Langages formels

Paul Gastin

Paul.Gastin@lsv.fr

http://www.lsv.fr/~gastin/Langages/

L3 Informatique Cachan
2019-2020

Plan

Motivations

Définition :

1. Description et analyse (lexicale et syntaxique) des langages (programmation, naturels, ...)
2. Modèles de calcul
3. Abstractions mathématiques simples de phénomènes complexes dans le but de
   - Prouver des propriétés.
   - Concevoir des algorithmes permettant de tester des propriétés ou de résoudre des problèmes.
4. Types de données

Références

[7] Olivier Carton.
Langages formels, calculabilité et complexité.
Vuibert, 2008.

[9] John E. Hopcroft et Jeffrey D. Ullman.
Introduction to automata theory, languages and computation.
Addison-Wesley, 1979.

[10] Dexter C. Kozen.
Automata and Computability.
Springer, 1997.

[13] Jacques Sakarovitch.
Éléments de théorie des automates.
Vuibert informatique, 2003.

[8] Hubert Comon, Max Dauchet, Remi Gilleron, Florent Jacquemard, Denis Lugiez, Sophie Tison, Marc Tommasi.
Tree Automata Techniques and Applications.
http://www.grappa.univ-lille3.fr/tata/

Références

[1] Alfred V. Aho, Ravi Sethi et Jeffrey D. Ullman.
Compilers: principles, techniques and tools.
Addison-Wesley, 1986.

[2] Alfred V. Aho et Jeffrey D. Ullman.
The theory of parsing, translation, and compiling. Volume I: Parsing.
Prentice-Hall, 1972.

[3] Luc Albert, Paul Gastin, Bruno Petazzoni, Antoine Petit, Nicolas Puech et Pascal Weil.
Cours et exercices d'informatique.
Vuibert, 1998.

[4] Jean-Michel Autebert.
Théorie des langages et des automates.
Masson, 1994.

Références

[5] Jean-Michel Autebert, Jean Berstel et Luc Boasson.
Context-Free Languages and Pushdown Automata.
Handbook of Formal Languages, Vol. 1, Springer, 1997.

[6] Jean Berstel.
Transduction and context free languages.
Teubner, 1979.

[11] Jean-Éric Pin.
Automates finis et applications.
Polycopié du cours à l'École Polytechnique, 2004.

[12] Grzegorz Rozenberg et Arto Salomaa, éditeurs.
Handbook of Formal Languages,
Vol. 1, Word, Language, Grammar,
Springer, 1997.

[14] Jacques Stern.
Fondements mathématiques de l'informatique.
Mc Graw Hill, 1990.

Plan

Plan

Plan

Plan

Automates à pile

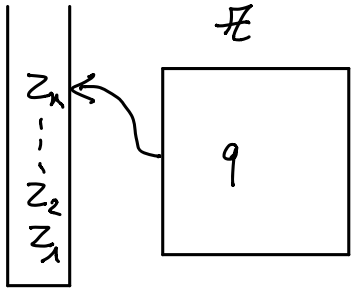
Définition : $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0, F)$ où

- 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.

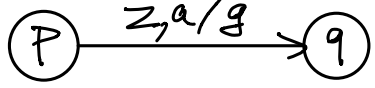
Définition : Système de transitions (infini) associé

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



Configuration $qz_n \cdots z_2 z_1$: automate $\mathcal{A}$ dans l'état q sommet de pile (tête de lecture) sur z_n

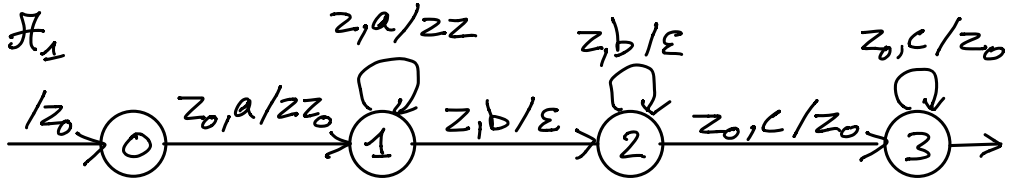
---

Une transition $(pz, a, qg) \in T$ se représente graphiquement par 

configuration initiale par 

Ex: $L_1 = \{ a^n b^n c^p \mid n, p > 0 \}$

$\mathcal{A}_1$



exemple de calcul acceptant pour $a^3 b^3 c^2$

$0 z_0 \xrightarrow{a} 1 z z_0 \xrightarrow{a} 1 z z z_0 \xrightarrow{a} 1 z^3 z_0 \xrightarrow{b} 2 z^2 z_0 \xrightarrow{b} 2 z z_0$
$\xrightarrow{b} 2 z_0 \xrightarrow{c} 3 z_0 \xrightarrow{c} 3 z_0$

Rem $\mathcal{A}_1$ est déterministe (D)
$\mathcal{A}_1$ est temps réel (TR)
$\mathcal{A}_1$ est "à fond de pile testable", i.e.,
la pile ne contient plus qu'un symbole ssi le sommet de pile est z_0

Automates à pile

Exemples :

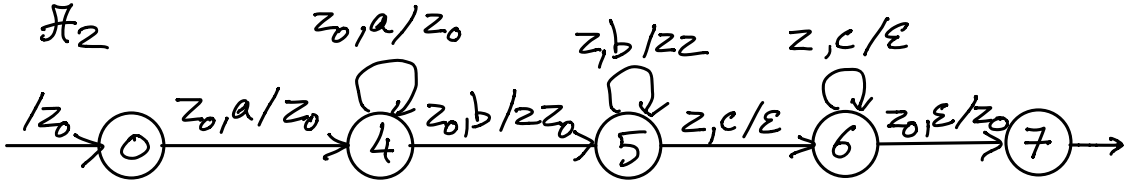
- $L_1 = \{a^n b^n c^p \mid n, p > 0\}$ et $L_2 = \{a^n b^p c^p \mid n, p > 0\}$
- $L = L_1 \cup L_2$ (non déterministe)

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 $T' \subseteq Q'Z \times (\Sigma \cup \{\varepsilon\}) \times Q'Z^{\leq 2}$.
4. Soit $\mathcal{A}$ un automate à pile. Montrer qu'on peut construire un automate à pile équivalent $\mathcal{A}'$ tel que les mouvements de la pile sont uniquement du type *push* ou *pop*.

Ex: $L_2 = \{ a^n b^p c^p \mid n,p > 0 \}$

$\mathcal{A}_2$



exemple de calcul acceptant pour $a b^2 c^2$

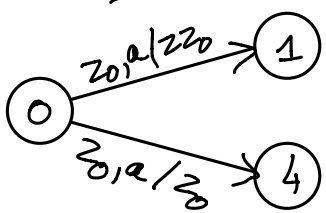
$0 z_0 \xrightarrow{a} 4 z_0 \xrightarrow{b} 5 z z_0 \xrightarrow{b} 5 z^2 z_0 \xrightarrow{c} 6 z z_0 \xrightarrow{c} 6 z_0 \xrightarrow{\varepsilon} 7 z_0$

L'automate $\mathcal{A}_2$ est D mais pas TR.

Ex: $L_3 = L_1 \cup L_2$

Il suffit de faire $\mathcal{A}_3 = \mathcal{A}_1 \cup \mathcal{A}_2$ en fusionnant l'état initial 0.

Mais $\mathcal{A}_3$ est non déterministe (ND) à cause de



Rem: pas d'automate D pour L_3 : $D \subsetneq ND$

Exercice: Construire un automate à pile déterministe pour

$L_4 = \{ a^n b^p c^n \mid n,p > 0 \} \cup \{ a^n b^p d^p \mid n,p > 0 \}$

Rem: Pas d'automate D+TR pour L_4

$D+TR \subsetneq D$

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 $0 \leq i < n$ alors $p_0 g_0 \xrightarrow{a_1} p_1 g_1 \cdots \xrightarrow{a_k} p_k g_k$ est un calcul de $\mathcal{A}$.

$\Longleftarrow$: 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 :

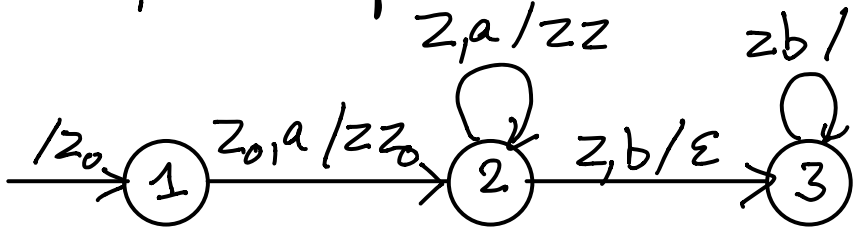
$L = \{a^n b^n \mid n \geq 0\}$ peut être accepté par pile vide ou par sommet de pile.

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.

Exemple : $L_5 = \{ a^n b^n \mid n \geq 0 \}$

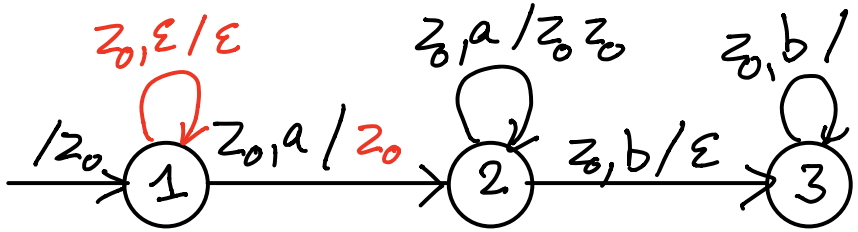
- acceptation par sommet de pile



$Z' = \{ z_0 \}$

déterministe

- Acceptation par pile vide



non déterministe

non ambigu

Acceptation généralisée

Preuve : Acceptation généralisée

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile et $K \subseteq QZ^*$ un langage reconnu par l'automate fini déterministe $\mathcal{B} = (P, Z \cup Q, \delta, p_0, F)$ avec $P \cap Q = \emptyset$.
Intuition: l'automate $\mathcal{A}'$ commence par simuler $\mathcal{A}$, puis (choix ND) lit la configuration atteinte $qh \in QZ^*$ dans l'automate $\mathcal{B}$ pour voir si elle est acceptante.
Soit $\mathcal{A}' = (Q', \Sigma, Z \uplus \{\bot\}, T', q_0' \bot, \{f\})$ avec $Q' = Q \uplus P \uplus \{q_0', f\}$, et

1. $q_0' \cdot \bot \xrightarrow{\varepsilon} q_0 \cdot z_0 \bot \in T'$, Initialisation
2. $T \subseteq T'$, Simulation
3. $q \cdot z \xrightarrow{\varepsilon} \delta(p_0, q) \cdot z \in T'$, si $q \in Q$ et $z \in Z \uplus \{\bot\}$, Acceptation
4. $p \cdot z \xrightarrow{\varepsilon} \delta(p, z) \cdot \varepsilon \in T'$, si $p \in P$ et $z \in Z$, Acceptation
5. $p \cdot \bot \xrightarrow{\varepsilon} f \cdot \varepsilon \in T'$, si $p \in F$, Acceptation

Exercice: Montrer que $\mathcal{L}(\mathcal{A}, K) = \mathcal{L}(\mathcal{A}')$.

Remarque: $\mathcal{A}'$ reconnaît aussi $\mathcal{L}(\mathcal{A}, K)$ par pile vide.

Exercice: Modifier $\mathcal{A}'$ pour qu'il reconnaisse $\mathcal{L}(\mathcal{A}, K)$ par sommet de pile.

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 $\mathcal{L}(\mathcal{A})$.

De plus, si $\mathcal{A}$ est *temps-réel* alors G est en FNG.

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.

De plus, si G est en FNPG alors on peut construire un tel $\mathcal{A}$ *temps-réel*.

Si G est en FNGQ alors on peut construire un tel $\mathcal{A}$ *standardisé* ($T \subseteq Z \times \Sigma \times Z^{\leq 2}$).

Preuve Automate à Pile vers Grammaire

On construit $G=(\Sigma, V, P)$ avec

$V=\{ X_{p,z,q} \mid p,q \in Q \text{ et } z \in Z \}$

de telle sorte que l'on ait

Lemme: $\mathcal{L}_G(X_{p,z,q}) = \{ w \in \Sigma^* \mid pz \xrightarrow{w}_* q \text{ dans } \mathcal{A} \}$

Pour cela on définit les règles de G par

$P = \{ X_{p,z,q} \longrightarrow a X_{p_1,y_1,p_2} \cdots X_{p_n,y_n,q} \mid$

$pz \xrightarrow{a} p_1 y_1 \cdots y_n \in T$ (donc $a \in \Sigma \cup \{\varepsilon\}$, $n \geq 0$, $y_i \in Z$)

$p_1, p_2, \ldots, p_n \in Q \}$

Rem: si $n=0$ la règle est $X_{p,z,q} \to a$

Preuve du lemme:

C Récurrence sur longueur d'une dérivation.

$X_{p,z,q} \xrightarrow{m} w \in \Sigma^*$

- Si $m=1$ alors $w=a \in \Sigma \cup \{\varepsilon\}$ et $pz \xrightarrow{a} q \in T$: OK
- Si $m>1$ alors la dérivation s'écrit :

$X_{p,z,q} \longrightarrow a X_{p_1,y_1,p_2} \cdots X_{p_n,y_n,q} \xrightarrow{m-1} w$

avec $pz \xrightarrow{a} p_1 y_1 \cdots y_n \in T$

lemme fondamental : $X_{p_i,y_i,p_{i+1}} \xrightarrow{m_i} w_i$ dans G avec

$w = w_1 \cdots w_n$ et $m-1 = m_1 + \cdots + m_n$ $(p_{n+1}=q)$

HR : $p_i y_i \xrightarrow{w_i}_* p_{i+1}$ dans $\mathcal{A}$

On recolle en utilisant le LF pour $\mathcal{A}$

$pz \xrightarrow{a} p_1 y_1 \cdots y_n \xrightarrow{w_1}_* p_2 y_2 \cdots y_n \xrightarrow{w_2}_* p_3 y_3 \cdots y_n$

$\cdots p_n y_n \xrightarrow{w_n} p_{n+1} = q$

2 Récurrence sur longueur du calcul $px \xrightarrow{w}_m q$ dans $\mathcal{A}$

- $m=1$ $w=a \in \Sigma \cup \{\varepsilon\}$ et $px \xrightarrow{a} q \in T$
  donc $px \to aq \in P$: OK
- $m>1$ le calcul s'écrit
  $px \xrightarrow{a} p_1 y_1 \cdots y_n \xrightarrow[m-1]{v} q$ avec $n \geq 1$ et $w=av$

Donc $X_{pzq} \to a X_{p_1 y_1 p_2} \cdots X_{p_n y_n p_{n+1}} \in P$ et
LF des automates à Pile appliqué $n-1$ fois
$\exists p_2, \dots, p_n$ et des calculs $p_i y_i \xrightarrow[m_i]{v_i} p_{i+1}$ de $\mathcal{A}$
avec $m-1 = m_1 + \cdots + m_n$ et $v = v_1 \cdots v_n$ $(p_{n+1}=q)$
HR on a dans G les dérivations
$X_{p_i y_i p_{i+1}} \xrightarrow{*} v_i$
On recolle pour obtenir dans G

$$X_{pzq} \to a X_{p_1 y_1 p_2} \cdots X_{p_n y_n p_{n+1}}$$
$$\xrightarrow{*} a v_1 X_{p_2 y_2 p_3} \cdots X_{p_n y_n p_{n+1}}$$
$$\vdots$$
$$\xrightarrow{*} a v_1 \cdots v_{n-1} X_{p_n y_n p_{n+1}} \xrightarrow{*} a v_1 \cdots v_n = w \quad \square$$

Preuve Grammaire vers Automate à Pile

Définition : Automate LL ou expansion/vérification

Soit $G = (\Sigma, V, P, S)$ une grammaire réduite.
On construit l'automate à pile simple non déterministe qui accepte par pile vide :
$\mathcal{A} = (\Sigma, \Sigma \cup V, T, S)$ où les transitions de T sont des

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

Remarque : sommet de pile à gauche.

Lemme $\forall \alpha \in (\Sigma \cup V)^* \quad \forall w \in \Sigma^*$

$\alpha \xrightarrow[g]{*} w$ dans G ssi $\alpha \xrightarrow[*]{w} \mathcal{E}$ dans $\mathcal{A}$

$\Rightarrow$ Rec. sur n dans $\alpha \xrightarrow[g]{n} w$ dérivation

- $n=0$ $\alpha = w \in \Sigma^*$

  Vérifications : $\alpha \xrightarrow[*]{w} \mathcal{E}$ dans $\mathcal{A}$

- $n>0$ $\alpha = u x \delta$ avec $u \in \Sigma^*, x \in V, \delta \in (\Sigma \cup V)^*$

On applique la règle $x \to \gamma \in P$: $\alpha \xrightarrow[g]{1} u\gamma\delta \xrightarrow[g]{n-1} w$

LF $w = uv$ et $\gamma\delta \xrightarrow[g]{n-1} v$

HR $\gamma\delta \xrightarrow[*]{v} \mathcal{E}$ dans $\mathcal{A}$

donc $\alpha \xrightarrow[*]{u} \gamma\delta \xrightarrow[*]{v} \mathcal{E}$ dans $\mathcal{A}$

(vérifications)

$\Leftarrow$ Rec sur n dans $\alpha \xrightarrow[n]{w} \mathcal{E}$ calcul de $\mathcal{A}$

$n=0$: $w = \mathcal{E} = \alpha$ donc $\alpha \xrightarrow[g]{0} w$

$n>0$: 1) si $\alpha = a\delta$ avec $a \in \Sigma$ alors le calcul est $\alpha \xrightarrow[1]{a} \delta \xrightarrow[n-1]{v} \mathcal{E}$ avec $w = av$

HR $\delta \xrightarrow[g]{*} v$ donc $\alpha = a\delta \xrightarrow[g]{*} av = w$

2) $\alpha = x\delta$ avec $x \in V$ alors le calcul est

$\alpha \xrightarrow{\mathcal{E}} \gamma\delta \xrightarrow[n-1]{w} \mathcal{E}$

HR $\gamma\delta \xrightarrow[g]{*} w$ Donc $\alpha = x\delta \xrightarrow[g]{1} \gamma\delta \xrightarrow[g]{*} w$ □

Remarque : Si G est en FNPG, on remplace les expansions par $\{(x, a, \alpha) \mid x \to a\alpha \in P\}$

L'automate obtenu est TR.

Configurations accessibles (mots de pile)

Proposition : Reconnaissabilité des configurations accessibles

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile.
Pour $p \in Q$ et $g \in Z^*$, on note

$$\mathcal{C}(pg) = \{qh \in QZ^* \mid \exists\ pg \not\Rightarrow qh \text{ dans } \mathcal{T}\}$$

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

Preuve : Voir plus loin.

Corollaire : Décidabilité du vide

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0, F)$ un automate à pile.
On peut décider si $\mathcal{L}(\mathcal{A}) = \emptyset$.

Preuve

Il suffit de tester si $q_0 \in F$ ou $\mathcal{C}(q_0 z_0) \cap FZ^* \neq \emptyset$.

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

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.

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 $a \in \Gamma$.

Remarque : Soit $D = \{w \in \widetilde{\Gamma}^* \mid w \xrightarrow[*]{\text{red}} \varepsilon\}$.

On peut montrer que D est engendré par la grammaire $S \to \varepsilon + \sum_{a\in\Gamma} \overline{a}SaS$.
Il s'agit donc du langage de Dyck en considérant $\overline{a}$ comme une parenthèse ouvrante et a comme la parenthèse fermante correspondante.

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).

Clôture et réduction.

Preuve : Reconnaissabilité de la clôture

Soit $\mathcal{A} = (Q, \widetilde{\Gamma}, T, I, F)$ un automate fini reconnaissant L.

Pour $p, q \in Q$, on note $\mathcal{A}_{p,q} = (Q, \widetilde{\Gamma}, T, p, q)$ et $L_{p,q} = \mathcal{L}(\mathcal{A}_{p,q})$.

On peut décider si $L_{p,q} \cap D \neq \emptyset$ en PTIME

On peut calculer (PTIME) l'ensemble $R = \{(p, q) \in Q^2 \mid L_{p,q} \cap D \neq \emptyset\}$.

On définit $\mathcal{A}' = (Q, \widetilde{\Gamma}, T', I, F)$ en ajoutant à $\mathcal{A}$ des ε-transitions:

$$T' = T \cup \{(p, \varepsilon, q) \mid (p, q) \in R\}.$$

Remarque: $R^2 = R$ et donc dans un calcul de $\mathcal{A}'$ il est inutile de prendre deux (ou plus) ε-transitions consécutives.

Lemme: $\mathcal{L}(\mathcal{A}') = \mathrm{Clot}(L)$.

$\subseteq$: Soit $w = a_1 \cdots a_n \in \mathcal{L}(\mathcal{A}')$. On a un calcul acceptant dans $\mathcal{A}'$

$$p_0 \xrightarrow{\varepsilon} q_0 \xrightarrow{a_1} p_1 \xrightarrow{\varepsilon} q_1 \cdots \xrightarrow{a_n} p_n \xrightarrow{\varepsilon} q_n$$

pour tout $0 \leq i \leq n$, il existe un mot $v_i \in L_{p_i,q_i} \cap D$. On obtient

$$v = v_0 a_1 v_1 \cdots a_n v_n \xrightarrow[*]{\mathrm{red}} w$$

Clôture et réduction.

Preuve : Reconnaissabilité de la clôture

$\supseteq$: Par induction sur le nombre n de réductions: Soit $v \xrightarrow[n]{\text{red}} w$ avec $v \in L$.

Si $n = 0$ alors $w = v \in L = \mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$.

Si $n > 0$ alors $v \xrightarrow[n-1]{\text{red}} w' \xrightarrow[1]{\text{red}} w$.

On peut écrire $w' = w_1 \overline{a} a w_2$ et $w = w_1 w_2$ avec $a \in \Gamma$.

Par induction, on a un calcul acceptant pour w' dans $\mathcal{A}'$. Ce calcul s'écrit

$$q_0 \xrightarrow{w_1} p \xrightarrow{\overline{a}} p' \xrightarrow{\varepsilon} q' \xrightarrow{a} q \xrightarrow{w_2} q_f .$$

$(p', q') \in R$ donc il existe $u \in L_{p',q'} \cap D$.

On en déduit $\overline{a} u a \in L_{p,q} \cap D$ et donc $(p, q) \in R$.

On obtient le calcul acceptant pour w dans $\mathcal{A}'$:

$$q_0 \xrightarrow{w_1} p \xrightarrow{\varepsilon} q \xrightarrow{w_2} q_f .$$

Configurations accessibles (Preuve)

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile.

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

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

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

Preuve du Lemme

$\Longrightarrow$: par récurrence sur n. Soit $pg \xrightarrow[n]{} qh$ un calcul de $\mathcal{A}$.

Si $n = 0$ alors $qh = pg$ et on prend $w = \varepsilon \in K^0$.

Si $n > 0$ alors $g = xg'$ et le calcul s'écrit

$$pg = pxg' \xrightarrow[1]{a} p'h'g' \xrightarrow[n-1]{} qh \text{ avec } (px, a, p'h') \in T.$$

Par induction, il existe $w' \in K^{n-1}$ tel que $w'p'h'g' \xrightarrow[2(n-1)]{\text{red}} qh$.

Soit $w = w'(p'h'\overline{x}\,\overline{p}) \in K^n$. On a $wpg \xrightarrow[2n]{\text{red}} qh$.

Configurations accessibles (Preuve)

Preuve du Lemme

$\Longleftarrow$: par récurrence sur n. Soit $w \in K^n$ tel que $wpg \xrightarrow[2n]{\text{red}} qh$.

Si $n = 0$ alors $w = \varepsilon$ et $qh = pg$: on a $pg \xrightarrow[0]{} qh$ dans $\mathcal{A}$.

Si $n > 0$ alors $w = w'v$ avec $w' \in K^{n-1}$ et $v = p'h'\overline{x}\,\overline{r} \in K$.

Toutes les lettres barrées doivent être réduites à partir de $wpg = w'p'h'\overline{x}\,\overline{r}pg$.

On en déduit $r = p$ et $g = xg'$.

Donc, on a une transition $(px, a, p'h') \in T$ et une réduction $w'p'h'g' \xrightarrow[2(n-1)]{\text{red}} qh$.

Par induction, il existe un calcul $p'h'g' \xrightarrow[n-1]{} qh$ dans $\mathcal{A}$.

Finalement,

$$pg = pxg' \xrightarrow[1]{a} p'h'g' \xrightarrow[n-1]{} qh \text{ dans } \mathcal{A}.$$

Calculs d'accessibilité

Exercice : Calculs infinis

Soit $\mathcal{A} = (Q, \Sigma, Z, T, q_0 z_0)$ un automate à pile.
Montrer qu'on peut effectivement calculer les ensembles suivants :

1. $V = \{(p, x) \in Q \times Z \mid \exists\ px \xrightarrow[\omega]{}$ dans $\mathcal{T}\}$
2. $V' = \{(p, x) \in Q \times Z \mid \exists\ px \xrightarrow[\omega]{\varepsilon}$ dans $\mathcal{T}\}$

Preuve : Indications

On calcule l'ensemble V comme plus grand point fixe. Pour cela, on définit:

$$
\begin{aligned}
V_0 &= Q \times Z \\
V_{m+1} = \{(p, x) \in Q \times Z \mid\ & \exists (px, a, p_1 y_1 \cdots y_n) \in T, \\
& \exists 1 \leq k \leq n,\ \exists p_2, \ldots, p_k \in Q \text{ tels que} \\
& (p_k, y_k) \in V_m \text{ et } \forall 1 \leq i < k,\ (p_i, y_i, p_{i+1}) \in X\}
\end{aligned}
$$

Montrer alors que $(V_m)_{m \geq 0}$ est une suite décroissante et que

$$V = \bigcap_{m \geq 0} V_m$$

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 :

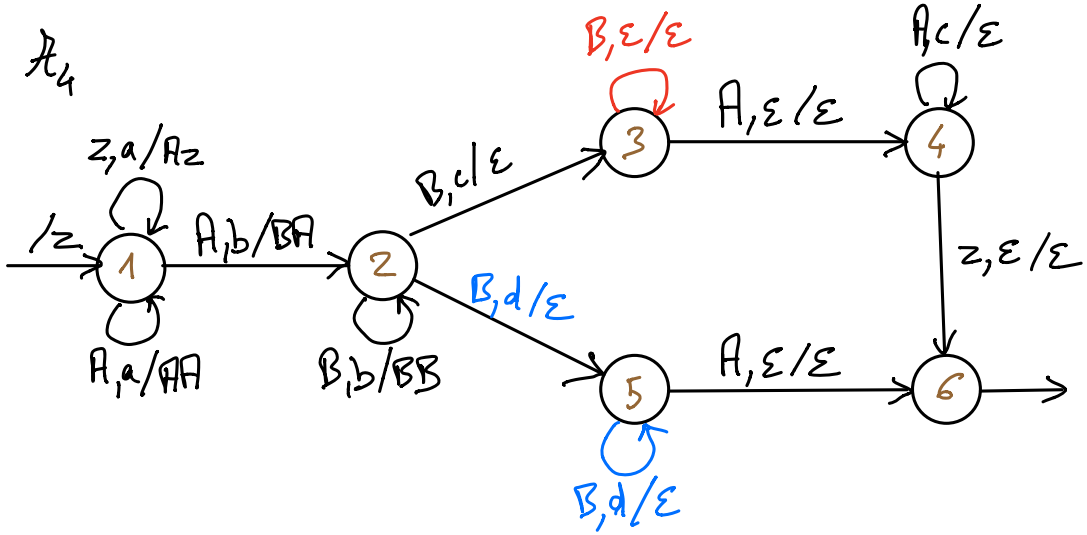
1. Le langage $L_4 = \{a^n b^p c^n \mid n, p > 0\} \cup \{a^n b^p d^p \mid n, p > 0\}$ est déterministe mais pas D+TR.
2. $L_5 = \{a^n b a^n \mid n > 0\}$ peut être accepté par un automate D+TR mais pas par un automate D+S car il n'est pas fermé par préfixe.

Exercices :

1. Montrer que D_n^* est D+TR mais pas D+S.
2. Montrer que le langage $\{a^n b^n \mid n > 0\} \cup \{a^n b^{2n} \mid n > 0\}$ est non ambigu mais pas déterministe.

Ex: $L_4 = \{ a^n b^p c^n \mid n,p>0 \} \cup \{ a^n b^p d^p \mid n,p>0 \}$

$\mathcal{A}_4$



Exemple de calcul

$1z \xrightarrow{a} 1Az \xrightarrow{a} 1AAz \xrightarrow{b} 2BAAz \xrightarrow{b} 2BBAAz \xrightarrow{c} 3BAAz$
$\xrightarrow{\varepsilon} 3AAz \xrightarrow{\varepsilon} 4Az \xrightarrow{c} 4z \xrightarrow{\varepsilon} 6$

$1z \xrightarrow{a} 1Az \xrightarrow{a} 1AAz \xrightarrow{b} 2BAAz \xrightarrow{b} 2BBAAz \xrightarrow{d} 5BAAz$
$\xrightarrow{d} 5AAz \xrightarrow{\varepsilon} 6Az$

L'automate $\mathcal{A}_4$ est déterministe mais pas temps-réel
On pourrait enlever les ε-transitions
$(3A, \varepsilon, 4)$, $(4z, \varepsilon, 6)$ et $(5A, \varepsilon, 6)$
mais pas $(3B, \varepsilon, 3)$
Il n'y a pas d'automate D+TR pour L_4
Donc D+TR $\subsetneq$ D

Acceptation par pile vide

Exemples :

1. Le langage $L_5 = \{a^n b a^n \mid n \geq 0\}$ peut être accepté par *pile vide* par un automate D+TR+S.
2. Le langage $L_4 = \{a^n b^p c^n \mid n, p > 0\} \cup \{a^n b^p d^p \mid n, p > 0\}$ peut être accepté par *pile vide* par un automate D.

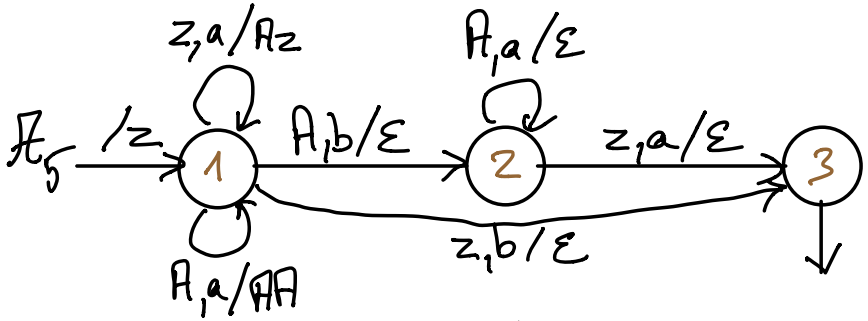
Exercices :

1. Montrer qu'un langage L est déterministe et préfixe ($L \cap L\Sigma^+ = \emptyset$) ssi il existe un automate déterministe qui accepte L par pile vide.
2. Montrer que pour les automates à pile déterministes, l'acceptation par pile vide est équivalente à l'acceptation par pile vide ET état final.

Exercice :

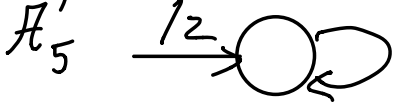
Montrer que D_n^* peut être accepté par sommet de pile par un automate D+TR+S.

Ex: $L_5 = \{ a^n b a^n \mid n \geq 0 \}$



$\mathcal{A}_5$ est D+TR

Mais pas d'automate simple avec acceptation par état final car L_5 n'est pas fermé par préfixe.



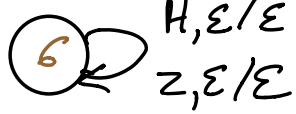
z,a/zA
z,b/ε
A,a/ε

$\mathcal{A}'_5$ est D+TR+S et accepte L_5 par pile vide.

Exemple de calcul sur $a^2 b a^2$

$z \xrightarrow{a} zA \xrightarrow{a} zAA \xrightarrow{a} zAAA \xrightarrow{b} AAA \xrightarrow{a} AA \xrightarrow{a} A \xrightarrow{a} \varepsilon$

Ex L_4 est reconnu par pile vide par l'automate $\mathcal{A}'_4$ obtenu en ajoutant à $\mathcal{A}_4$ les transitions $(6A, \varepsilon, 6)$ et $(6z, \varepsilon, 6)$



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 $\alpha \neq \varepsilon$).

Blocage

Preuve : Critère d'absence de blocage

Si $\mathcal{A}$ est sans blocage alors il satisfait clairement le critère.

Réciproquement, supposons que $\mathcal{A}$ satisfait le critère.

On calcule $X' = \{(r, x, s) \in Q \times Z \times Q \mid \exists\, rx \xrightarrow[+]{\varepsilon} s \text{ dans } \mathcal{T}\}$.

Soit $p\alpha$ une configuration accessible et $a \in \Sigma$.

On écrit $\alpha = x_1 x_2 \cdots x_n$ avec $x_i \in Z$.

Il n'existe pas de calcul $p\alpha \xrightarrow[+]{\varepsilon} p'$ car sinon la configuration p' est accessible ce qui contredit le point 1 du critère.

Donc il existe $1 \leq k \leq n$ et des états $p_2, \ldots, p_k$ tels que

$$(p, x_1, p_2), (p_2, x_3, p_3), \ldots, (p_{k-1}, x_{k-1}, p_k) \in X'$$

et il n'existe pas p_{k+1} avec $(p_k, x_k, p_{k+1}) \in X'$.

La configuration $p_k x_k \cdots x_n$ est accessible, donc $p_k x_k \not\xrightarrow[\omega]{\varepsilon}$ (critère point 3).

On en déduit $p_k x_k \xrightarrow[*]{\varepsilon} qy\beta$ avec $qy \not\xrightarrow{\varepsilon}$ et donc $qy \xrightarrow{a}$ (critère point 2).

Finalement, $$p\alpha \xrightarrow[*]{\varepsilon} p_k x_k \cdots x_n \xrightarrow[*]{\varepsilon} qy\beta x_{k+1} \cdots x_n \xrightarrow{a}$$

Décidabilité du critère: on vérifie que $\mathcal{C}(q_0 z_0) \cap Q = \emptyset$ (point 1) et pour chaque px tel que $px = q_0 z_0$ ou $\mathcal{C}(q_0 z_0) \cap pxZ^* \neq \emptyset$, on vérifie le point 2 du critère et que $(p, x) \notin V'$ (point 3).

Blocage

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

$Q' = Q \uplus \{q_0', d, f\}$, $F' = F \uplus \{f\}$, $Z' = Z \uplus \{\bot\}$, $z_0' = \bot$ 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 $px \overset{q}{\not\to}$ et $px \overset{\varepsilon}{\not\to}$ dans $\mathcal{A}$ alors $px \xrightarrow{a} dx \in T'$,
4. Si $px \overset{\varepsilon}{\underset{\omega}{\not\to}}$ 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. $p\bot \xrightarrow{\varepsilon} d\bot$, $d\bot \xrightarrow{a} d\bot$, $dx \xrightarrow{a} dx$ et $fx \xrightarrow{a} dx$.

Cette construction est effective.

Blocage

Preuve : $\mathcal{A}'$ est déterministe, sans blocage, constructible en PTIME

Déterministe: il suffit de le vérifier.

Sans blocage: on vérifie les conditions du critère.

1. $\mathcal{C}'(q'_0\perp) \cap Q = \emptyset$: on ne peut pas vider entièrement la pile car le fond de pile $\perp$ n'est jamais effacé.

2. Soit $px\alpha$ une configuration accessible. Supposons $px \overset{\varepsilon}{\not\to}$ dans $\mathcal{A}'$.

Si $p \in Q$ alors $x \neq \perp$ (7) et donc (2,3) $\forall\, a \in \Sigma$, $px \xrightarrow{a}$

On a $p \neq q'_0$ et si $p \in \{d, f\}$ alors (7) $\forall\, a \in \Sigma$, $px \xrightarrow{a}$ (notons que $px \neq f\perp$).

3. Soit $px\alpha$ une configuration accessible. Supposons $px \xrightarrow[\omega]{\varepsilon}$ dans $\mathcal{A}'$.

Ce calcul ne peut pas utiliser les états d et f: il n'utilise que les transitions (1,4).

La transition (1) est utilisée au plus une fois.

Si $px \neq q'_0\perp$ alors $px \xrightarrow[\omega]{\varepsilon}$ dans $\mathcal{A}$ et on ne peut pas utiliser la transition (4).

Si $px = q'_0\perp$ alors $q_0 z_0 \xrightarrow[\omega]{\varepsilon}$ dans $\mathcal{A}$ et la transition (4) est impossible aussi.

Constructible en PTIME:

Pour les conditions des transitions (4,5,6) on utilise les ensembles V', X' et W'.

Blocage

Preuve : $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$

Soit $q_0 z_0 \xrightarrow{w} q\beta$ un calcul acceptant de $\mathcal{A}$ pour $w \in \Sigma^*$.

Si le calcul ne passe pas par une configuration $px\alpha$ avec $px \xrightarrow[\omega]{\varepsilon} ((p,x) \in V')$ alors

$$q'_0\bot \xrightarrow[(1)]{\varepsilon} q_0 z_0 \bot \xrightarrow[(2,4)]{w} q\beta\bot$$

est un calcul acceptant de $\mathcal{A}'$.

Sinon, le calcul s'écrit $q_0 z_0 \xrightarrow{w} px\gamma \xrightarrow[*]{\varepsilon} q\beta$ avec $px \xrightarrow[\omega]{\varepsilon}$

Donc $\beta = \alpha\gamma$ avec $px \xrightarrow[*]{\varepsilon} q\alpha$ et $q \in F$. On obtient un calcul acceptant dans $\mathcal{A}'$:

$$q'_0\bot \xrightarrow[(1)]{\varepsilon} q_0 z_0 \bot \xrightarrow[(2,4)]{w} px\gamma\bot \xrightarrow[(5)]{\varepsilon} fx\gamma\bot$$

Preuve : $\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})$

Un calcul acceptant de $\mathcal{A}'$ ne peut pas atteindre l'état puits d. Deux possibilités:

Si le calcul accepte grâce à l'état f, il s'écrit:

$$q'_0\bot \xrightarrow[(1)]{\varepsilon} q_0 z_0 \bot \xrightarrow[(2,4)]{w} px\gamma\bot \xrightarrow[(5)]{\varepsilon} fx\gamma\bot$$

On en déduit $q_0 z_0 \xrightarrow{w} px\gamma$ dans $\mathcal{A}$ et il existe $px \xrightarrow[*]{\varepsilon} q\alpha$ dans $\mathcal{A}$ avec $q \in F$.

Donc w est accepté par le calcul $q_0 z_0 \xrightarrow{w} px\gamma \xrightarrow[*]{\varepsilon} q\alpha\gamma$ de $\mathcal{A}$.

Sinon, le calcul de $\mathcal{A}'$ s'écrit $q'_0\bot \xrightarrow[(1)]{\varepsilon} q_0 z_0 \bot \xrightarrow[(2,4)]{w} q\beta\bot$ avec $q \in F$

et on obtient le calcul acceptant $q_0 z_0 \xrightarrow{w} q\beta$ dans $\mathcal{A}$ pour w.

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 $\mathcal{A}'$.

Exercice :

Montrer que tout langage déterministe est non ambigu.

Complémentaire

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 $p\alpha \not\xrightarrow{\varepsilon}$.
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

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.

Exercice :

Soit $\mathcal{A}$ un automate à pile déterministe. Montrer qu'on peut effectivement construire un automate à pile déterministe qui reconnaît le même langage et dont les ε-transitions sont uniquement effaçantes : $px \xrightarrow{\varepsilon} q$.

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 [4]

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 :

- $R \subseteq L$? $\qquad R \subseteq L \iff R \cap \overline{L} = \emptyset$
- $L = R$? $\qquad R = L \iff R \cap \overline{L} = \emptyset = \overline{R} \cap L$
- L est-il rationnel ?
- $L = L'$? $\qquad$ 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 ?

Plan

Plan

Plan