Langages formels

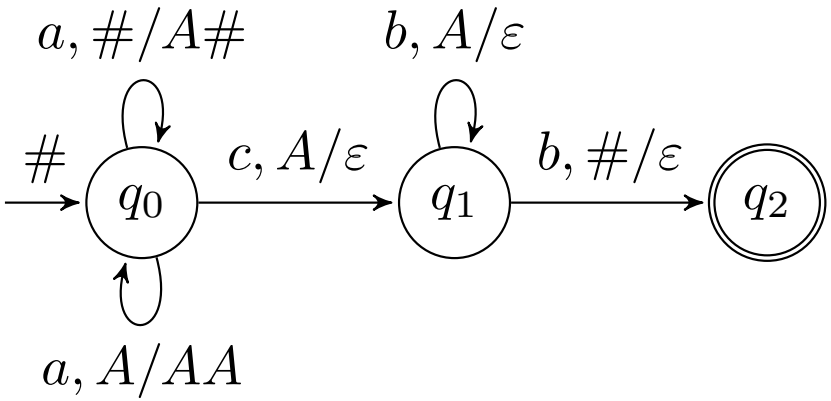
L3 Informatique/Math-Info ENS Paris-Saclay
2024-2025

Transparents élaborés par Paul Gastin et Stefan Schwoon

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 ;
- en prenant la transition, A est remplacé par une suite w de symboles, le nouveau sommet de pile étant le premier symbole de w (sommet à gauche) ;
- la flèche marquant 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.
  On s'autorise d'omettre l'état dans ce cas-là.
- Un AAP avec $Z = \{z\}$ et z/z sur toute transition est équivalent à un automate fini.

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 état $ph \in QZ^*$ de $\mathcal{T}$
- Transitions de $\mathcal{T}$: $T' = \{pzh \xrightarrow{a} qgh \mid (pz, a, qg) \in T\}$.

Notation:

- $pg \xrightarrow[n]{w}_{\mathcal{A}} qh$ si un chemin de longueur n et étiqueté par w existe entre pg et qh dans $\mathcal{T}$; $pg \xrightarrow{w}{}^*_{\mathcal{A}} qh$ pour un chemin de longueur quelconque.
- On omet w ou $\mathcal{A}$ au besoin.

Définition : Langage d'un AAP

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

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

Exemple de calcul:

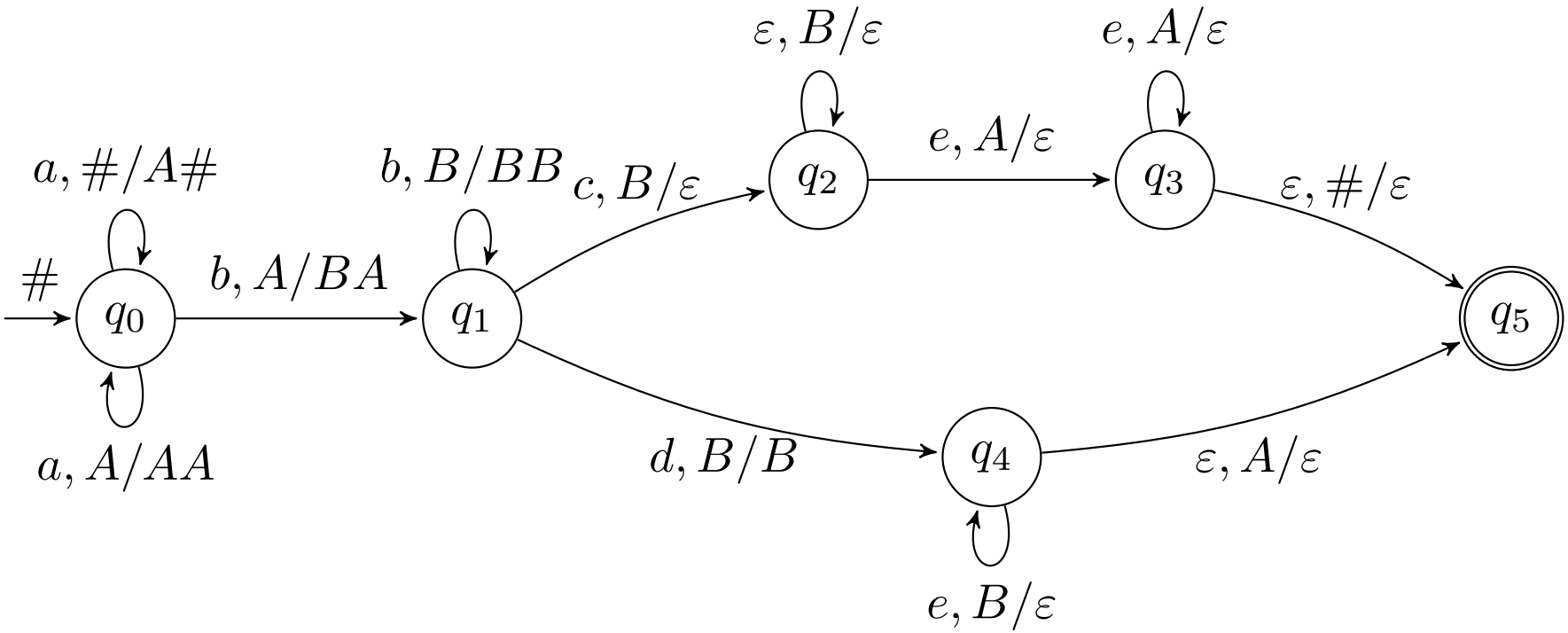
$$q_0\# \xrightarrow{a} q_0 A\# \xrightarrow{a} q_0 AA\# \xrightarrow{c} q_1 A\# \xrightarrow{b} q_1\# \xrightarrow{b} q_2$$

AAP avec ε-transitions

Attention, les ε-transitions peuvent enchaîner des modifications de pile !

Exemple : Automate $\mathcal{A}_2$

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



Automates à pile: Exemples

Exemples :

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

Exercices :

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

Intérêt des AAP :

- étude d'un modèle de calcul situé entre les automates finis et les machines de Turing ;
- pile = composant naturel d'un ordinateur
- analyse syntaxique

Rappel: Grammaires

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

- Σ alphabet *terminal* (fini)
- V *variables* (ensemble fini)
- $P \subseteq (\Sigma \cup V)^+ \times (\Sigma \cup V)^*$ ensemble fini de *productions*
- S variable initiale

Grammaire de type 2 (*hors contexte* ou *algébrique*) : $P \subseteq V \times (\Sigma \cup V)^*$

Exemple: La grammaire $S \to aSb \mid acb$ engendre $\{ a^n c b^n \mid n \geq 1 \} = \mathcal{L}(\mathcal{A}_1)$.

Remarques :

- $\alpha \to_G^* w$ (ou $\alpha \to^* w$ si G est évident) dénote un dérivation dans G
- dérivation gauche $\to_g^*$: remplacer toujours la variable la plus à gauche
- dérivation droite $\to_d^*$: analogue

Programme (pour aujourd'hui) :

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

Propriété fondamentale des AAP

Lemme :

Soient $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un automate à pile, $p, r \in Q$, $g, h \in Z^*$, $w \in \Sigma^*$ et $n \geq 0$. Les conditions suivantes sont équivalentes:

1. $pgh \xrightarrow[n]{w} r$ est un calcul de $\mathcal{A}$,
2. il existe $q \in Q$, $w_1 w_2 = w$ et $n_1 + n_2 = n$ et deux calculs $pg \xrightarrow[n_1]{w_1} q$ et $qh \xrightarrow[n_2]{w_2} r$ de $\mathcal{A}$.

Preuve

$\Longrightarrow$: Dans le calcul $pgh \xrightarrow[n]{w} r$, on considère la première fois que le contenu de la pile est h (possible car initialement gh, finalement ε). Soit qh la configuration correspondante après n_1 étapes. Le calcul s'écrit $pgh \xrightarrow[n_1]{w_1} qh \xrightarrow[n_2]{w_2} r$.

Si $p_0 g_0 h \xrightarrow{a_1} p_1 g_1 h \cdots \xrightarrow{a_k} p_k g_k h$ est un calcul de $\mathcal{A}$ tel que $g_i \neq \varepsilon$ pour $0 \leq i < k$ alors $p_0 g_0 \xrightarrow{a_1} p_1 g_1 \cdots \xrightarrow{a_k} p_k g_k$ est un calcul de $\mathcal{A}$ (on obtient $pg \xrightarrow[n_1]{w_1} q$ avec $k = n_1$, $g_{n_1} = \varepsilon$, $p_k = q$).

$\Longleftarrow$: On utilise la remarque suivante: Si $sf \xrightarrow[k]{v} s'f'$ est un calcul de $\mathcal{A}$ avec $s \in Q$, $f, f', f'' \in Z^*$ alors $sff'' \xrightarrow[k]{v} s'f'f''$ est aussi un calcul de $\mathcal{A}$.

Acceptation généralisée

Définition :

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un automate à pile 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 on bascule dans un automate fini pour K qui vérifie le contenu de la pile.
D'abord, supposons que dans tout calcul de $\mathcal{A}$, il reste au moins un symbol sur la pile. (Si ce n'est pas déjà le cas, ajouter un nouvel état et un nouveau symbole initial de pile, disons $q_0'\#'$, et une transition $\langle q_0'\#', \varepsilon, q_0\# \rangle$ si $q_0\#$ était l'ancienne configuration initiale.)
Soit $\mathcal{A}_f = \langle Q \cup Z, S, \delta, s_0, F \rangle$ un automate déterministe complet qui accepte K.
Alors $\mathcal{A}' = \langle Q \uplus S, \Sigma, Z, T', q_0 z_0, F \rangle$ avec

$$T' = T \cup \{ \langle qz, \varepsilon, \delta(s_0, q)z \rangle \mid q \in Q, z \in Z \}$$
$$\cup \{ \langle sz, \varepsilon, \delta(s, z)\varepsilon \rangle \mid s \in S, z \in Z \}$$

Attention, la phase vérification détruit la pile !

Automates à pile et grammaires

Remarque: Dans la suite on ne traite que les grammaires algébriques.

Proposition : Grammaire vers automate à pile

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

Preuve (Idée)

On construit l'automate à pile simple non déterministe acceptant par pile vide : $\mathcal{A} = \langle \Sigma, \Sigma \cup V, T, S \rangle$ dont les transitions T sont

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

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

- $\Rightarrow$: 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): $\langle q, z, q' \rangle \to_G^* w$ ssi $qz \xrightarrow{w}_{\mathcal{A}}^* q'$
(par récurrence sur longueur de dérivation resp. calcul)

Clôture et réduction

Soit Γ un alphabet, $\overline{\Gamma} = \{\overline{a} \mid a \in \Gamma\}$ une copie de Γ et $\widetilde{\Gamma} = \Gamma \uplus \overline{\Gamma}$.
On définit la réduction sur $\widetilde{\Gamma}^*$ par $\overline{a}a \xrightarrow{\text{red}} \varepsilon$ pour $a \in \Gamma$.
Pour $L \subseteq \widetilde{\Gamma}^*$ on pose $\mathrm{Clot}(L) = \{\, w \in \widetilde{\Gamma}^* \mid \exists v \in L : v \xrightarrow[*]{\text{red}} w \,\}$.

Lemme : Reconnaissabilité de la clôture

Si $L \subseteq \widetilde{\Gamma}^*$ est un langage reconnaissable alors $\mathrm{Clot}(L) \subseteq \widetilde{\Gamma}^*$ aussi.
On peut construire un automate pour $\mathrm{Clot}(L)$ à partir d'un automate pour L (PTime).

Configurations accessibles

Définition : Configurations accessibles

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un automate à pile.
Soit $L \subseteq QZ^*$ un ensemble de configurations, on note

$$post(L) = \{\, qh \in QZ^* \mid \exists pg \in L : pg \rightarrow^+ qh \,\}$$

l'ensemble des configurations accessibles à partir de celles de L.

Exemple : Automate $\mathcal{A}_1$

$post(\{q_0\#\}) = \{\, q_0 A^n \# \mid n \geq 0 \,\} \cup \{\, q_1 A^n \# \mid n \geq 0 \,\} \cup \{q_2\}$

Configurations accessibles

Proposition : Reconnaissabilité des configurations accessibles

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un AAP et $L \subseteq QZ^*$ reconnaissable.
On peut effectivement construire un automate fini $\mathcal{B}$ qui reconnaît $post(L)$.

Lemme :

On définit $\Gamma = Q \uplus Z$ et le langage fini $K = \{\, qh\overline{x}\,\overline{p} \mid \langle p, x, a, q, h\rangle \in T \,\} \subseteq \widetilde{\Gamma}^+$.
Soit $n \geq 0$. Il existe un calcul $pg \xrightarrow{}{}^{n} qh$ dans $\mathcal{A}$ ssi il existe $w \in K^n$ t.q. $wpg \xrightarrow{\text{red}}_{2n} qh$

Corollaire : $post(L) = \mathrm{Clot}(K^+L) \cap QZ^*$.

Puisque K est un langage fini, le langage K^+L est reconnaissable et on peut construire (PTIME) un automate $\mathcal{B}$ qui reconnaît $\mathrm{Clot}(K^+L) \cap QZ^*$.

Par analogie, $pre(L) = \{\, qh \in QZ^* \mid \exists pg \in L : qh \to^+ pg \,\}$ est reconnaissable.

Calculs d'accessibilité dans une grammaire

Proposition : Reconnaissabilité des configurations précédentes

Soit $G = \langle \Sigma, V, P, S \rangle$ une grammaire algébrique.
Soit $L \subseteq (\Sigma \cup V)^*$ reconnaissable (par un automate $\mathcal{A}$).
On note $pre(L) = \{ \alpha \in (\Sigma \cup V)^* \mid \exists \beta \in L : \alpha \rightarrow_G^* \beta \}$.

On peut construire (en PTIME) un automate fini $\mathcal{B}$ qui reconnaît $pre(L)$.

Preuve : Voir transparent suivant.

Corollaire : Les problèmes suivants sont décidables en PTIME:

- mot: pour $w \in \Sigma$, $w \in \mathcal{L}(G)$?
- variable productive: pour $A \in V$, existe-il $w \in \Sigma^*$ t.q. $A \rightarrow^* w$?
- mot vide : pour $A \in V$, $A \rightarrow^* \varepsilon$?
- $\mathcal{L}(G)$ est-il fini ?

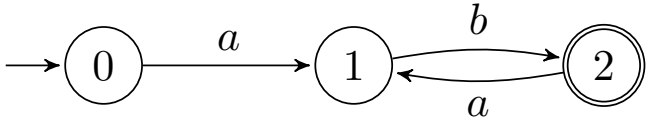
Automate pour $pre(L)$

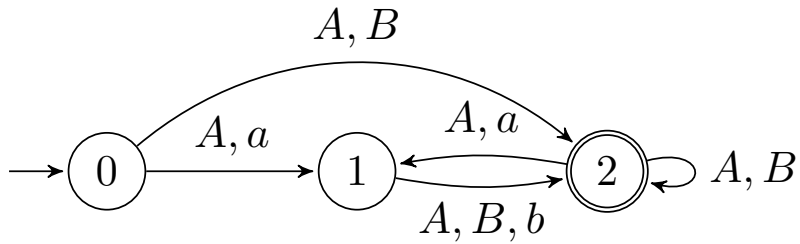
Proposition :

Soit $\mathcal{A} = \langle Q, \Sigma \cup V, \delta, q_0, F \rangle$ un automate reconnaissant L.
Construire $\mathcal{B}$ en ajoutant (itérativement) des transitions selon la règle suivante :

$$\text{Si } \langle A, \beta \rangle \in P \text{ et } q \xrightarrow{\beta}{}^*_{\mathcal{B}} q', \text{ ajouter } \langle q, A, q' \rangle \text{ dans } \mathcal{B}.$$

Exemple: $A \to a \mid BB, \quad B \to AB \mid b$





Preuve (idée): $L(\mathcal{B}) = pre(\mathcal{L}(\mathcal{A}))$

- $\subseteq$: récurrence sur nombre de transitions ajoutées
  Clairement, l'inclusion tient après 0 transition ajoutées. Supposons que l'inclusion tient après i transitions, que $\langle q, A, q' \rangle$ est ajouté dans l'étape $i+1$ et que $q_0 \xrightarrow{\alpha_0}{}^* q \xrightarrow{A} q' \xrightarrow{\alpha_1}{}^* \cdots \xrightarrow{\alpha_{n-1}}{}^* q \xrightarrow{A} q' \xrightarrow{\alpha_n}{}^* q_f$ est un chemin acceptant en conséquence. Du coup il existe $A \to \beta$ dans G tel que $w = \alpha_0 \beta \alpha_1 \cdots \beta \alpha_n$ était accepté après i ajouts, et $\alpha_0 A \alpha_1 \cdots A \alpha_n$ est prédécesseur de w.
- $\supseteq$: récurrence sur longueur de dérivation

AAP régulier

Remarques: Dans ce transparent et le suivant seulement:

- Σ' note $\Sigma \cup \{\varepsilon\}$;
- l'état et sommet de pile sont notés à droite !

Définition : AAP régulier

$\mathcal{A} = \langle Q, \Sigma, Z, T, q_0, F \rangle$, avec $|T|$ fini et

$$T \subseteq (\mathrm{Rec}(Z^*) \times Q \times \Sigma' \times Z \times Q) \cup (\mathrm{Rec}(Z^*) \times Q \times \Sigma' \times Q)$$

1. $wq \xrightarrow{a} wzq'$ si $\langle L, q, a, z, q' \rangle \in T$ et $w \in L$ (push)
2. $wzq \xrightarrow{a} wq'$ si $\langle L, q, a, q' \rangle \in T$ et $wz \in L$ (pop)

AAP régulier $\rightarrow$ AAP ordinaire

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0, F \rangle$ un AAP régulier avec $k := |T|$
et $\forall i : \mathcal{A}_i = \langle Q_i, Z, \delta_i, \iota_i, F_i \rangle$ DCA acceptant les langages dans T. Définissons:

- $\mathcal{Q} := Q_1 \times \cdots \times Q_k, \quad \iota := \langle \iota_1, \ldots, \iota_k \rangle$
- $\mathcal{F}_i := \{ \langle q_1, \ldots, q_k \rangle \in \mathcal{Q} \mid q_i \in F_i \}$
- $\delta : \mathcal{Q} \times Z \rightarrow \mathcal{Q}$ avec $\delta(\langle q_1, \ldots, q_k \rangle, z) := \langle \delta_1(q_1, z), \ldots, \delta_k(q_k, z) \rangle$.

Construction d'un AAP ordinaire équialent à $\mathcal{A}$

$\mathcal{A}' := \langle \mathcal{Q} \times Q, \Sigma, \mathcal{Q} \times Z, T', \langle \iota, q_0 \rangle, \mathcal{Q} \times F \rangle$, avec:

- (push) pour tout $\langle L_i, q, a, z, q' \rangle \in T$ et $f \in \mathcal{F}_i$, on a $\langle \langle f, q \rangle, a, \langle f, z \rangle, \langle \delta(f, z), q' \rangle \rangle \in T'$;
- (pop) pour tout $\langle L_i, q, a, q' \rangle \in T$, $z \in Z$, $q'' \in \mathcal{Q}$ et $f \in F_i$, on a $\langle \langle q'', z \rangle, \langle f, q \rangle, a, \langle q'', q' \rangle \rangle \in T'$.

Invariante: Si $\mathcal{A}$ accède à une configuration $z_1 \cdots z_n q$, alors $\mathcal{A}'$ accède à $\langle q'_0, z_1 \rangle \cdots \langle q'_{n-1}, z_n \rangle \langle q'_n, q \rangle$, avec $q'_0 = \iota$ et $q'_{i+1} = \delta(q'_i, z_{i+1})$ pour $i = 0, \ldots, n-1$.

Arbres de dérivation

Définition :

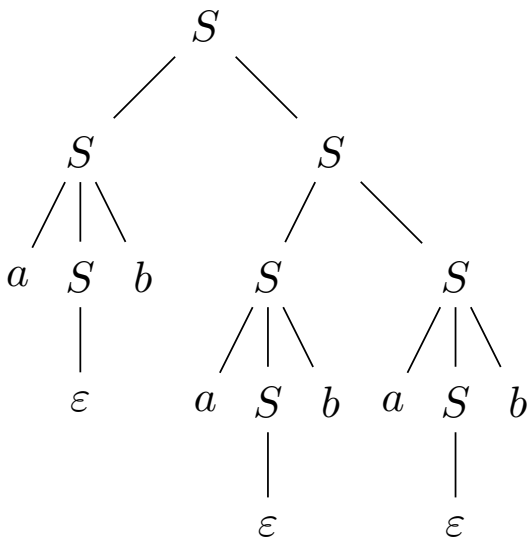
Soit $G = (\Sigma, V, P, S)$ une grammaire.

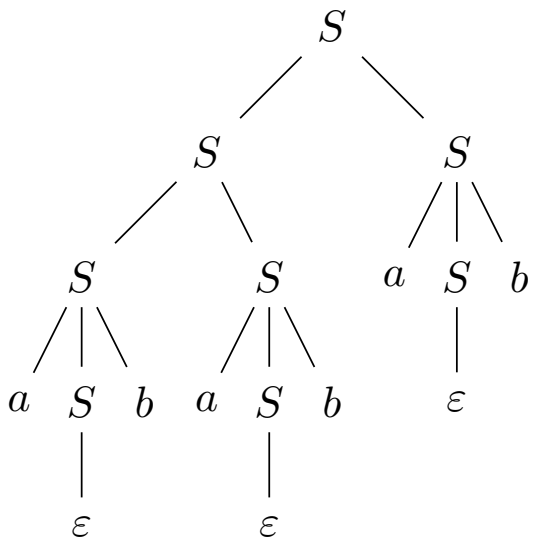
Un arbre de dérivation pour G est un arbre t étiqueté dans $V \cup \Sigma \cup \{\varepsilon\}$ tel que chaque sommet interne u est étiqueté par une variable $x \in V$ et si les fils de u portent les étiquettes $\alpha_1, \ldots, \alpha_k$ alors $(x, \alpha_1 \cdots \alpha_k) \in P$.

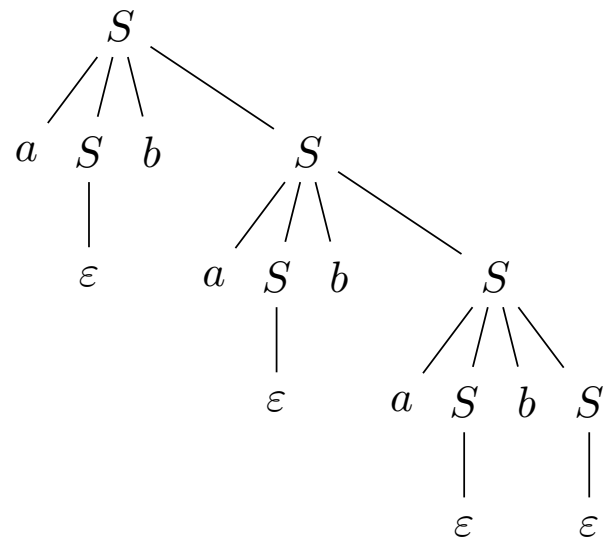
De plus, si $k > 1$, on peut supposer $\alpha_1, \ldots, \alpha_k \neq \varepsilon$.

Exemple :

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







Ambigüité

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 : Grammaires G_1 et G_2

G_1 et G_2 engendrent le même langages, mais G_1 est ambigüe alors que G_2 ne l'est pas.

Remarques :

- Tout arbre de dérivation peut être associé avec une dérivation gauche (resp. droite) et inversement.
- Dans une grammaire non ambigüe, tout mot engendré possède donc une dérivation gauche (resp. droite) unique.

Forme normale de Chomsky

Définition : FNC

Soit $G = (\Sigma, V, P, S)$ une grammaire. G est dite en *forme normale de Chomsky* (FNC) si $P \subseteq (V \times ((V \setminus \{S\})^2 \cup \Sigma)) \cup \{S \to \varepsilon\}$; autrement dit, toute production est de la forme (i) $A \to BC$, (ii) $A \to a$ ou (iii) $S \to \varepsilon$.

Théorème : Conversion en FNC

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

Preuve (idée):

1. Introduire une nouvelle variable initiale S_0 et $S_0 \to S$.
2. Pour tout $a \in \Sigma$, introduire variable V_a; remplacer toute occurrence de a dans les productions par V_a et ajouter $V_a \to a$.
3. Limiter la longueur de la côte droite de toute production à deux (p.ex. remplacer $A \to BCD$ par $A \to C'D$ et $C' \to BC$).
4. Pour toute variable B telle que $B \to_G^* \varepsilon$ et toute production $A \to BC$ ou $A \to CB$, ajouter $A \to C$.
5. Supprimer toute production $A \to \varepsilon$, mais ajouter $S_0 \to \varepsilon$ si $\varepsilon \in \mathcal{L}(G)$.
6. Pour toute paire $A \to B$ et $B \to \beta$, ajouter $A \to \beta$; itérer cette étape.
7. Supprimer toute production de la forme $A \to B$.

FNC: Exemple

Exemple : $S \to aSbS \mid \varepsilon$

1. Ajouter $S_0 \to S$.
2. On obtient $S_0 \to S$, $A \to a$, $B \to b$, $S \to ASBS \mid \varepsilon$.
3. Remplacer $S \to ASBS$ par $S \to AU$, $U \to SV$, $V \to BS$.
4. S engendre ε, ajouter $U \to V$ et $V \to B$.
5. Supprimer $S \to \varepsilon$, ajouter $S_0 \to \varepsilon$. Il nous reste

$$S_0 \to S \mid \varepsilon,\ S \to AU,\ A \to a,\ B \to b,\ U \to SV \mid V,\ V \to BS \mid B$$

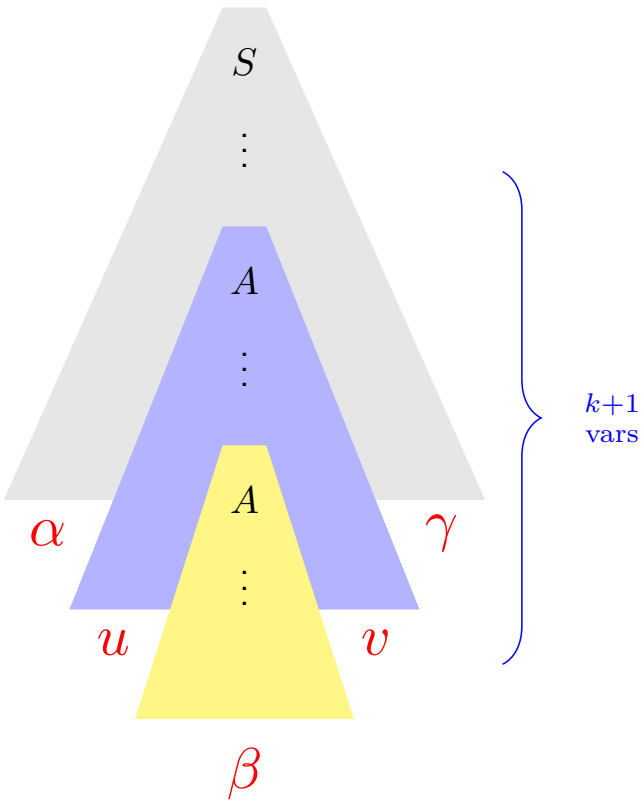
6. Ajouter $V \to b$, $U \to BS \mid B \mid b$, $S_0 \to AU$.
7. Après suppression des productions unitaires, il nous reste

$$S_0 \to AU \mid \varepsilon,\ S \to AU,\ A \to a,\ B \to b,\ U \to SV \mid BS \mid b,\ V \to BS \mid b$$

Lemme d'itération

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

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



Preuve (idée): 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, et une variable A se répète parmi les $k+1$ dernières. Du coup il existe des dérivations $S \to^* \alpha A \gamma$, $A \to^* uAv$ et $A \to^* \beta$. D'ailleurs, puisque G est FNC, on a $|uv| > 0$ et $|u\beta v| \leq N$. Par ailleurs, G engendre $\alpha u^n \beta v^n \gamma$ en appliquant $A \to^* uAv$ exactement n fois, puis $A \to^* \beta$.

Exemple :

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

Par contraposée, pour N donné, un étudie $w := a^N b^N c^N$. Quelque soit sa factorisation $\alpha u \beta v \gamma$ avec $|uv| > 0$ et $|u\beta v| \leq N$, on trouve que u et v ne contiennent qu'au plus deux lettres parmi a, b, c. Alors $\alpha\beta\gamma \notin L_1$ (pour $i = 0$).

Notons que L_1 est l'intersection de $\{ a^n b^n c^p \mid n, p \geq 0 \}$ et $\{ a^p b^n c^n \mid n, p \geq 0 \}$, deux langages algébriques.

Corollaire :

Les langages algébriques ne sont pas fermés par intersection ou complémentaire.