A NOTE ON SYNCHRONIZED EXTENSION SYSTEMS

Ferucio Laurenţiu Ţiplea and Erkki Mäkinen

DEPARTMENT OF COMPUTER AND INFORMATION SCIENCES
UNIVERSITY OF TAMPERE

REPORT A-2000-11

UNIVERSITY OF TAMPERE
DEPARTMENT OF COMPUTER AND
INFORMATION SCIENCES
SERIES OF PUBLICATIONS A
A-2000-11, JULY 2000

A NOTE ON SYNCHRONIZED EXTENSION SYSTEMS

Ferucio Laurenţiu Ţiplea and Erkki Mäkinen

University of Tampere
Department of Computer and Information Sciences
P.O.Box 607
FIN-33014 University of Tampere, Finland

ISBN 951-44-4882-0
ISSN 1457-2060

A Note on Synchronized Extension Systems

Ferucio Laurenţiu Ţiplea [1]

Faculty of Computer Science, "Al.I.Cuza" University of Iasi, 6600 Iasi, Romania

Erkki Mäkinen [2]

Department of Computer and Information Sciences, P.O. Box 607, FIN-33014 University of Tampere, Finland

Abstract

The concept of a *synchronized extension system* (SE-system, for short) has been introduced in [2] as a 4-tuple $G = (V, L_1, L_2, S)$, where V is an alphabet and L_1, L_2 and S are languages over V. Such systems generate languages extending L_1 by L_2 to the left or to the right, and synchronizing on words in S. In [2] it has been shown that the language of type r^- generated by an SE-system of type (r, r, f) is regular. As a particular case, the stack language of a pushdown automaton is regular.

In this note we prove the converse. That is, using the fact that the stack language of a pushdown automaton is regular, we obtain that the language of type r^- generated by an SE-system of type (r, r, f) is regular.

Keywords: formal languages, pushdown automata, stack languages.

1 Preliminaries

Synchronized extension systems (SE-systems, for short) have been introduced in [2] as 4-tuples $G = (V, L_1, L_2, S)$, where V is an alphabet and L_1, L_2 and S are languages over V. L_1 is called the *initial language*, L_2 the *extending language*, and S the *synchronization set* of G. For an SE-system G, define the binary relations $\Rightarrow_{G,r}$, $\Rightarrow_{G,r^-}$, $\Rightarrow_{G,l}$ and $\Rightarrow_{G,l^-}$ over V^* as follows:

(i) $u \Rightarrow_{G,r} v$ iff $(\exists w \in L_2)(\exists s \in S)(\exists x, y \in V^*)(u = xs \wedge w = sy \wedge v = xsy)$;
(ii) $u \Rightarrow_{G,r^-} v$ iff $(\exists w \in L_2)(\exists s \in S)(\exists x, y \in V^*)(u = xs \wedge w = sy \wedge v = xy)$;
(iii) $u \Rightarrow_{G,l} v$ iff $(\exists w \in L_2)(\exists s \in S)(\exists x, y \in V^*)(u = sx \wedge w = ys \wedge v = ysx)$;

[1] E-mail:fltiplea@infoiasi.ro
[2] E-mail: em@cs.uta.fi. Work supported by the Academy of Finland (Project 35025).

(iv) $u \Rightarrow_{G,l^-} v$ iff $(\exists w \in L_2)(\exists s \in S)(\exists x, y \in V^*)(u = sx \wedge w = ys \wedge v = yx)$.

In an SE-system $G = (V, L_1, L_2, S)$, the words in S act as synchronization words; they can be kept or neglected in the final result. r, r^-, l and l^- are called (basic) *modes of synchronizations.* They can be used to define another four new modes of synchronization, (l, r), (l^-, r), (l, r^-) and (l^-, r^-), by a disjunctive combination. For example, the relation $\Rightarrow_{G,(l^-,r)}$ is defined by

$$u \Rightarrow_{G,(l^-,r)} v \ \text{ iff } \ u \Rightarrow_{G,l^-} v \ \vee \ u \Rightarrow_{G,r} v.$$

Whenever an SE-system G is understood from the context we omit the subscript G from the notation of the relations above and, as usual, by $\stackrel{*}{\Rightarrow}$ we denote the reflexive and transitive closure of the binary relation $\Rightarrow$. A derivation $u \stackrel{*}{\Rightarrow}_x v$, where x is a mode of synchronization, is called an *x-derivation* (*of u into v*, or *of v from u*, or *of v*, or *from u*). The *language of type x* generated by an *SE-system* $G = (V, L_1, L_2, S)$, where x is a mode of synchronization, is defined by

$$L^x(G) = \{v \in V^* | \exists u \in L_1 : \ u \stackrel{*}{\Rightarrow}_{G,x} v\}.$$

Let $G = (V, L_1, L_2, S)$ be an SE-system and let p_1, p_2 and p_3 be predicates on $\mathcal{P}(V^*)$ [3]. We say that G is *of type* (p_1, p_2, p_3) if the formula $p_1(L_1) \ \wedge \ p_2(L_2) \ \wedge \ p_3(S)$ holds true. We shall use the abbreviation f (i, r, cf, cs, rec, re, respectively) for the predicate "$f(L)$ iff L is finite (infinite, regular, context-free, context-sensitive, recursive, recursively enumerable, respectively)".

We consider the concept of a *pushdown automata* (*pda*, for short) as in [1]. That is, a *pda over an alphabet* V is a 5-tuple $\mathcal{A} = (Q, Z, i, K, T)$, where Q is the set of *states*, Z is the *stack alphabet*, $i \in Q \times Z^*$ is the *initial internal configuration*, $K \subseteq Q \times Z^*$ is a set of *accepting internal confugurations*, and T is a subset of $(V \cup \{\lambda\}) \times Q \times Z \times Z^* \times Q$ (each element of T being called a *transition rule*).

The elements of $Q \times Z^*$ $(V^* \times Q \times Z^*)$ are called *internal configurations* (*configurations*) of $\mathcal{A}$. The *set of internal accepting configurations* are of the form $K = F \times Z^*$, where F is a subset of Q, called the *set of accepting states.* The *transition relation* over configurations, induced by $\mathcal{A}$, is defined by:

$$(ax, q, wz) \rightarrow (x, q', w\alpha) \quad \Leftrightarrow \quad (a, q, z, \alpha, q') \in T.$$

It is also convenient to denote $(q, w) \stackrel{x}{\rightarrow} (q', w')$ instead of $(x, q, w) \stackrel{*}{\rightarrow} (\lambda, q', w')$. The *stack language of* $\mathcal{A}$ is defined as being the language

$$Stack(\mathcal{A}) = \{w \in Z^* | \exists x, y \in V^*, \ \exists q \in Q, \ \exists k \in K : \ i \stackrel{x}{\rightarrow} (q, w) \stackrel{y}{\rightarrow} k\}.$$

[3] A predicate on a non-empty set A is a function from A into the set $\{0, 1\}$.

The notation $u \leq_{suff} v$ means that the word u is a *suffix* of the word v.

2 The Result

In [1] it is shown that, for each mode of acceptance, the stack language of a pda is regular. We use this result to show that for any SE-system of type (r, r, f), the language $L^{r^-}(G)$ is regular. This fact can be considered as a converse of a result established in [2].

Theorem 1 *For any SE-system of type (r, r, f), the language $L^{r^-}(G)$ is regular.*

Proof. Let $G = (V, L_1, L_2, S)$ be an SE-system of type (r, r, f), and $G_1 = (V_N^1, V_T^1, X_0^1, P_1)$ and $G_2 = (V_N^2, V_T^2, X_0^2, P_2)$ right linear grammars generating the languages L_1 and L_2, respectively. We may assume that these two grammars have distinct sets of nonterminals.

Define the following pda $\mathcal{A} = (Q, Z, i, K, T)$ over an alphabet with one symbol x:

- $Q = \{q_0, q_1\} \cup \{q^{s'} | \exists s \in S: \ s' \leq_{suff} s\}$;
- $Z = \{z_0\} \cup V \cup V_N^1 \cup V_N^2$, where z_0 is a new symbol;
- $i = (q_0, z_0)$;
- $K = \{q_1\} \times Z^*$;
- T contains the following groups of rules:
  - $(x, q_0, z_0, z_0 X_0^1, q_0)$
    (inserts X_0^1 in the stack in order to start simulating a derivation in G_1);
  - (x, q_0, A, aB, q_0), if $A \to aB \in P_1$
    (a derivatation step in G_1)
    (x, q_0, A, a, q_1), $(x, q_0, A, a, q^\lambda)$, if $A \to a \in P_1$
    (ends the derivation with acceptance in state q_1, or prepares a synchronization in state q^λ);
  - $(x, q^{s'}, a, \lambda, q^{as'})$, whenever s' and as' are suffixes of some synchronization words
    (tries to find a synchronization word as a suffix of the stack word);
  - $(x, q^\lambda, a, aX_0^2, q_0)$, if $\lambda \in S$
    (λ is a synchronization word and, therefore, any word in L_2 could be catenated to the stack word)
  - $(x, q^{as'}, b, bX_0^2, q^{as'})$, if $as' \in S$

(as' is a synchronization word and, therefore, any word $as'v \in L_2$ could be catenated to the stack word)

- $(x, q^{as'}, A, aB, q^{s'})$, if $A \to aB \in P_2$ and $s' \neq \lambda$
  (checks the synchronization with a word in L_2)

- (x, q^a, A, aB, q_0), if $A \to aB \in P_2$
  (the synchronization is done)

- (x, q^a, A, a, q_1), if $A \to a \in P_2$
  (accepts in the case where the synchronization word is in L_2)

- (x, q_0, A, aB, q_0), if $A \to a \in P_2$
  (continues the derivation in G_2 after synchronization)

- (x, q_0, A, a, q_1), $(x, q_0, A, a, q^\lambda)$, if $A \to a \in P_2$
  (ends the derivation with acceptance in state q_1, or prepares a synchronization in state q^λ).

It is not difficult to see that the stack language of $\mathcal{A}$ is $\{z_0\}L^{r^-}(G)$. Therefore, according to [1], this language is regular. Since the family of regular languages is closed under left derivatives, we can conclude that $L^{r^-}(G)$ is regular.

The construction in the proof of the theorem above cannot be extended to the case of infinite sets of synchronization words because the set of states of a pda should be finite.

References

[1] J.-M. Autebert, J. Berstel, and L. Boasson. Context-Free Languages and Pushdown Automata, In: G. Rozenberg and A. Salomaa, (eds.), *Handbook of Formal Languages, vol. 1*, Springer, 1997, pp. 111–174.

[2] F.L. Ţiplea, E. Mäkinen, and C. Apachiţe. *Synchronized Extension Systems*, Technical Report A-2000-1, Dept. of Computer and Information Scieces, University of Tampere, Finland, January 2000. (Submitted to *Acta Inform.*)