Langages formels

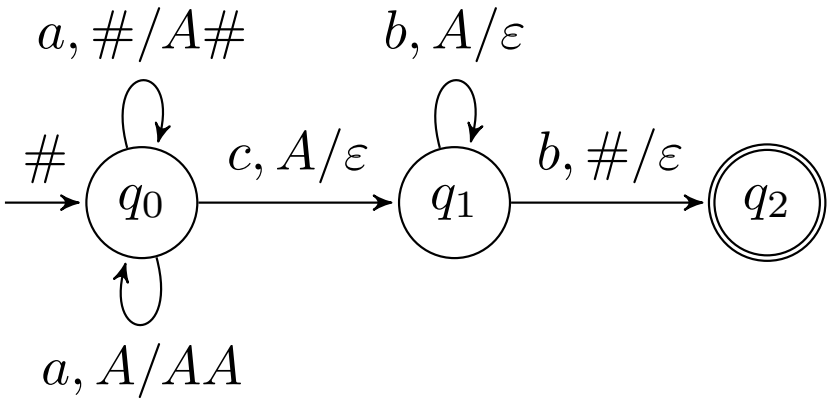
L3 Informatique/Math-Info ENS Paris-Saclay
2024-2025

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:

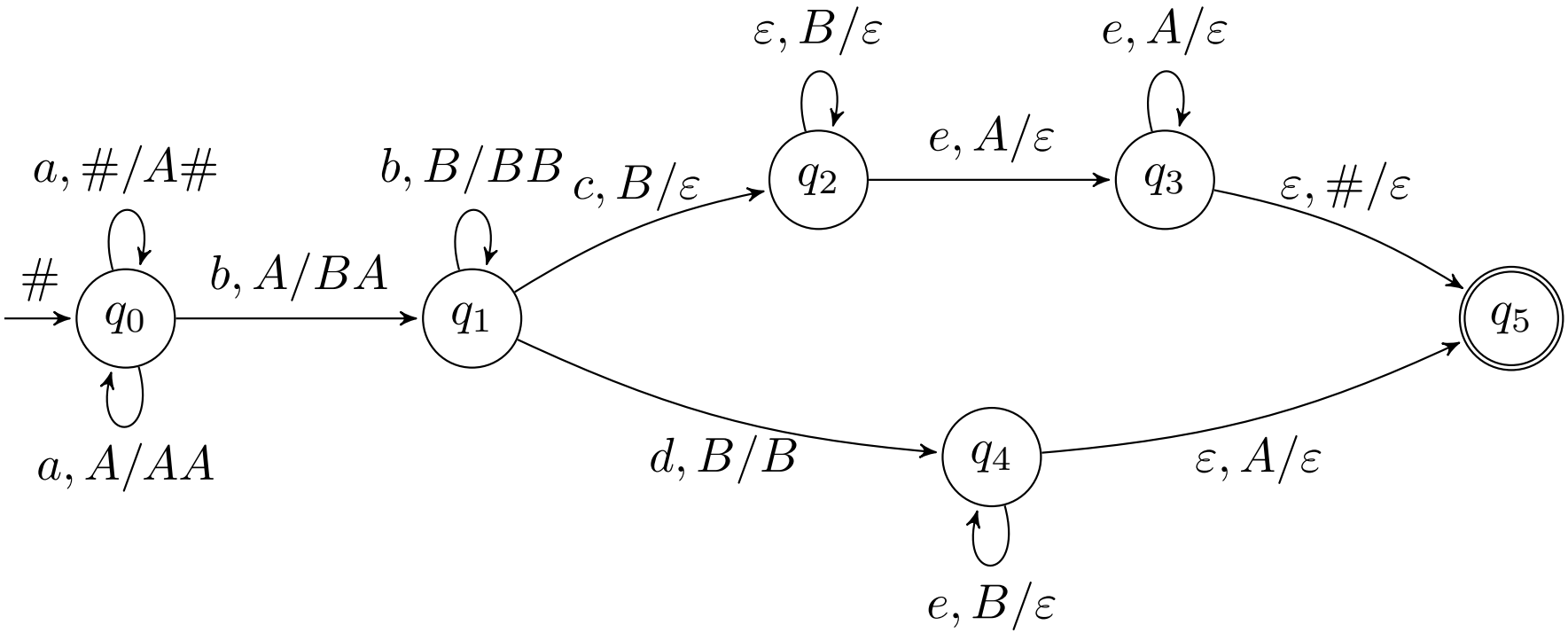
$$q_0\# \xrightarrow{a} q_0 A\# \xrightarrow{a} q_0 AA\# \xrightarrow{c} q_1 A\# \xrightarrow{b} q_1\# \xrightarrow{b} q_2$$

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 ?

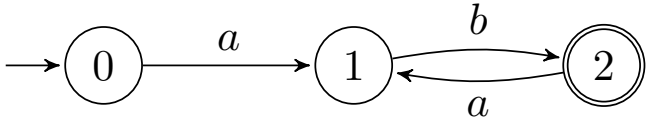
Automate pour $pre(L)$

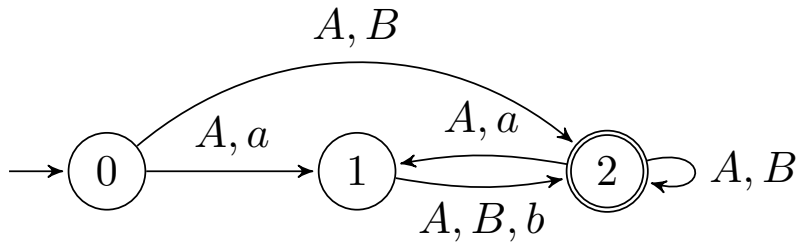
Proposition :

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

- $\subseteq$: 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 :

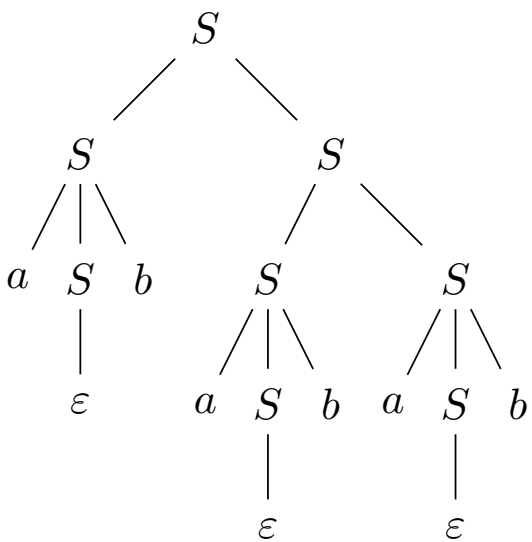
Soit $G = (\Sigma, V, P, S)$ une grammaire.

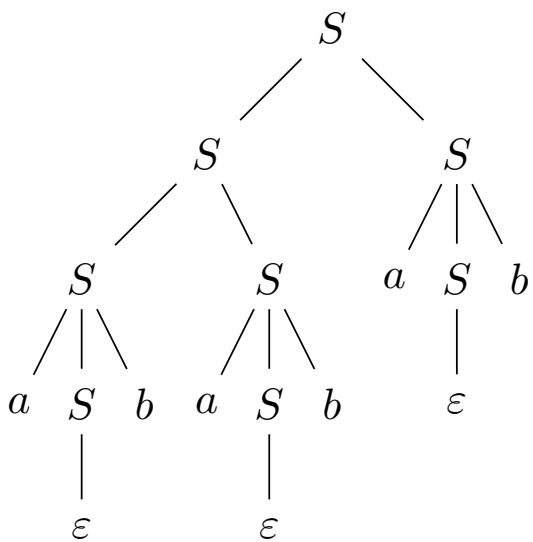
Un arbre de dérivation pour G est un arbre t étiqueté dans $V \cup \Sigma \cup \{\varepsilon\}$ tel que chaque sommet interne u est étiqueté par une variable $x \in V$ et si les fils de u portent les étiquettes $\alpha_1, \ldots, \alpha_k$ alors $(x, \alpha_1 \cdots \alpha_k) \in P$.

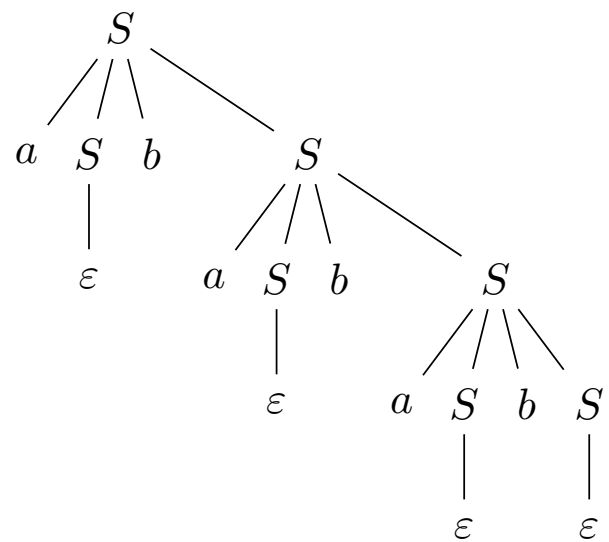
De plus, si $k > 1$, on peut supposer $\alpha_1, \ldots, \alpha_k \neq \varepsilon$.

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

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

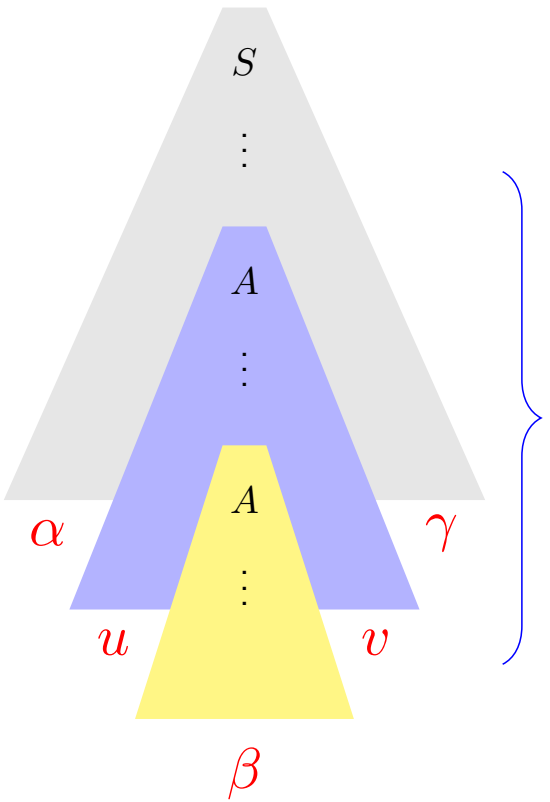
6. Ajouter $V \to b$, $U \to BS \mid B \mid b$, $S_0 \to AU$.
7. Après suppression des productions unitaires, il nous reste

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

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. Dans un chemin de longueur maximale, 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.

Langages déterministes

Définition : Automate à pile déterministe

$\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ est *déterministe* si

- $\forall (pz, a) \in QZ \times (\Sigma \cup \{\varepsilon\}) : |T(pz, a)| \leq 1$,
- $\forall\, pz \in QZ : T(pz, \varepsilon) \neq \emptyset \implies \forall\, a \in \Sigma : T(pz, a) = \emptyset$

Un langage $L \subseteq \Sigma^*$ est *déterministe* s'il existe un AAP déterministe qui accepte L par état final.

Remarque: Un AAPD $\mathcal{A}$ admet, pour tout $w \in \Sigma^*$, au plus un calcul maximal (fini ou infini).

Exemples :

1. Le langage $L_1 = \{a^n b^n \mid n > 0\} \cup \{a^n b^{2n} \mid n > 0\}$ est non ambigu mais pas déterministe.
2. Le langage $L_2 = \{a^n b^p c^n \mid n, p > 0\} \cup \{a^n b^p d^p \mid n, p > 0\}$ est déterministe mais pas D+TR.

Complémentaire

Théorème : Les déterministes sont fermés par complémentaire.

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ un AAP déterministe. On peut effectivement construire un automate à pile déterministe $\mathcal{A}'$ qui reconnaît $\Sigma^* \setminus \mathcal{L}(\mathcal{A})$.

Dans la construction de $\mathcal{A}'$, il y a deux difficultés principales :

1. Un automate déterministe peut se bloquer (deadlock) ou entrer dans un ε-calcul infini (livelock). Dans ce cas il y a des mots qui n'admettent aucun calcul dans l'automate.
2. À la fin d'un mot, un AAPD peut enchaîner des ε-transitions et ainsi passer par plusieurs états, certains acceptants et d'autres non.

Blocage

Définition : Blocage

Un AAP $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ est sans blocage si pour toute configuration accessible $p\alpha$ et pour toute lettre $a \in \Sigma$ il existe un calcul $p\alpha \xrightarrow{\varepsilon}{}^{*} \xrightarrow{a}$.

Proposition : Critère d'absence de blocage

Un automate *déterministe* est sans blocage si et seulement si pour toute configuration accessible $p\alpha$ on a

1. $\alpha \neq \varepsilon$, et donc on peut écrire $\alpha = x\beta$ avec $x \in Z$,
2. $px \xrightarrow{\varepsilon}$ ou $\forall a \in \Sigma$, $px \xrightarrow{a}$,
3. $\neg(px \xrightarrow{\varepsilon}{}^{\omega})$.

De plus, ce critère est décidable.

Remarque :

Si $\mathcal{A}$ est sans blocage, alors chaque mot $w \in \Sigma^*$ admet un calcul maximal unique (et fini) $q_0 z_0 \xrightarrow{w}{}^{*} p\alpha$ avec $\alpha \neq \varepsilon$ et $p\alpha \not\xrightarrow{\varepsilon}$.

Rappel: Calculs d'accessibilité

Corollaire :

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0 \rangle$ un automate à pile.
On peut effectivement calculer les ensembles suivants :

1. $X = \{ \langle p, x, q \rangle \in Q \times Z \times Q \mid px \to^+ q \}$
2. $Y = \{ \langle p, x, q, y \rangle \in Q \times Z \times Q \times Z \mid px \to^+ qy \}$
3. $W = \{ \langle p, x, q, y \rangle \in Q \times Z \times Q \times Z \mid \exists h : px \to^+ qyh \}$
4. $X' = \{ \langle (p, x, q \rangle \in Q \times Z \times Q \mid px \xrightarrow{\varepsilon}{}^+ q \}$
5. $Y' = \{ \langle p, x, q, y \rangle \in Q \times Z \times Q \times Z \mid px \xrightarrow{\varepsilon}{}^+ qy \}$
6. $W' = \{ \langle p, x, q, y \rangle \in Q \times Z \times Q \times Z \mid \exists h : px \xrightarrow{\varepsilon}{}^+ qyh \}$

Preuve

1. $\langle p, x, q \rangle \in X$ ssi $q \in post(px)$.
2. $\langle p, x, q, y \rangle \in Y$ ssi $qy \in post(px)$.
3. $\langle p, x, q, y \rangle \in W$ ssi $post(px) \cap qyZ^* \neq \emptyset$.

Pour X', Y', Z', on applique ce qui précède à l'automate obtenu à partir de $\mathcal{A}$ en ne conservant que les ε-transitions.

Suppression des blocages

Proposition : Suppression des blocages

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ un AAPD, on peut effectivement construire un automate à pile déterministe *sans blocage* $\mathcal{A}' = \langle Q', \Sigma, Z', T', q'_0 z'_0, F' \rangle$ qui reconnaît le même langage.

Preuve

$Q' = Q \uplus \{q'_0, d, f\}$, $F' = F \uplus \{f\}$, $Z' = Z \uplus \{\bot\}$, $z'_0 = \bot$ et pour $p \in Q$, $a \in \Sigma$ et $x \in Z$

1. $q'_0 \bot \xrightarrow{\varepsilon} q_0 z_0 \bot$;
2. Si $px \xrightarrow{a} q\alpha \in T$ alors $px \xrightarrow{a} q\alpha \in T'$;
3. Si $px \not\xrightarrow{\varepsilon}$ et $px \not\xrightarrow{a}$ dans $\mathcal{A}$ alors $px \xrightarrow{a} dx \in T'$;
4. Si $px \not\xrightarrow{\varepsilon}{}^{\omega}$ dans $\mathcal{A}$ et $px \xrightarrow{\varepsilon} q\alpha \in T$, alors $px \xrightarrow{\varepsilon} q\alpha \in T'$;
5. Si $px \xrightarrow{\varepsilon}{}^{\omega}$ dans $\mathcal{A}$ et $\exists q\alpha \in FZ^* : px \xrightarrow{\varepsilon}{}^{*} q\alpha$, alors $px \xrightarrow{\varepsilon} fx \in T'$;
6. Si $px \xrightarrow{\varepsilon}{}^{\omega}$ dans $\mathcal{A}$ et $\forall q\alpha : px \xrightarrow{\varepsilon}{}^{*} q\alpha$ on a $q \notin F$, alors $px \xrightarrow{\varepsilon} dx \in T'$;
7. $p\bot \xrightarrow{\varepsilon} d\bot$, $d\bot \xrightarrow{a} d\bot$, $dx \xrightarrow{a} dx$ et $fx \xrightarrow{a} dx$.

Cette construction est effective.

Complémentaire (construction)

Preuve

On suppose $\mathcal{A}$ déterministe et sans blocage, et on pose $Q' = Q \times \{1,2,3,4\}$.
L'état initial est $q'_0 = \langle q_0, 1\rangle$ si $q_0 \notin F$ et $q'_0 = \langle q_0, 2\rangle$ sinon.

1. Si $px \xrightarrow{\varepsilon} q\alpha \in T$ et $i \in \{1,2\}$ alors
$$\langle p,i\rangle x \xrightarrow{\varepsilon} \langle q,j\rangle\alpha \in T' \text{ avec } j = \begin{cases} 1 & \text{si } i = 1 \wedge q \notin F \\ 2 & \text{sinon.} \end{cases}$$
2. Si $px \xrightarrow{a} q\alpha \in T$ et $i \in \{1,2\}$ alors $\langle p,i\rangle x \xrightarrow{\varepsilon} \langle p,i+2\rangle x \in T'$ et
$$\langle p,i+2\rangle x \xrightarrow{a} \langle q,j\rangle\alpha \in T' \text{ avec } j = \begin{cases} 1 & \text{si } q \notin F \\ 2 & \text{sinon.} \end{cases}$$

L'automate $\mathcal{A}'$ est déterministe et sans blocage.
L'unique calcul *maximal* de $\mathcal{A}'$ sur $w \in \Sigma^*$ s'écrit $q'_0 z_0 \xrightarrow{w}{}^* \langle p,j\rangle\alpha$ avec $j = 3$ si le calcul de $\mathcal{A}$ n'a pas visité F depuis la dernière lettre, et $j = 4$ sinon.

- Avec $F' = Q \times \{3\}$ on obtient $\mathcal{L}(\mathcal{A}') = \Sigma^* \setminus \mathcal{L}(\mathcal{A})$.
- Avec $F' = Q \times \{4\}$ on obtient $\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A})$.

Langages déterministes: Divers

Remarque : Propriétés de la construction précédente

- Dans l'automate obtenu il n'y a pas d'ε-transition à partir d'un état de F'.
- De plus, chaque mot $w \in \mathcal{L}(\mathcal{A}')$ a un unique calcul acceptant dans $\mathcal{A}'$.

Exercice :

Montrer que tout langage déterministe est non ambigu.

Exercice :

Soit $\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, K \rangle$ un AAPD avec acceptation généralisée par le langage rationnel $K \subseteq QZ^*$. Montrer qu'on peut effectivement construire un AAPD équivalent reconnaissant par état final.

Résumé: Propriétés de clôture

Proposition :

Les langages algébriques :

- sont clôturés par union ;
- ne sont pas clôturés par complément (voir $\{ ww \mid w \in \Sigma^* \}$) ni intersection.

Proposition :

Les langages déterministes :

- sont clôturés par complément ;
- ne sont pas clôturés par intersection (voir $\{ a^n b^p c^n \mid n, p \geq 1 \}$, $\{ a^n b^n c^p \mid n, p \geq 1 \}$) ni union ;
- sont strictement moins expressif que les algébriques en général.

Problèmes indécidables

Proposition :

Soient L, L' deux langages algébriques et R un langage rationnel.
Les problèmes suivants sont indécidables :

- $L \cap L' = \emptyset$?
- L est-il ambigu ?
- $L = \Sigma^*$?
- $L = L'$?
- $L \subseteq L'$?
- $R \subseteq L$?
- L est-il rationnel ?
- L est-il déterministe ?
- $\overline{L}$ est-il algébrique ?
- $L \cap L'$ est-il algébrique ?

Langages déterministes

Proposition : Décidabilité et indécidabilité

On ne peut pas décider si un langage algébrique est déterministe.

Soient L, L' deux langages déterministes et R un langage rationnel.
Les problèmes suivants sont décidables :

- $R \subseteq L$?
- $L = R$?
- $L = L'$?
- L est-il rationnel ?

$$R \subseteq L \Longleftrightarrow R \cap \overline{L} = \emptyset$$

$$R = L \Longleftrightarrow R \cap \overline{L} = \emptyset = \overline{R} \cap L$$

Géraud Sénizergues, prix Gödel 2002

Les problèmes suivants sont indécidables :

- $L \cap L' = \emptyset$?
- $L \subseteq L'$?
- $L \cap L'$ est-il algébrique ?
- $L \cap L'$ est-il déterministe ?
- $L \cup L'$ est-il déterministe ?

Automates à pile visible

Nous allons étudier une sous-classe des langages déterministes TR qui récupère de propriétés intéressantes non satisfaites par les algébriques et déterministes en général (clôture par intersection et complément, déterminisation).

Remarque : Rappel

Le langage $L_2 = \{a^n b^k c^n \mid n, k > 0\} \cup \{a^n b^k d^k \mid n, p > 0\}$ est déterministe mais pas D+TR.

Preuve : Voir lemme de pompage d'Igarashi (1985).

Définition : Alphabet visible

Un *alphabet visible* est un triplet d'alphabets $\Sigma = \langle \Sigma_2, \Sigma_1, \Sigma_0 \rangle$.

Dans la suite, Σ note un tel alphabet visible.

Définition : AAP visible

$\mathcal{A} = \langle Q, \Sigma, Z, T, q_0 z_0, F \rangle$ où $T \subseteq \bigcup_{i=0}^{2} (Q \times \Gamma) \times \Sigma_i \times (Q \times \Gamma^i)$
Un langage est *visible* s'il existe un AAPV qui l'accepte.

AAPV: Exemples

Exemples:

- $L_1 = \{ a^n b^n c^p \mid n, p > 0 \}$ est accepté par un AAPV avec $\Sigma_1 = \langle \{a\}, \{c\}, \{b\} \rangle$;
- $L_2 = \{ a^n b^p c^p \mid n, p > 0 \}$ est accepté par un AAPV avec $\Sigma_2 = \langle \{b\}, \{a\}, \{c\} \rangle$.

On considère que $\Sigma_1 \neq \Sigma_2$!

Exemple:

- $L_1 \cup L_2$ est bien algébrique mais pas visible.
- $L_3 = \{ w \in \{a, b\}^* \mid |w|_a = |w|_b \}$ est déterministe TR mais pas visible.

Union et intersection des AAPV

Dans la suite, on fixe un alphabet Σ visible.

Proposition : Les langages visibles sur Σ sont clôturés par union.

Par juxtaposition, comme pour les algébriques.

Proposition : Les langages visibles sur Σ sont clôturés par intersection.

Par produit ; soit $\mathcal{A} = \langle Q, \Sigma, Z, T_1, q_0 z_0, F \rangle$ et $\mathcal{B} = \langle R, \Sigma, Y, T_2, r_0 y_0, G \rangle$, alors soit $\mathcal{A} \times \mathcal{B} = \langle Q \times R, Z \times Y, T, \langle q_0, r_0 \rangle \langle z_0, y_0 \rangle, F \times G \rangle$ avec :

$$\frac{a \in \Sigma_2 \quad \langle qz, a, q' z_1 z_2 \rangle \in T_1 \quad \langle ry, a, r' y_1 y_2 \rangle \in T_2}{\langle \langle q, r \rangle \langle z, y \rangle, a, \langle q', r' \rangle \langle z_1, y_1 \rangle \langle z_2, y_2 \rangle \rangle}$$

Les règles pour Σ_1 et Σ_0 sont analogues. On remarque que la hauteur de la pile après avoir lu un mot w ne dépend que de Σ et de w:

Proposition : Les langages visibles sur Σ sont clôturés par intersection.

Par déterminisation, voir suite.

Déterminisation des AAPV

Déterminisation

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0, F)$ un AAPV qui accepte le langage L.
Il existe un AAPV $\mathcal{A}'$ déterministe qui accepte L.

Soit $\mathcal{A} := \langle Q, Z, T, q_0 z_0, F \rangle$. On construit donc $\mathcal{A}' := \langle Q, Z', T', q'_0 \bot, F' \rangle$ avec

- $Q' := 2^Q \times 2^{Q \times Q} \times Z$;
- $Z' := (Q' \times \Sigma_2 \times Z) \cup \{\bot\}$;
- $q'_0 := \langle \{q_0\}, \{ \langle q, q \rangle \mid q \in Q \}, z_0 \rangle$;
- $F' := \{ \langle R, S \rangle \mid R \cap F \neq \emptyset \}$;
- $T := T_2 \cup T_1 \cup T_0 \cup T_\bot$.

Suite déterminisation

Les transitions de $\mathcal{A}'$ sont défini ainsi :

- Pour les lettres de Σ_1 :

$$T_1 := \{\, \langle\langle R, S, z_1\rangle z', a, \langle R', S', z_2\rangle z'\rangle \mid \\ R' := \{\, q' \mid q \in R, \langle q z_1, a, q' z_2\rangle \in T \,\},\ z' \in Z', \\ S' := \{\, \langle q'', q'\rangle \mid \exists q : \langle q'', q\rangle \in S, \langle q z_1, a, q' z_2\rangle \in T \,\} \,\}$$

- Pour les lettres de Σ_2 :

$$T_2 := \{\, \langle\langle R, S, z_1\rangle z', a, \langle R', S', z_2\rangle\langle R, S, z_3, a, z_2\rangle z'\rangle \mid \\ R' := \{\, q' \mid \exists q \in R, \langle q z_1, a, q' z_2 z_3\rangle \in T \,\},\ z' \in Z', \\ S' := \{\, \langle q, q\rangle \mid q \in Q \,\}\}$$

Suite déterminisation

Les transitions de $\mathcal{A}'$ sont défini ainsi :

- Pour les lettres de Σ_0 :

$$
\begin{aligned}
T_0 := \{\, \langle\langle R, S, z\rangle\langle R', S', z_3, a', z_2\rangle, a, \langle R'', S'', z_3\rangle\rangle \mid \\
R'' := \{\, q' \mid \exists q \in R' : \langle q, q'\rangle \in U \,\} \\
S'' := \{\, \langle q, q'\rangle \mid \exists q'' : \langle q, q''\rangle \in S'', \langle q'', q'\rangle \in U \,\} \\
\text{avec } U := \{\, \langle q, q'\rangle \mid \exists q_1, q_2, z : \\
\langle q z_1, a', q_1 z_2 z_3\rangle \in T, \langle q_1, q_2\rangle \in S, \langle q_2 z, a, q'\rangle \in T \,\} \,\}
\end{aligned}
$$

- Pour le cas spécial de $\perp$:

$$
\begin{aligned}
T_\perp := \{\, \langle\langle R, S, z\rangle\perp, a, \langle R', S', \perp\rangle\rangle \mid \\
R' := \{\, q' \mid \exists q \in R, \langle q, a, z, q'\rangle \in T_r \,\}, \\
S' := \{\, \langle q, q''\rangle \mid \exists q' : \langle q, q'\rangle \in S, \langle q' z, a, q''\rangle \in T \,\} \,\}
\end{aligned}
$$

AAPV et MSO

On considère une extension de MSO (qu'on appellera MSO_μ) qui rajoute un prédicat $\mu(x, y)$ donnant vrai ssi x est la position d'un Σ_2 et y la position d'un Σ_0 correspondant. Les autres notions de MSO sont utilisées comme d'habitude. Par exemple, si $\Sigma = \langle\{a\}, \emptyset, \{b\}\rangle$, alors la formule

$$\forall x(P_a(x) \to \exists y(P_b(y) \wedge \mu(x, y)))$$

exige que tout a possède un b correspondant.

Proposition :

Un langage est visible si et seulement s'il est définissable en MSO_μ.

Idée:

$\mathrm{MSO}_\mu \to \mathrm{AAPV}$: La preuve utilise la même construction que pour MSO sur des automates finis, le seul sous-cas à ajouter sont des sous-formules du type $\mu(x, y)$. Pour cela, on construit un AAPV avec $Z = \{z_1, z_2\}$. En lisant une lettre de Σ_2, il empile z_2 ssi x est vrai, et en lisant une lettre de Σ_0, il dépile z_2 ssi y est vrai (et entre dans un état acceptant dans ce cas).

Suite MSO_μ

$\mathrm{AAPV} \to \mathrm{MSO}_\mu$: Un AAPV avec n lettres et k symboles de pile se traduit en une formule ϕ qui :

- tout comme pour les automates finis, ϕ partitionne les positions du mot en n ensembles qui correspondent aux états et vérifie que la suite d'états est en adéquation avec celle des lettres ;
- en plus, ϕ partitionne les position en $3k$ ensembles C_z, E_z, R_z, pour tout $z \in Z$;
- on vérifie que les C_z (E_z, R_z) correspondent aux positions des lettres de Σ_2 (Σ_1, Σ_0) tel que z est au sommet de la pile ;
- pour toute paire x, y, on vérifie que $\mu(x, y)$ implique existence de z, z' t.q. $x \in C_z$, $y \in R_{z'}$, et il existe une transition $\langle qz, a, q'z''z' \rangle$;
- enfin, ϕ assure les propriétés utiles de ϕ (bien imbriqué).

Fonctions séquentielles

Définition : Automates séquentiels et fonctions séquentielles

$\mathcal{A} = \langle Q, A, B, q_0, \odot, \circledast, m, \rho \rangle$ où

- Q est un ensemble fini d'états ;
- A et B sont les alphabets d'*entrée* et de *sortie*,
- $\odot : Q \times A \to Q$ est une fonction *partielle* de transition;
- $\circledast : Q \times A \to B^*$ fonction partielle de sortie avec $\mathrm{dom}(\circledast) = \mathrm{dom}(\odot)$;
- $q_0 \in Q$ est l'état initial et $m \in B^*$ est le préfixe initial ;
- $\rho : Q \to B^*$ est la fonction partielle finale.

Remarques :

- On appelle état final un état dans $\mathrm{dom}(\rho)$.
- L'automate $\langle Q, A, q_0, \odot, \mathrm{dom}(\rho) \rangle$ est déterministe.
- On étend $\odot$ et $\circledast$ à $Q \times A^*$ par
  - $q \odot \varepsilon = q$ et $q \odot ua = (q \odot u) \odot a$
  - $q \circledast \varepsilon = \varepsilon$ et $q \circledast ua = (q \circledast u) \cdot ((q \odot u) \circledast a)$

Sémantique des fonctions séquentielles

Définition : Sémantique : $[\![\mathcal{A}]\!] : A^* \to B^*$

La sémantique de $\mathcal{A}$ est la fonction partielle $[\![\mathcal{A}]\!] : A^* \to B^*$ définie par

- $[\![\mathcal{A}]\!](u) = m \cdot (q_0 \circledast u) \cdot \rho(q_0 \odot u)$.

$f : A^* \to B^*$ est *séquentielle* s'il existe un automate séquentiel $\mathcal{A}$ tel que $f = [\![\mathcal{A}]\!]$.

Exemples :

1. Transformation d'un texte en majuscules.
2. Codage et décodage avec le code préfixe défini par

$$a \mapsto 0000 \qquad c \mapsto 001 \qquad e \mapsto 011 \qquad g \mapsto 11$$
$$b \mapsto 0001 \qquad d \mapsto 010 \qquad f \mapsto 10$$

3. Division par 3 d'un entier écrit en binaire en commençant par le bit de poids *faible*.
4. La fonction $f : A^* \to A^*$ définie par $f(u) = u(ab)^{-1}$.

Propriétés des fonctions séquentielles

Lemme :

Une fonction séquentielle peut être réalisée par un automate séquentiel ayant un préfixe initial vide ($m = \varepsilon$).

Proposition :

Une fonction séquentielle peut être réalisée par un automate émondé, i.e., tel que $\forall p \in Q$, $\exists u, v \in A^*$ tels que $q_0 \odot u = p$ et $p \odot v \in \mathrm{dom}(\rho)$.

Proposition :

- Le domaine d'une fonction séquentielle est un langage reconnaissable.
- L'image d'une fonction séquentielle est un langage reconnaissable.

Théorème : Image directe, image inverse

Soient $f : A^* \to B^*$ une fonction séquentielle.

- Si $L \subseteq A^*$ est rationnel alors $f(L)$ est rationnel.
- Si $L \subseteq B^*$ est rationnel alors $f(L)$ est rationnel.

Fonctions séquentielles pures

Définition : Automates séquentiels purs (Mealy machine)

Un automate séquentiel $\mathcal{A} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ est pur si

- $m = \varepsilon$: le préfixe initial est vide
- $\rho(q) = \varepsilon$ pour tous $q \in Q$: tous les états sont finaux avec sortie vide.

Une fonction $f : A^* \to B^*$ est séquentielle pure s'il existe un automate séquentiel pur $\mathcal{A}$ qui la réalise : $f = [\![\mathcal{A}]\!]$.

Exemples :

1. Transformation d'un texte en majuscules.
2. Codage avec le code préfixe défini par

$$a \mapsto 0000 \qquad c \mapsto 001 \qquad e \mapsto 011 \qquad g \mapsto 11$$
$$b \mapsto 0001 \qquad d \mapsto 010 \qquad f \mapsto 10$$

3. Division par 3 d'un entier écrit en binaire en commençant par le bit de poids *fort*.

Composition

Théorème : Composition

Soient $f : A^* \to B^*$ et $g : B^* \to C^*$ deux fonctions partielles.

1. Si f et g sont séquentielles alors $g \circ f : A^* \to C^*$ est aussi séquentielle.
2. Si f et g sont séquentielles *pures* alors $g \circ f$ est aussi séquentielle *pure*.

Exemple : Multiplication par 5

Dans cet exemple, $A = C = \{0, 1\}$, $B = \{0, 1\}^2$ et les mots représentent des entiers codés en binaire en commençant par le bit de poids faible.
On considère les fonctions séquentielles $f : A^* \to B^*$ et $g : B^* \to C^*$ définies par $f(n) = (n, 4n)$, i.e., $f(u) = (u00, 00u)$ et $g(n, m) = n + m$.
La fonction $g \circ f$ code la multiplication par 5.
Construire les automates séquentiels réalisant f et g.
En déduire un automate séquentiel pour $g \circ f$.

Produit en couronne

Définition : Produit en couronne

Soient $\mathcal{A} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ et $\mathcal{A}' = (Q', B, C, q'_0, \odot, \circledast, m', \rho')$ deux automates séquentiels.
Le produit en couronne $\mathcal{A}' \circ \mathcal{A} = (Q'', A, C, q''_0, \odot, \circledast, m'', \rho'')$ est défini par

- $Q'' = Q \times Q'$, $q''_0 = (q_0, q'_0 \odot m)$ et $m'' = m' \cdot (q'_0 \circledast m)$,
- $(p, p') \odot a = (p \odot a, p' \odot (p \circledast a))$,
- $(p, p') \circledast a = p' \circledast (p \circledast a)$,
- $\rho''((p, p')) = (p' \circledast \rho(p)) \cdot \rho'(p' \odot \rho(p))$.

Lemme : Extension à A^*

Pour tout $u \in A^*$, on a

- $(p, p') \odot u = (p \odot u, p' \odot (p \circledast u))$,
- $(p, p') \circledast u = p' \circledast (p \circledast u)$.

Preuve (Composition)

1. Si f et g sont réalisées par $\mathcal{A}$ et $\mathcal{A}'$ alors $g \circ f$ est réalisée par $\mathcal{A}' \circ \mathcal{A}$.
2. Si $\mathcal{A}$ et $\mathcal{A}'$ sont purs alors $\mathcal{A}' \circ \mathcal{A}$ est pur.

Plus grand préfixe commun

Définition :

- Tout sous ensemble $\emptyset \neq X \subseteq B^*$ admet un plus grand préfixe commun, i.e., une borne inférieure pour l'ordre préfixe.
  Cette borne inférieure est notée $\bigwedge X$.
- Noter que $\emptyset$ n'admet pas de plus grand préfixe commun.
  Donc $\bigwedge \emptyset$ n'est pas défini.

Remarque :

1. Soit $u \in B^*$ et $\emptyset \neq X \subseteq B^*$.
   - $\bigwedge u \cdot X = u \cdot \bigwedge X$
   - si $u \leq \bigwedge X$ alors $\bigwedge u^{-1} \cdot X = u^{-1} \cdot \bigwedge X$
2. Soit $f : A^* \to B^*$ une fonction partielle, on a $f(A^*) = \{f(u) \mid u \in \mathrm{dom}(f)\}$.

   Donc $\bigwedge f(A^*)$ est défini si $\mathrm{dom}(f) \neq \emptyset$.

Résiduels

Définition : Résiduels

Soit $f : A^* \to B^*$ une fonction partielle et soit $u \in A^*$.

Le résiduel $f_u : A^* \to B^*$ est défini par

- $\mathrm{dom}(f_u) = u^{-1}\mathrm{dom}(f)$ et
- $f_u(v) = (\bigwedge f(uA^*))^{-1} f(uv)$ pour $uv \in \mathrm{dom}(f)$.

$\bigwedge f(uA^*)$ représente tout ce qu'on peut écrire si on sait que la donnée commence par u. Le résiduel $f_u(v)$ est donc ce qui reste à écrire si la donnée est uv.

Exemples : Calculer les résiduels des fonctions suivantes

- "quotient" $f : A^* \to A^*$ définie par $f(w) = w(ab)^{-1}$.
- "copie" $g : A^* \to A^*$ définie par $g(w) = ww$.

Théorème : Caractérisation par résiduels

$f : A^* \to B^*$ est séquentielle si et seulement si elle a un nombre fini de résiduels.

Normalisation

Exemple :

Donner un automate séquentiel réalisant la fonction $f : A^* \to A^*$ définie par $f(a^{2n}b) = (ab)^n a$.

Cet automate devra sortir les lettres du résultat le plus rapidement possible.

Définition : Automate normalisé

Intuitivement, un automate est normalisé s'il écrit son résultat au plus tôt.

Soit $\mathcal{A} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ un automate séquentiel et $p \in Q$.

On définit $\mathcal{A}_p = (Q, A, B, p, \odot, \circledast, \varepsilon, \rho)$ et $m_p = \bigwedge [\![\mathcal{A}_p]\!](A^*)$ si $[\![\mathcal{A}_p]\!](A^*) \neq \emptyset$.

L'automate $\mathcal{A}$ est normalisé si pour tout $p \in Q$, $[\![\mathcal{A}_p]\!](A^*) = \emptyset$ ou $m_p = \varepsilon$.

Proposition : Effectivité

Étant donné un automate séquentiel $\mathcal{A}$, on peut calculer les m_p en temps quadratique (preuve admis).

Normalisation

Proposition : Normalisation

Tout automate séquentiel est équivalent à un automate séquentiel normalisé, qui peut être choisi émondé ou complet.

Preuve

Soit $\mathcal{A} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ un automate séquentiel émondé
(donc m_p est bien défini pour tous $p \in Q$).

On définit $\mathcal{A}' = (Q, A, B, q_0, \odot, \circledast', m', \rho')$ par :

- $m' = m m_{q_0} = \bigwedge [\![\mathcal{A}]\!](A^*)$,
- $p \circledast' a = m_p^{-1}((p \circledast a) \cdot m_{p \odot a})$ si $(p, a) \in \mathrm{dom}(\circledast) = \mathrm{dom}(\odot)$
- $\rho'(p) = m_p^{-1} \rho(p)$ si $p \in \mathrm{dom}(\rho)$

On vérifie que $\mathcal{A}'$ est normalisé et $[\![\mathcal{A}']\!] = [\![\mathcal{A}]\!]$.

Pour obtenir un automate complet, il suffit d'ajouter un état puits.

Résiduels

Théorème : Caractérisation par résiduels

Une fonction $f : A^* \to B^*$ est séquentielle si et seulement si elle a un nombre fini de résiduels.

Lemme :

Soit $\mathcal{A} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ un automate normalisé complet.
Soit $u \in A^*$ et $p = q_0 \odot u$. Alors $f_u = [\![\mathcal{A}_p]\!]$.

On en déduit qu'une fonction séquentielle réalisée par $\mathcal{A}$ a au plus $|Q|$ résiduels.

Lemme : Composition

Soient $u, v \in A^*$. On a $f_{uv} = (f_u)_v$, i.e.,
$\mathrm{dom}(f_{uv}) = v^{-1}\mathrm{dom}(f_u)$ et $f_{uv}(w) = (\bigwedge f_u(vA^*))^{-1} f_u(vw)$.

Automate des résiduels

L'automate des résiduels de f est $\mathcal{R} = (Q, A, B, q_0, \odot, \circledast, m, \rho)$ où

- $Q = \{f_u \mid u \in A^*\}$ (supposé fini pour la réciproque du théorème),
- $q_0 = f_\varepsilon$ et $m = \bigwedge f(A^*)$ si $\mathrm{dom}(f) \neq \emptyset$, et $m = \varepsilon$ sinon,
- $f_u \odot a = (f_u)_a = f_{ua}$,
- $f_u \circledast a = \bigwedge f_u(aA^*)$ si $\mathrm{dom}(f_{ua}) \neq \emptyset$, et $f_u \circledast a = \varepsilon$ sinon,
- $\rho(f_u) = f_u(\varepsilon)$ si $\varepsilon \in \mathrm{dom}(f_u)$, et $f_u \notin \mathrm{dom}(\rho)$ sinon.

Lemme :

1. Soient $u, v \in A^*$. On a $f_u \odot v = f_{uv}$.
2. Soient $u, v \in A^*$. On a $f_u \circledast v = \bigwedge f_u(vA^*)$ si $\mathrm{dom}(f_{uv}) \neq \emptyset$.
3. Soit $u \in A^*$. On a $f_u = [\![\mathcal{R}_{f_u}]\!]$.
4. $f = [\![\mathcal{R}]\!]$.
5. L'automate des résiduels est normalisé, accessible et complet.

Exemple :

Calculer l'automate des résiduels de la fonction *multiplication par 5* où les entiers sont codés en binaire en commençant avec le bit de poids faible.