrique est *non ambigu* s'il existe une grammaire non ambigüe qui l'engendre. Dans le cas contraire, on dit qu'il est *inhéremment ambigu*.

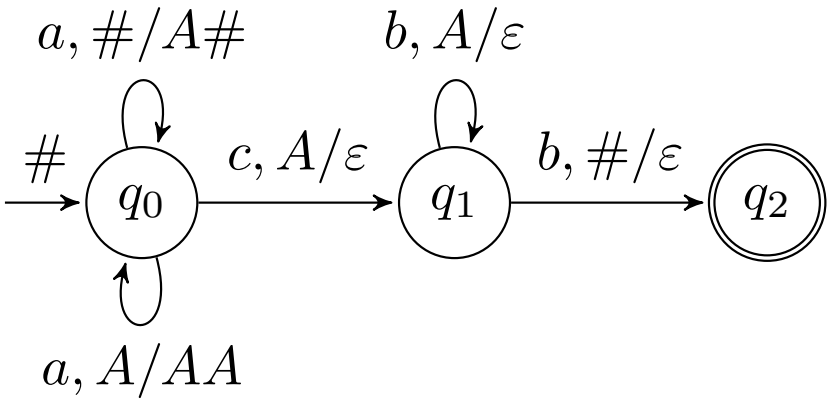
Exemple :

Arbres de dérivation pour $G_1 := S \rightarrow SS \mid aSb \mid \varepsilon$ et $G_2 := S \rightarrow aSbS \mid \varepsilon$.

Automate à pile (Exemple)

Exemple : Automate $\mathcal{A}_1$

On considère un espèce d'automate équipé d'une pile de symboles.



Les transitions portent deux annotations supplémentaires, notées A/w :

- A note le symbole qui doit être au sommet de la pile.
- La transition remplace A par w (un mot), le nouveau sommet de pile étant le premier symbole de w (sommet à gauche).
- La flèche qui marque l'état initial contient le contenu initial de la pile.

Automates à pile (AAP)

Définition : $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ où

- Q ensemble fini d'états
- Σ alphabet d'entrée
- Z alphabet de pile
- $T \subseteq QZ \times (\Sigma \cup \{\varepsilon\}) \times QZ^*$ ensemble fini de transitions
- $q_0 z_0 \in QZ$ configuration initiale
- $F \subseteq Q$ acceptation par état final.

Notation graphique : voir transparent précédent

Cas spéciaux :

- $\mathcal{A}$ est *temps-réel* (TR) s'il n'a pas d'ε-transition.
- $\mathcal{A}$ est *simple* (S) s'il ne possède qu'un seul état (que l'on ne note pas).

Sémantique d'un AAP

Définition : Système de transitions (infini) associé

- $\mathcal{T} = \langle QZ^*, T', q_0 z_0, FZ^* \rangle$
- Une configuration de $\mathcal{A}$ est un élement $ph \in QZ^*$ de $\mathcal{T}$
- Transitions de $\mathcal{T}$: $T' = \{\, pzh \xrightarrow{a} qgh \mid \langle pz, a, qg \rangle \in T,\ h \in Z^* \,\}$
- $\mathcal{L}(\mathcal{A}) = \{\, w \in \Sigma^* \mid \exists qh \in FZ^* : q_0 z_0 \xrightarrow{w}{}^* qh \,\}$

Exemple : $\mathcal{L}(\mathcal{A}_1) = \{\, a^n c b^n \mid n \geq 1 \,\}$

Exemple de calcul:

$$q_0\# \xrightarrow{a} q_0 A\# \xrightarrow{a} q_0 AA\# \xrightarrow{c} q_1 A\# \xrightarrow{b} q_1\# \xrightarrow{b} q_2$$

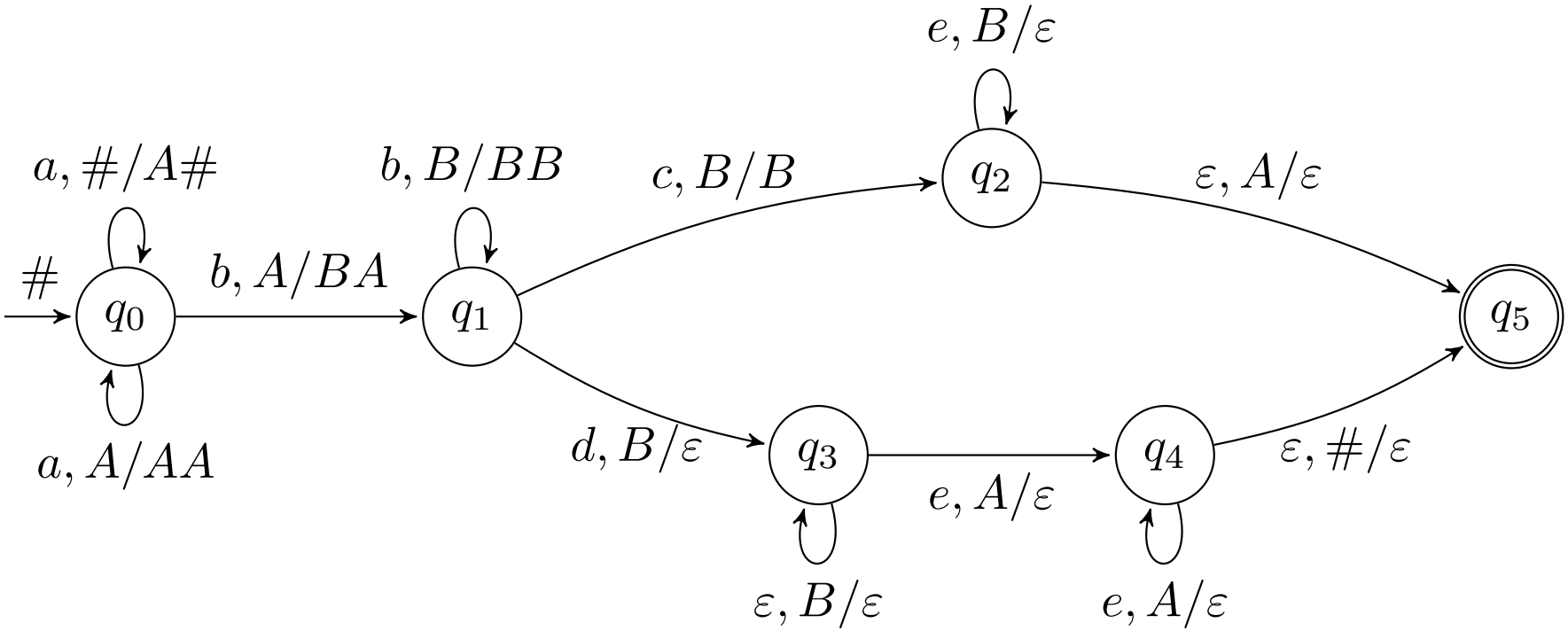
Remarque : On récupère les NFA avec $Z = \{z\}$ et z/z sur toute transition.

AAP avec ε-transitions

Attention, les ε-transitions peuvent s'enchaîner, en modifiant la pile :

Exemple : Automate $\mathcal{A}_2$

Cet automate accepte $\{ a^n b^k c e^k \mid n, k \geq 1 \} \cup \{ a^n b^k d e^n \mid n, k \geq 1 \}$.



Automates à pile: Exemples

Exemples :

- $L_1 = \{ a^n b^n c^k \mid n, k > 0 \}$ et $L_2 = \{ a^n b^k c^k \mid n, k > 0 \}$
- $L = L_1 \cup L_2$ (non déterministe)

Exercices :

1. Montrer que le langage $\{ w\tilde{w} \mid w \in \Sigma^* \}$ et son complémentaire peuvent être acceptés par un AAP. (Note: $\tilde{w}$ = miroir de w)
2. Montrer que le *complémentaire* du langage $\{ ww \mid w \in \Sigma^* \}$ peut être accepté par un AAP.
3. Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ un AAP. Montrer qu'on peut construire un AAP équivalent $\mathcal{A}'$ tel que $T' \subseteq Q'Z \times (\Sigma \cup \{\varepsilon\}) \times Q'Z^{\leq 2}$.
4. Même question pour les AAP temps-réel.

Motivation

Intérêt des AAP et grammaires algébriques :

Début dans les années 1950s :

- étude des grammaires formelles (connection linguistique/informatique)
- premiers ordinateurs : ajout d'un pile pour faciliter la programmation modulaire
- développement des langages de programmation : analyse syntaxique les machines de Turing

$\rightarrow$ équivalences entre grammaires et modèles de calcul

Programme (pour aujourd'hui) :

- établir quelques propriétés fondamentales des AAP
- équivalence entre AAP et grammaires algébriques

Dans la suite, sauf mention particulière, toute *grammaire* est *algébrique*.

Acceptation généralisée

Définition :

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un AAP et $K \subseteq QZ^*$ un langage reconnaissable.
Le langage reconnu par $\mathcal{A}$ avec *acceptation généralisée* K est

$$\mathcal{L}_K(\mathcal{A}) = \{\, w \in \Sigma^* \mid \exists qh \in K : q_0 z_0 \xrightarrow{w}{}^* qh \,\}$$

Cas particuliers :

- $K = FZ^*$: acceptation classique par état final.
- $K = Q$: acceptation par pile vide.
- $K = F$: acceptation par pile vide et état final.
- $K = QZ'Z^*$ avec $Z' \subseteq Z$: acceptation par sommet de pile.

Exemple :

$\mathcal{L}(\mathcal{A}_1) = \{\, a^n c b^n \mid n \geq 1 \,\}$ peut être accepté par pile vide ou par sommet de pile.

Proposition : Acceptation généralisée

Soit $\mathcal{A}$ un automate à pile avec acceptation généralisée K. On peut effectivement construire un automate à pile $\mathcal{A}'$ acceptant par état final tel que $\mathcal{L}_K(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$.
Du coup, tous les modes d'acceptation ci-dessus sont équivalents.

Preuve

Idée : $\mathcal{A}'$ se comporte comme $\mathcal{A}$, mais à n'importe quel moment il peut basculer dans une phase de vérification qui simule un automate fini pour K.
Soit $\mathcal{A}_f = \langle Q \cup Z, S, \delta, s_0, F \rangle$ un NFA qui accepte K. Alors $\mathcal{A}' = \langle Q \uplus S, \Sigma, Z, T', q_0 z_0, F \rangle$ avec

$$T' = T \cup \{ \langle q, z, \varepsilon, s', z \rangle \mid \langle s_0, q, s' \rangle \in \delta,\ z \in Z \}$$
$$\cup \{ \langle s, z, \varepsilon, s', \varepsilon \rangle \mid \langle s, z, s' \rangle \in \delta \}$$

Attention, cette vérification détruit la pile !

Automates à pile et grammaires

Proposition : Grammaire vers AAP

Soit $G = \langle \Sigma, V, P, S \rangle$ une grammaire. On peut construire un AAP *simple* $\mathcal{A}$ qui accepte $\mathcal{L}_G(S)$ par pile vide.

Preuve (dérivation gauche / top-down – idée)

$\mathcal{A} = \langle \Sigma, \Sigma \cup V, T, S \rangle$ avec deux types de transitions dans T :

- expansions : $\langle A, \varepsilon, \alpha \rangle$ pour tout $A \to \alpha \in P$,
- vérifications : $\langle a, a, \varepsilon \rangle$ pour tout $a \in \Sigma$.

Pour tout $\alpha \in (\Sigma \cup V)^*$ et $w \in \Sigma^*$, prouver $\alpha \to_g^* w$ dans G ssi $\alpha \xrightarrow{w}{}^* \varepsilon$ dans $\mathcal{A}$:

- $\Rightarrow$: recurrence sur longueur de dérivation
- $\Leftarrow$: recurrence sur longueur du calcul

Automates à pile et grammaires

Proposition : Automate à pile vers grammaire

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un automate à pile reconnaissant par pile vide. On peut construire une grammaire G qui engendre $\mathcal{L}(\mathcal{A})$.

Construction:

- $G = \langle \Sigma, \{S\} \cup (Q \times Z \times Q), P, S \rangle$
- pour tout $q \in Q$, $S \to \langle q_0, z_0, q \rangle \in P$;
- pour tout $a \in \Sigma \cup \{\varepsilon\}$ et $\langle qz, a, q' \rangle \in T$, $\langle q, z, q' \rangle \to a \in P$;
- pour tout $a \in \Sigma \cup \{\varepsilon\}$, $n \geq 1$, $\langle qz, a, q' z_1 \cdots z_n \rangle \in T$ et $q_1, \ldots q_n \in Q$,

$$\langle q, z, q_n \rangle \to a \langle q', z_1, q_1 \rangle \langle q_1, z_2, q_2 \rangle \cdots \langle q_{n-1}, z_n, q_n \rangle \in P$$

Preuve (idée): $\langle q, z, q' \rangle \to^* w$ dans G ssi $qz \xrightarrow{w}{}^* q'$ dans $\mathcal{A}$
(par récurrence sur longueur de dérivation resp. calcul)

Forme normale de Chomsky

Définition : FNC

Soit $G = (\Sigma, V, P, S)$ une grammaire. G est dite en *forme normale de Chomsky* (FNC) si $P \subseteq (V \times (V^2 \cup \Sigma)) \cup \{S \to \varepsilon\}$; autrement dit, toute production est de la forme (i) $A \to BC$, (ii) $A \to a$ ou (iii) $S \to \varepsilon$.
Si $S \to \varepsilon \in P$, alors $B \neq S$ et $C \neq S$ dans les productions de type (i).

Théorème : Conversion en FNC

Pour toute grammaire G il existe une grammaire FNC G' telle que $\mathcal{L}(G) = \mathcal{L}(G')$.

Preuve (idée):

- Introduire une nouvelle variable initiale S_0 et $S_0 \to S$.
- Pour tout $a \in \Sigma$, introduire variable V_a; remplacer toute occurrence de a dans les productions par V_a et ajouter $V_a \to a$.
- Limiter la longueur de la côte droite de toute production à deux (p.ex. remplacer $A \to BCD$ par $A \to C'D$ et $C' \to BC$).
- Éliminer toutes les productions du type $B \to \varepsilon$ (sauf $S_0 \to \varepsilon$).
- Pour toute paire $A \to B$ et $B \to \beta$, ajouter $A \to \beta$.
- Supprimer toute production de la forme $A \to B$.

Lemme d'itération

Théorème : Bar-Hillel, Perles, Shamir ou Lemme d'itération

Soit $L \subseteq \Sigma^*$ algébrique. Il existe $N \geq 0$ tel que pour tout $w \in L$, si $|w| \geq N$ alors on peut trouver une factorisation $w = \alpha u \beta v \gamma$ avec (i) $|uv| > 0$ et (ii) $|u\beta v| \leq N$ et (iii) $\alpha u^n \beta v^n \gamma \in L$ pour tout $n \geq 0$.

Preuve: Soit G une grammaire FNC avec k variables et $\mathcal{L}(G) = L$. Choisir $N := 2^k$. Un arbre de dérivation de n'importe quel mot $w \in L$ avec $|w| \geq N$ contient un chemin avec au moins $k+1$ variables. Parmi les $k+1$ dernières variables d'un chemin de longueur maximale, une variable A se répète, du coup:

$$S \to^* \alpha A \gamma \to^+ \alpha u A v \gamma \to^+ \alpha u \beta v \gamma$$

avec les propriétés (i), (ii) et (iii).

Exemple :

Le langage $L_1 = \{a^n b^n c^n \mid n \geq 0\}$ n'est pas algébrique.

Corollaire :

Le langages algébriques ne sont pas fermées par intersection ou complémentaire.