Rappel: Grammaires

Définition : $G = (\Sigma, V, \to, S)$ où

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

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

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

Intérêt des langages algébriques :

- spécification de langages au-délà des reconnaissables, analyse syntaxique
- correspondent aux *automates à pile*

Automates à pile

Définition : $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0, F)$ 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.

De plus, $\mathcal{A}$ est *temps-réel* s'il n'a pas d'ε-transition.

Définition : Système de transitions (infini) associé

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

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

Propriétés fondamentales

Lemme : fondamental

Soient $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ 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 deux calculs $pg \xrightarrow[n_1]{w_1} q$ et $qh \xrightarrow[n_2]{w_2} r$ de $\mathcal{A}$ avec $q \in Q$, $w = w_1 w_2$ et $n = n_1 + n_2$.

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 $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile et $K \subseteq QZ^*$ un langage reconnaissable. Le langage reconnu par $\mathcal{A}$ avec *acceptation généralisée* K est

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

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 :

$L = \{a^n b^n \mid n \geq 0\}$ 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 $\mathcal{L}_K(\mathcal{A}) = \mathcal{L}(\mathcal{A}')$. Donc, tous les modes d'acceptation ci-dessus sont équivalents.

Automates à pile et grammaires

Proposition : Grammaire vers automate à pile

Soit $G = (\Sigma, V, P, S)$ une grammaire. On peut construire un automate à pile simple (un seul état) $\mathcal{A}$ qui accepte $L_G(S)$ par pile vide.

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

- des expansions : $x \xrightarrow{\varepsilon} \alpha$ avec $(x, \alpha) \in P$,
- ou des vérifications : $a \xrightarrow{a} \varepsilon$ avec $a \in \Sigma$.

Remarques :

- AAP simple: un seul état (non noté dans les transitions)
- dérivation gauche $\rightarrow_g^*$: remplacer toujours la variable la plus à gauche
- dérivation droite $\rightarrow_d^*$: analogue

Preuve (idée) : pour tout $\alpha \in (\Sigma \cup V)^*$ et $w \in \Sigma^*$, prouver $\alpha \rightarrow_g^* w$ 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 $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile reconnaissant par pile vide. On peut construire une grammaire G qui engendre $\mathcal{L}(\mathcal{A})$.

Construction:

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

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

Preuve (idée): $\langle q, z, q' \rangle \rightarrow w$ dans G ssi $qz \xrightarrow{w}{}^{*} q'$ dans $\mathcal{A}$
(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 ($\mathrm{PT{\small IME}}$).

Configurations accessibles

Proposition : Reconnaissabilité des configurations accessibles

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile.
Soit $L \subseteq QZ^*$ un langage reconnaissable, on note

$$\mathcal{C}(L) = \{qh \in QZ^* \mid \exists pg \in L : pg \underset{*}{\rightarrow} qh \text{ dans } \mathcal{T}\}$$

l'ensemble des configurations accessibles à partir de celles de L.
On peut effectivement construire un automate fini $\mathcal{B}$ qui reconnaît $\mathcal{C}(L)$.

On définit $\Gamma = Q \uplus Z$ et le langage fini $K = \{qh\overline{x}\,\overline{p} \mid \exists\ (px, a, qh) \in T\} \subseteq \widetilde{\Gamma}^+$.

Lemme : Soit $n \geq 0$

il existe un calcul $pg \underset{n}{\rightarrow} qh$ dans $\mathcal{A}$ ssi il existe $w \in K^n$ tel que $wpg \xrightarrow[2n]{\text{red}} qh$

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

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

Par analogie, $\overline{\mathcal{C}}(L) = \{qh \in QZ^* \mid \exists pg \in L : qh \underset{*}{\rightarrow} pg \text{ dans } \mathcal{T}\}$ est reconnaissable.

Calculs d'accessibilité

Corollaire :

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

1. $X = \{(p, x, q) \in Q \times Z \times Q \mid px \xrightarrow[+]{} q \text{ dans } \mathcal{T}\}$
2. $Y = \{(p, x, q, y) \in Q \times Z \times Q \times Z \mid px \xrightarrow[+]{} qy \text{ dans } \mathcal{T}\}$
3. $W = \{(p, x, q, y) \in Q \times Z \times Q \times Z \mid \exists h : px \xrightarrow[+]{} qyh \text{ dans } \mathcal{T}\}$
4. $X' = \{(p, x, q) \in Q \times Z \times Q \mid px \xrightarrow[+]{\varepsilon} q \text{ dans } \mathcal{T}\}$
5. $Y' = \{(p, x, q, y) \in Q \times Z \times Q \times Z \mid px \xrightarrow[+]{\varepsilon} qy \text{ dans } \mathcal{T}\}$
6. $W' = \{(p, x, q, y) \in Q \times Z \times Q \times Z \mid \exists h : px \xrightarrow[+]{\varepsilon} qyh \text{ dans } \mathcal{T}\}$

Preuve

1. $(p, x, q) \in X$ ssi $q \in \mathcal{C}(px)$.
2. $(p, x, q, y) \in Y$ ssi $qy \in \mathcal{C}(px)$.
3. $(p, x, q, y) \in W$ ssi $\mathcal{C}(px) \cap qyZ^* \neq \emptyset$.

On applique ce qui précède à l'automate $\mathcal{A}'$ obtenu à partir de $\mathcal{A}$ en ne conservant que les ε-transitions.

Calculs d'accessibilité dans une grammaire

Proposition : Reconnaissabilité des configurations précédentes

Soit $G = \langle \Sigma, V, \rightarrow, S \rangle$ 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 : Corollaires

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. $A \rightarrow_G^* w$?
- $\mathcal{L}(G)$ est-il fini ?

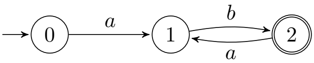
Automate pour $pre(L)$

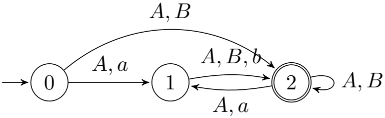
Proposition :

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 :

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

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





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

- $\subseteq$: récurrence sur nombre de transitions ajoutées
- $\supseteq$: 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 $\to$ 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:

- $\mathcal{Q} := Q_1 \times \cdots \times Q_k, \quad \iota := \langle \iota_1, \ldots, \iota_k \rangle$
- $\mathcal{F}_i := \{ \langle q_1, \ldots, q_k \rangle \in \mathcal{Q} \mid q_i \in F_i \}$
- $\delta : \mathcal{Q} \times Z \to \mathcal{Q}$ 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 nœud 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 $G_2 := S \to aSbS \mid \varepsilon$.

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*.

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) $S \to \varepsilon$.

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):

- 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 $C' \to BC$).
- Pour toute variable B telle que $B \to_G^* \varepsilon$ et toute production $A \to BC$ ou $A \to CB$, ajouter $A \to C$.
- Supprimer toute production $A \to \varepsilon$, mais ajouter $S_0 \to \varepsilon$ si $\varepsilon \in \mathcal{L}(G)$.
- 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 $|uv| > 0$ et $|u\beta v| < N$ 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 $N := 2^{k+1}$. Un arbre de dérivation de n'importe quel mot $w \in L$ avec $|w| \geq N$ contient un chemin de longueur au moins $N+1$, du coup une variable A se répète sur le chemin. Du coup il existe $A \to_G^* \beta$ et $A \to_G^* uAv$.

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.

Langages déterministes

Définition : Automate à pile déterministe

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

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

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

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

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. Même avec un automate déterministe, un mot peut avoir plusieurs calculs (ε-transitions à la fin), certains réussis et d'autres non.

Blocage

Définition : Blocage

Un automate à pile $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ 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. $px \not\xrightarrow[\omega]{\varepsilon}$.

De plus, ce critère est décidable.

Remarque :

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

Suppression des blocages

Proposition : Suppression des blocages

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

Preuve

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

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

Cette construction est effective.

Complémentaire

Proposition :

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

Proposition :

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0, F)$ un automate à pile déterministe, on peut effectivement construire un automate à pile déterministe équivalent $\mathcal{A}'$ tel qu'on ne puisse pas faire d'ε-transition à partir d'un état final de $\mathcal{A}'$.

Complémentaire (construction)

Preuve : Complémentaire

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

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

L'automate $\mathcal{A}'$ est déterministe et sans blocage.
Soit $w \in \Sigma^*$ et $q_0 z_0 \xrightarrow{w} p\alpha$ l'unique calcul *maximal* de $\mathcal{A}$ sur w. On a $p\alpha \stackrel{\varepsilon}{\not\to}$.
L'unique calcul *maximal* de $\mathcal{A}'$ sur w s'écrit $q'_0 z_0 \xrightarrow{w} (p,j)\alpha$ avec $j = 3$ si le calcul de $\mathcal{A}$ n'a pas visité F depuis la dernière transition visible ($\neq \varepsilon$), et $j = 4$ sinon.

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

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

Langages déterministes: Exercices

Exercice :

Montrer que tout langage déterministe est non ambigu.

Exercice :

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

Lemme d'itération pour les déterministes

Lemme : Itération

Soit $L \subseteq \Sigma^*$ un langage déterministe. Il existe un entier $N \in \mathbb{N}$ tel que tout mot $w \in L$ contenant au moins N lettres distinguées se factorise en $w = \alpha u \beta v \gamma$ avec

1. $\forall p \geq 0 : w = \alpha u^p \beta v^p \gamma \in \mathcal{L}(\mathcal{A})$,
2. $u\beta v$ contient moins de N lettres distinguées,
3. soit α, u, β soit β, v, γ contiennent des lettres distiguées,
4. pour tout $\gamma' \in \Sigma^*$,

$$\exists p : \alpha u^p \beta v^p \gamma' \in L \quad \Longrightarrow \quad \forall p : \alpha u^p \beta v^p \gamma' \in L$$

Preuve Admis.

Voir p.ex. Autebert, *Théorie des langages et des automates*, 1994

Remarque : Deterministes + Temps réel

Les langages D+TR sont un sous-ensemble strict des langages déterministes. Un lemme de d'itération pour les D+TR est présenté par Igarashi (1985).

Rappels: 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.