Theoretical Computer Science 11 (1980) 221–225


# NOTE

# AUTOMATES A FILE

Bernard VAUQUELIN et Paul FRANCHI-ZANNETTACCI

*U.E.R. de Mathématiques et Informatique, Laboratoire Associé au C.N.R.S., Université de Bordeaux I, 33405 Talence, France*

Communiqué par M. Nivat
Reçu janvier 1979

## 1. Introduction

Les notions de pile (LIFO) et de file (FIFO) sont deux structures fondamentales dans l'écriture d'algorithme.

Or, en théorie des langages formels, si les automates à pile occupent une place fondamentale [6], il n'en est pas de même des automates obtenus intuitivement en remplaçant la pile par une file que l'on peut appeler automates à file.

Une raison de cette absence d'intérêt réside vraisemblablement dans le fait que les automates à file ont la puissance d'une machine de Turing.

Cependant, l'étude de ces automates, qui est l'objet de cette note, permet d'exhiber un générateur de la famille des langages récursivement énumérables. Nous appelons anti-Dyck le générateur (noté $Q_2'^*$) en raison de ses similitudes avec le langage de Dyck ($D_2'^*$) générant la famille des langages algébriques [3].

On montre, d'autre part, que la famille des langages reconnus par un automate à file à délai borné (Quasi-Real-Time) est un AFL [4] inclus dans la famille des langages dépendant du contexte (context-sensitive).

## 2. Définition des Automates à file

Un automate à file $A$ est défini par:
- un ensemble fini d'*états* $Q$,
- un *état initial* $q_0 \in Q$,
- un ensemble d'*états finals* $Q_1 \subset Q$,
- un alphabet d'*entrée* $X$,
- un alphabet de *file* $Y$,

- et un ensemble de transitions:

$$T \subset Q \times (X \cup \{\varepsilon\}) \times (Y \cup \bar{Y} \cup \{\varepsilon\}) \times Q$$

où $\bar{Y}$ est une copie de $Y$ telle que $Y \cap \bar{Y} = \emptyset$.

On définit la relation $\vdash_A$ dans l'ensemble $Q \times X^* \times Y^*$ des configurations de l'automate par:

$$\langle q, u, v\rangle \underset{A}{\vdash} \langle q, u', v'\rangle$$

si et seulement si:

(1) $u = xu'$ et $x \in X \cup \{\varepsilon\}$ et

(2) $(q, x, \varepsilon, q') \in T$ et $v' = v$,
$(q, x, y, q') \in T$ et $v' = vy$, $y \in Y$,
$(q, x, \bar{y}, q') \in T$ et $yv' = v$, $y \in Y$.

Le langage accepté par $A$ est alors:

$$L_A = \{u \in X^* \mid \langle q_0, u, \varepsilon\rangle \overset{*}{\underset{A}{\vdash}} \langle q_1, \varepsilon, \varepsilon\rangle \ \ q_1 \in Q_1\}$$

où $\overset{*}{\underset{A}{\vdash}}$ est la fermeture réflexive et transitive de $\vdash_A$.

**Example.** Les langages $\{f^2 \mid f \in X^*\}$ et $\{f^k \mid f \in X^*, k > 0\}$ sont acceptés par des automates à file.

## 3. L'anti-Dyck générateur de la famille

### *3.1. Définition et propriétés de l'anti-Dyck*

On note:

$$Q_n'^* = \{v \in (Y \cup \bar{Y})^* \mid v \overset{*}{\rightarrow} \varepsilon\}$$

où $n = |Y|$ et $\overset{*}{\rightarrow}$ est la fermeture transitive de $\rightarrow$ définie dans $(Y \cup \bar{Y})^*$ par:

$$\forall y \in Y \quad yv_1\bar{y}v_2 \rightarrow v_1v_2 \text{ ssi } v_1 \in Y^*.$$

En remarquant que comme $D_n'^*$ (Dyck restreint à $n$ paires de lettres) le langage $Q_n'^*$ permet de coder des arbres étiquetés, on a la propriété suivante:

**Propriété 1.** *Il existe une bijection $\varphi_n$ entre $D_n'^*$ et $Q_n'^*$ qui préserve le passage à l'image commutative.*

En notant $D_y'$ *le langage* $D_1'$ sur l'alphabet $\{y, \bar{y}\}$ on obtient

**Propriété 2.** *$D_n'^*$ et $Q_n'^*$ étant définis sur le même alphabet $\{y_1, y_2, \ldots, y_n\} \cup \{\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_n\}$ on a:* $D_n'^* \cap Q_n'^* = (D_{y_1}' \cup D_{y_2}' \cup \cdots \cup D_{y_n}')^*$.

### *3.2. L'anti-Dyck générateur des récursivement énumérables*

On remarque que dans la définition d'un automate à file, l'ensemble $T$ des transitions définit un transducteur fini, réalisant une transduction $\tau$ telle que:

$$L_A = \{u \in X^* \mid \tau(u) \cap Q_n'^* \neq \emptyset\} = \tau^{-1}(Q_n'^*).$$

On obtient donc le résultat suivant

**Lemme 1.** *La famille des langages acceptés par un automate à file forme un cône rationnel* [1] *engendré par* $Q_n'^*$.

D'autre part on a

**Lemme 2.** $Q_2'^*$ *est rationnellement équivalent à* $Q_n'^*$ *pour tout n.*

Il suffit d'appliquer le morphisme $\lambda_n$ défini de $\{y_1, y_2, \ldots, y_n, \bar{y}_1, \bar{y}_2, \ldots, \bar{y}_n\}^*$ dans $\{a, b, \bar{a}, \bar{b}\}^*$ par

$$\lambda_n(y_i) = a^i b, \qquad \lambda_n(\bar{y}_i) = \bar{a}^i \bar{b}.$$

En vertu des lemmes précédents et en utilisant le fait que l'on peut simuler une machine de Turing par un automate à file, on démontre:

**Théorème 1.** $Q_2'^*$ *est générateur du cône rationnel des langages récursivement énumérables.*

## 4. Les automates bornés en temps

### *4.1. L'AFL des langages acceptés par un automate à délai borné*

Un automate à file est à *délai borné* ('Quasi-Real-Time') s'il existe un entier $k$ tel que l'automate utilise au plus $k$ transitions consécutives par $\varepsilon$.

On note *QRTF* la famille des langages acceptés par un automate *à délai borné.*

Il est clair que QRTF est le cône fidèle engendré par $Q_2'^*$ et de plus:

**Théorème 2.** QRTF *est un* AFL.

Pour obtenir ce résultat on utilise le fait que $(Q_2'^* y_3 \bar{y}_3)^*$ est rationnellement équivalent à $Q_3'^*$ [5].

### *4.2. Langages acceptés par un automate à file polynomialement borné*

On définit de même les automates à file bornés en temps par une fonction polynomiale $P(l)$ de la longueur du mot d'entrée.

On note LTF la famille des langages acceptés dans le cas d'une fonction $P$ linéaire. On montre alors que:

$$\text{QRTF} \subseteq \text{LTF} \subseteq \text{CS} \quad \text{('context sensitive')}.$$

### 4.3. *Exemples de langages*

Les exemples suivants permettent de se faire une idée de la puissance des automates mentionnés plus haut:

- Les langages $\{f^k \mid f \in (a, b)^*, k > 0\}$ et $\{f^{|f|} \mid f \in (a, b)^*\}$ appartiennent à QRTF;
- Les langages *COPY, COPY*, SHIFT définis par Book et al. [2] appartiennent à QRTF;
- En utilisant le fait que les langages $\{a^n b a^{n^2} \mid n > 0\}$ et $\{a^{n^2} b a^n \mid n > 0\}$ sont dans QRTF, on montre que;
- Le langage $L_1 : \{a^n g^2 b^n \mid g \in (c, d)^*, n > 0\}$ appartient aussi à QRTF;
- Le langage $L_2 : \{f g^2 f \mid f \in (a, b)^*, g \in (c, d)^*\} \in \text{LTF}$;
- $D_2'^*$ et donc tout langage algébrique est accepté par un automate à file borné par une fonction $P(l)$ de degré 2.

## 5. Problèmes ouverts

Une question importante reste sans réponse, celle de savoir si les inclusions de Section 4.2 sont strictes ou non.

**Conjectures.** (1) Le langage $L_2 \notin$ QRTF;
(2) Le langage $\{f\tilde{f} \mid f \in (a, b)^*\} \notin$ RTF.

**Question.** Quelles sont les familles acceptées par les différentes classes d'automates définies plus haut quand ils sont déterministes?

En particulier, nous conjecturons que $L_1$ n'est pas accepté par un automate à délai borné déterministe.

## Bibliographie

[1] L. Boasson and M. Nivat, Sur diverses familles de langages fermées par transduction rationnelle, *Acta informat.* **2** (1973) 180–188.
[2] R. Book, S. Greibach and C. Wrathall, Comparisons and reset machines in automata, languages and Programming Lectures Notes in Computer Science **62** (Springer, Berlin, 1978) 113–124.
[3] N. Chomsky and M.P. Schutzenberger, The algebraic theory of context-free languages, in: *Computer Programming and Formal Systems* (North-Holland, Amsterdam, 1963) 118–161.

[4] S. Ginsburg and S. Greibach, Abstract families of languages, in: *Studies in Abstract Families of Languages, Memoir* **87** (Amer. Math. Soc., Providence, RI, 1969) 1–32.
[5] S. Ginsburg and S. Greibach, Principal AFL, *J. Comput. System Sci.* **4** (1970) 308–338.
[6] M.P. Schutzenberger, On context-free languages and pushdownautomata, *Information and Control* **6**(3) (1963) 246–264.