Recognition of Linear-Slender Context-Free Languages by Real Time One-Way Cellular Automata

Véronique Terrier

Recognition of linear-slender context-free languages by real time one-way cellular automata

Véronique Terrier

GREYC, UMR 6072, Université de Caen Basse-Normandie
Campus Côte de Nacre, boulevard Maréchal Juin, 14032 Caen, France
veronique.terrier@unicaen.fr

Abstract. A linear-slender context-free language is a context-free language whose number of words of length n is linear in n. Its structure has been finely characterized in a work of Ilie, Rozenberg and Salomaa. Thanks to this characterization, we show that every linear-slender context-free language is recognizable by a real time one-way cellular automaton.

1 Introduction

One-way cellular automaton (OCA) is one of the simplest parallel recognizing devices. Since its introduction by Dyer [5], it has been the subject of much interest. Numerous studies have been directed towards the real time OCA, the class of languages decidable in minimal time, and have already provided a good understanding of its abilities and limitations.

In particular, it is known that the real time OCA and the family of context free languages are incomparable [12]. This calls into question which languages they have in common. That is not the case for deterministic CFL: there exits a LL(1) CFL which is not a real time OCA one [11]. On the other hand, linear CFL and visible pushdown languages have been proved to be real time OCA ones [3, 11].

With the aim to identify more common languages, we will take into consideration the counting function that measures the number of words of length n in the language. Noticing that all CFL known to be not real time OCA have their counting function of exponential order, the best is to look at sparse languages. These are the poly-slender CFL whose counting functions are polynomial in n and, more specifically, the linear-slender CFL whose counting functions are linear. The purpose here is to show that linear-slender CFL are real time OCA ones.

The present work essentially relies on a paper by Ilie, Rozenberg and Salomaa which presents a characterization of poly-slender CFL in terms of Dyck loops [7]. Čulík's OCA which recognizes in real time the language $\{a^n b^{n+m} a^m : n, m \geq 0\}$ and Okhotin's characterization of real time OCA by linear conjunctive grammars will come also into play [2, 10].

2 Preliminary

2.1 Basic notions

We first recall some basic definitions and notations.

For any language L, the number of words in L of length n is denoted by $\sharp_n(L)$.
For an integer k, a language L is *k-poly-slender* if $\sharp_n(L)$ is in $\mathcal{O}(n^k)$.
A language L is *poly-slender* if L is k-poly-slender for some k. A language L is *linear-slender* if $\sharp_n(L)$ is in $\mathcal{O}(n)$.

A language L is *bounded* if there exists a finite number of words $u_1, u_2, \cdots, u_k$ such that $L \subseteq u_1^* u_2^* \cdots u_k^*$.
For any word $w \in \Sigma^+$, the primitive root of w is denoted by $\rho(w)$ and corresponds to the minimal word such that $w \in (\rho(w))^*$.

2.2 Poly-slender context free languages

We review the definitions and results presented in [7] that will be fundamental ingredients throughout this paper.

Definition 1 (Dyck loop). *Let $z = z_1 z_2 \cdots z_{2k}$ be a Dyck word on $\{[,]\}$ of length $2k$. Let l_i and r_i denote the respective positions of the i-th opening parenthesis [and its corresponding closing one] in z. Given some words $y_0, \cdots, y_{2k}$, $x_1, \cdots, x_{2k}$, consider the map $h_{n_1,\cdots,n_k}(z_1 z_2 \cdots z_{2k}) = y_0 x_1^{e_1} y_1 x_2^{e_2} y_2 \cdots x_{2k}^{e_{2k}} y_{2k}$ where each pair of parentheses shares the same exponent: $e_{l_i} = e_{r_i} = n_i$.*
A k-Dyck loop *is any set $D = \{h_{n_1,\cdots,n_k}(z_1 z_2 \cdots z_{2k}) \colon n_i \geq 0\}$ for some underlying Dyck word $z_1 z_2 \cdots z_{2k}$ and words x_i, y_i.*

Example 1. $\{ab^{n_1}a(ba)^{n_2+1}a^{n_2+1}(ba)^{n_1+n_3+2}b^{n_3} \colon n_1, n_2, n_3 \geq 0\}$ is a 3-Dyck loop for the underlying Dyck word [[]][] and words $y_0 = a, x_1 = b, y_1 = aba$, $x_2 = ba, y_2 = a, x_3 = a, y_3 = \varepsilon, x_4 = ba, y_4 = bab, x_5 = ab, y_5 = a, x_6 = b, y_6 = \varepsilon$.

Example 2. $\{a^{n_1}b^{n_1+n_2}a^{n_2} \colon n_1, n_2 \geq 0\}$ is a 2-Dyck loop for the underlying Dyck word [][] and words $x_1 = x_4 = a, x_2 = x_3 = b, y_0 = y_1 = y_2 = y_4 = \varepsilon$.

The structure of poly-slender CFL finely corresponds to Dyck loops:

Theorem 1. *For any $k \geq 0$, a context-free language is k-poly-slender if and only if it is a finite union of $(k+1)$-Dyck loops.*

The following notion captures whether the position of some word w can be distinguished or not.

Definition 2 (Link). *Let $u, v \in \Sigma^+$ and $w \in \Sigma^*$. The word w* links *u with v (*$\texttt{link}(u, w, v)$*) if and only if $\rho(u)w = w\rho(v)$. That means $\rho(u) = pq$, $\rho(v) = qp$ and $w = (pq)^* p$ for some p, q. And so $u^m w v^n$ is a prefix of $(pq)^*$*

Example 3. With x_i and y_i as defined in *Example 1*, $\texttt{link}(x_4, y_4, x_5)$ holds but $\texttt{link}(x_i, y_i, x_{i+1})$ does not hold for $i = 1, 2, 3, 5$.
With x_i and y_i as defined in *Example 2*, $\texttt{link}(x_2, y_2, x_3)$ holds but neither $\texttt{link}(x_1, y_1, x_2)$ nor $\texttt{link}(x_3, y_3, x_4)$ holds.

We will make great use of the following lemma:

Lemma 1. *Consider some words $x_i \in \Sigma^+$ and $y_i \in \Sigma^*$, and some non-negative integers n_i, m_i. Denote $w = y_0 x_1^{n_1} y_1 x_2^{n_2} y_2 \cdots x_r^{n_r} y_r$, $w' = y_0 x_1^{m_1} y_1 x_2^{m_2} y_2 \cdots x_r^{m_r} y_r$ and assume that $\texttt{link}(x_i, y_i, x_{i+1})$ holds for no i.*
Then there is a constant N_0, depending only on the lengths of the words x_i and y_i, such that, if n_i, m_i are larger than N_0 and there is i with $n_i \neq m_i$, then $w \neq w'$.

For a general overview, we recall the result of Latteux and Thierrin [8] and the one of Ginsburg and Spanier [6]:

Theorem 2. *A context-free language is poly-slender if and only if it is bounded.*

Theorem 3. *The family of bounded languages is the smallest family which contains all finite languages and is closed under the following operations:*

1. *union*
2. *catenation*
3. $(x, y)^\star L = \bigcup_{n \geq 0} x^n L y^n$ *for any x, y words*

2.3 Real time one-way cellular automaton

A *one-way cellular automaton* is a one-dimensional array of finite automata (the cells) indexed by $\mathbb{N}$. The cells evolve synchronously at discrete time steps. Each cell takes one value from a finite set of states Q. At each step, the evolution of a cell is defined by its own state and the state of its right neighbor according to a transition function δ. Formally, denoting $\langle c, t \rangle$ the state of the cell c at time t, $\langle c, t+1 \rangle = \delta\left(\langle c, t \rangle, \langle c+1, t \rangle\right)$.

In order that an OCA acts as a language recognizer, we specify two subsets of Q: the alphabet Σ of input symbols and the set of accepting states Q_{accept}. The input mode is parallel. At initial time 0, the i-th bit of the input word $w \in \Sigma^*$ is fed to the cell i: $\langle i, 0 \rangle = w_i$. An OCA is said to accept (resp. reject) a word w in real time, if on input w the cell 0 enters an accepting (resp. non-accepting) state at time $|w| - 1$. It corresponds to the minimal time for the cell 0 to know the whole input. A language is a *real time OCA language* if there exists some OCA $(Q, \Sigma, Q_{\text{accept}}, \delta)$ which accepts in real time exactly the words $w \in L$.

The computation of an OCA is usually represented by a time-space diagram (see Fig 1). The t-row corresponds to the cellular array at time t. Only those sites involved in the real time computation are depicted. As a matter of fact, there are two topologically equivalent ways to display the time-space diagram. We will use here the bilateral symmetric layout, i.e., the right one.

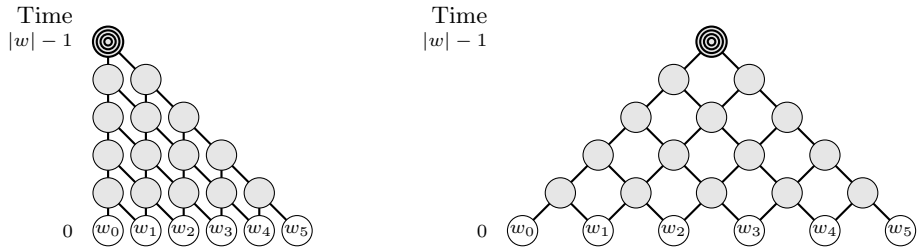


Fig. 1. Time-space diagram of a real time OCA

The real time OCA class is robust: introducing the notion of conjunctive grammar, Okhotin has exhibited a characterization of real time OCA in terms of a generating device [10]. Let us recall its statement, it will be one of the main ingredients in this paper. An alternating grammar is a grammar enhanced with a conjunctive operation symbolized by &. Denoting N the set of variables and Σ the set of terminals, each production is of the form $A \to \alpha_1 \,\&\, \cdots \,\&\, \alpha_k$ where $A \in N$ and $\alpha_1, \cdots, \alpha_k \in (A \cup \Sigma)^*$. Such a production denotes that the language generated by A is the intersection of the languages generated by $\alpha_1, \cdots, \alpha_k$. Analogously to linear context free grammars, a linear conjunctive grammar is defined as an alternating grammar with the restriction that for every production $A \to \alpha_1 \,\&\, \cdots \,\&\, \alpha_k$, no α_i has more than one instance of a variable. An algorithm describing how to recognize any linear CFL on a real time OCA was already known [3]. More radically, to extend linear context free grammars with the conjunctive operation & leads to a complete characterization of real time OCA [10]:

Theorem 4. *A language L is recognized in real time by an OCA if and only if L is generated by a linear conjunctive grammar.*

Let us recall the conversion from a real time OCA to a linear conjunctive grammar that we will need later. See Fig. 2 to get some insight about the translation.

Lemma 2. *Given any language L recognizable by some real time OCA $\mathcal{A} = (Q, \Sigma, Q_{accept}, \delta)$, L is generated by the linear conjunctive grammar $\mathcal{G} = (\Sigma, \{S\} \cup \{A_q \colon q \in Q\}, S, \mathcal{R})$ where $\mathcal{R}$ contains the following rules:*

$$S \to A_q \text{ for all } q \in Q_{accept}$$
$$A_a \to a \text{ for all } a \in \Sigma$$
$$A_{\delta(g,d)} \to A_g b \,\&\, c A_d \text{ for all } g, d \in Q \text{ and all } b, c \in \Sigma$$

At last, let us give some examples and properties of real time OCA to illustrate their abilities and limits. The Dyck language, the linear-slender CFL $\{a^n b^{n+m} a^m \colon n, m \geq 0\}$, the inherently exponentially ambiguous CFL L^* with $L = \{a^i b^j c^k \colon i = j \text{ or } j = k\}$ [9], the non CFL languages $\{a^n b^n c^n \colon n \geq 0\}$ and $\{a^n b^{2^n} \colon n \geq 0\}$ with a non semilinear Parikh image, are all real time OCA languages. In addition, real time OCA languages include all linear CFL and visible

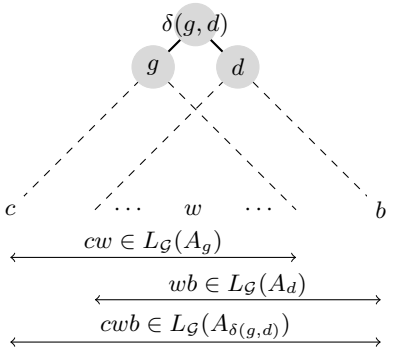


Fig. 2. The OCA's transition $\delta(g,d)$ is converted into the rule $A_{\delta(g,d)} \to A_g b \,\&\, cA_d$

pushdown languages [3, 11]. On the other hand, several languages are known not to be real time OCA: all non regular languages over a one letter alphabet, the inherently ambiguous CFL L_1L_1 square of the linear CFL $L_1 = \{1^k 0 u 1 0^k : k > 0, u \in \{0,1\}^*\}$, the language $\{uvu : u, v \in \{0,1\}^*, |u| > 1\}$, the deterministic CFL (to be more precise: LL(1) language) $\{c^m a^{l_0} b a^{l_1} b \cdots a^{l_m} b \cdots a^{l_z} b d^n : m, n, l_i \geq 0, z \geq 1, l_m = n\}$ [1, 12, 13, 11].
Further, the real time OCA class is closed under boolean operations, reverse, $\star$ operation (the one defined in Theorem 3), left and right concatenation with regular languages. The proofs are folklore. A contrario, the real time OCA class is not closed under morphism, concatenation, Kleene star and cycle [4, 12, 14].

2.4 Poly-slender context free languages and real time one-way cellular automata

The question behind this paper is whether the poly-slender CFL are real time OCA languages. According to Theorem 3, the poly-slender CFL are the smallest family which contains all finite languages and is closed under union, concatenation and $\star$ operation. Of course, real time OCA languages include all finite languages and are closed under union and $\star$ operation. This is easy to prove using the grammar characterization of real time OCA. The problematic point is concatenation: can we assert that the concatenation of two Dyck loops is a real time OCA language? We do not answer this question but in the simple case: 1-Dyck loops. Precisely we will show the following result:

Theorem 5. *Linear-slender context-free languages are recognizable in real time by one-way cellular automata.*

The rest of the paper will be devoted to the proof of this theorem. Concretely we have to show that 2-Dyck loops are recognized in real time by OCA.

3 Recognition of 2-Dyck loops by real time OCA

As an introduction, we may observe that 1-Dyck loops are real time OCA languages. Indeed a 1-Dyck loop corresponds to a set $D = \{y_0x_1^n y_1 x_2^n y_2 \colon n \in \mathbb{N}\}$ for some words y_0, x_1, y_1, x_2, y_2. It is a linear CFL and so it is a real time OCA language. As a consequence, all constant-slender languages are real time OCA languages.

Let us now consider 2-Dyck loops. According to whether the underlying Dyck word is [[]] or [][], they are of shape $\{y_0x_1^{n_1}y_1x_2^{n_2}y_2x_3^{n_2}y_3x_4^{n_1}y_4 \colon n_1, n_2 \geq 0\}$ or $\{y_0x_1^{n_1}y_1x_2^{n_1}y_2x_3^{n_2}y_3x_4^{n_2}y_4 \colon n_1, n_2 \geq 0\}$ A first simplification is to assume that the two ends y_0 and y_4 are empty knowing that if L is a real time OCA language then so is $\{y_0\}L\{y_4\}$ whatever the words y_0 and y_4 are. We will suppose also that the words x_i are non-empty, the degenerate cases being easy to handle.

3.1 The underlying Dyck word is [[]].

That is the simple case. The corresponding Dyck loop $\{x_1^{n_1}y_1x_2^{n_2}y_2x_3^{n_2}y_3x_4^{n_1} \colon n_1, n_2 \geq 0\}$ is clearly a linear CFL and so it is real time OCA recognizable. Here it is basically the closure under the $\star$ operation which is involved.

3.2 The underlying Dyck word is [][].

Now it is the closure under concatenation of 1-Dyck loops which is involved.

Case 1. $\texttt{link}(x_i, y_i, x_{i+1})$ holds for *no* $i = 1, 2, 3$

Let N_0 be a constant as defined in Lemma 1. The Dyck loop can be divided into three subsets D_1, D_2 and D_3, where

$$\begin{aligned} D_1 &= \{x_1^{n_1}y_1x_2^{n_1}y_2x_3^{n_3}y_3x_4^{n_3} \colon n_1 < N_0, n_3 \in \mathbb{N}\},\\ D_2 &= \{x_1^{n_1}y_1x_2^{n_1}y_2x_3^{n_3}y_3x_4^{n_3} \colon n_1 \in \mathbb{N}, n_3 < N_0\},\\ D_3 &= \{x_1^{n_1}y_1x_2^{n_1}y_2x_3^{n_3}y_3x_4^{n_3} \colon n_1, n_3 \geq N_0\}. \end{aligned}$$

Observe that $D_1 = \bigcup_{0 \leq i < N_0} \{x_1^i y_1 x_2^i y_2 x_3^n y_3 x_4^n \colon n \in \mathbb{N}\}$ is a finite union of linear CFL's and thus is alinear CFL. The subset D_2 is as well a linear CFL. Further, Lemma 1 ensures that D_3 can be specified as the intersection of two linear CFL's : $D_3 = \{x_1^{n_1}y_1x_2^{n_1}y_2x_3^{n_3}y_3x_4^{n_4} \colon n_1, n_3, n_4 \geq N_0\} \cap \{x_1^{n_1}y_1x_2^{n_2}y_2x_3^{n_3}y_3x_4^{n_3} \colon n_1, n_2, n_3 \geq N_0\}$ and so is real time recognizable by an OCA.

Case 2. $\texttt{link}(x_i, y_i, x_{i+1})$ holds for *one* $i = 1, 2, 3$

The case where y_1 links x_1 with x_2 (or in a symmetric way y_3 links x_3 with x_4) does not present any difficulty. By Definition 2, y_1 links x_1 with x_2 if $\rho(x_1) = pq$, $\rho(x_2) = qp$ and $y_1 = (pq)^\gamma p$ for some words p, q and integer γ. Setting $x_1 = (pq)^\alpha$, $x_2 = (qp)^\beta$, $x_1^{n_1}y_1x_2^{n_2}y_2$ can be rewritten as $(pq)^{(\alpha+\beta)n_1+\gamma}py_2$ and thus specifies a regular language. Moreover $x_3^{n_3}y_3x_4^{n_4} = x_3^{n_3}y_3x_4^{n_3}$ corresponds to a linear CFL. Their concatenation is linear CFL and so real time recognizable by an OCA.

It remains to examine the most technical case where y_2 links x_2 with x_3 but $\texttt{link}(x_i, y_i, x_{i+1})$ does not hold for $i = 1$ and $i = 3$. As y_2 links x_2 with x_3 there exists some p, q and α, β, γ such that $x_2 = (pq)^\alpha$, $x_3 = (qp)^\beta$ and $y_2 = (pq)^\gamma p$. Then $x_1^{n_1} y_1 x_2^{n_2} y_2 x_3^{n_3} y_3 x_4^{n_4}$ can be rewritten as $x_1^{n_1} y_1 (pq)^{\alpha n_1 + \beta n_3 + \gamma} p y_3 x_4^{n_3}$. Setting $z_1 = x_1, z_2 = y_1, z_3 = pq, z_4 = py_3, z_5 = x_4$, such 2-Dyck loops can be reshaped into $z_1^{n_1} z_2 z_3^{\alpha n_1 + \beta n_3 + \gamma} z_4 z_5^{n_3}$ with the properties that z_2 does not link z_1 with z_3 and, readily verifiable, z_4 does not link z_3 with z_5. So our task is to show that $\{z_1^n z_2 z_3^{\alpha n + \beta m + \gamma} z_4 z_5^m : n, m \geq 0\}$ is a real time OCA language. It will be done in two steps:

1. Given a five letters alphabet $\mathcal{A} = \{a_1, \cdots, a_5\}$, whatever $\alpha, \beta \geq 1$ and $\gamma \geq 0$ are, we will present a real time OCA which recognizes the language $L_{\alpha,\beta,\gamma} = \{a_1^n a_2 a_3^{\alpha n + \beta m + \gamma} a_4 a_5^m : n, m \geq 0\}$.
2. For any homomorphism h on $\mathcal{A}$ such that neither $\texttt{link}(h(a_1), h(a_2), h(a_3))$ nor $\texttt{link}(h(a_3), h(a_4), h(a_5))$ holds, we will verify that $h(L_{\alpha,\beta,\gamma})$ is a real time OCA language.

Proposition 1. *The language $L_{\alpha,\beta,\gamma} = \{a_1^n a_2 a_3^{\alpha n + \beta m + \gamma} a_4 a_5^m : n, m > 0\}$, where the symbols $a_1, \cdots, a_5$ are distinct, is recognizable in real time by an OCA.*

Proof. The main ingredient is Čulík's OCA which recognizes in real time the language $\{a^n b^{n+m} a^m : n, m \geq 0\}$ [2]. His construction makes ingenious use of a firing squad synchronization.

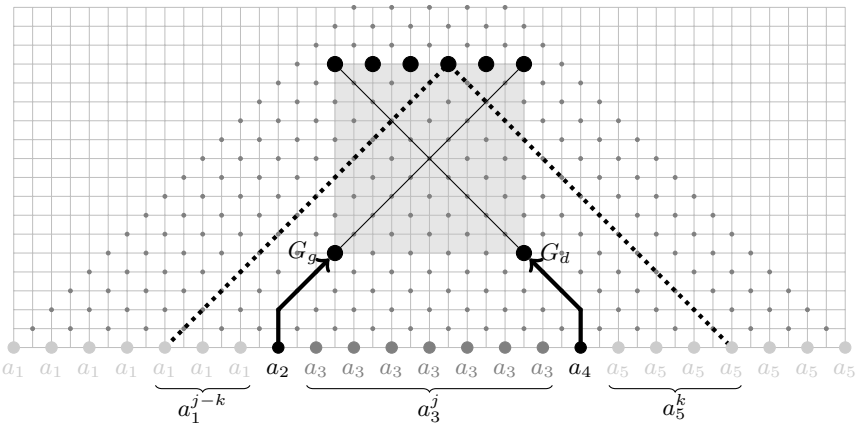
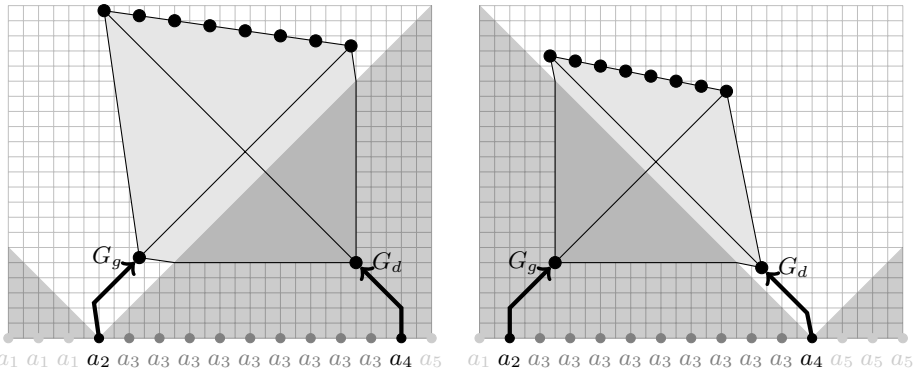


Fig. 3. Čulík's OCA

Let us begin by recalling the process (see Fig. 3). For convenience, we identify the sites of the OCA with integer coordinates (x, y) where $x + y$ is even. In this way, the input symbols are placed at positions $(2x, 0)$. The unique symbol a_2 is chosen to be at the origin $(0, 0)$ and, when the input is of shape $a_1^+ a_2 a_3^j a_4 a_5^+$,

the unique symbol a_4 is at $(2j+2, 0)$. The process is set up using a firing squad synchronization with two generals G_g and G_d located according to the unique symbols a_2 and a_4. Precisely, the left general G_g is at $(3, 5)$ and the right general G_d at $(2j-1, 5)$. Thus the synchronization occurs at points $(1+2k, 1+2j)$ for all k with $0 < k < j$. We check that the k-th firing points $(1+2k, 1+2j)$ corresponds to the output of the input subword which begins at $(2(k-j), 0)$ and ends at $(2(1+j+k), 0)$, namely the subword $a_1^{j-k} a_2 a_3^j a_4 a_5^k$.

Next, Čulík showed that the construction can be adapted to recognize the languages $\{a^i b^j a^k \colon i, k \geq 0, mj + c = i + k\}$ for all $m \geq 1$, $c \geq 0$. Actually, the same approach works in these more general settings: it exists a real time OCA deciding the language $L_{\alpha,\beta,\gamma} = \{a_1^i a_2 a_3^j a_4 a_5^k \colon i, j, k \in \mathbb{N}, j = \alpha i + \beta k + \gamma\}$ for every non-negative rational numbers α, β, γ. Let us outline the modified construction. Be warned that we will use rational coordinates but the technique to revert to an OCA diagram is standard. Firstly observe that we may reduce the range of the synchronization in locating the two generals later: with the left general G_g at $(3+\gamma, 5+\gamma)$ and the right general G_d at $(2j-1-\gamma, 5+\gamma)$, the set of firing points becomes $\{(1+2k+\gamma, 1+2j-\gamma) \colon k \in \mathbb{N}, 0 < k < j-\gamma\}$. We explain now the construction in terms of geometric transformations. It will be achieved by the composition of two directional scalings (see Fig. 4).



(a) Scaling in the up-left to down-right direction with factor $1/\alpha = 4/3$

(b) Scaling in the up-right to down-left direction with factor $1/\beta = 2/3$

Fig. 4. Two scalings

Initiated from the unique symbol a_2, the first transformation starts at point $(0, 0)$ and applies on its cone of consequences $\{(c, t) \colon |c| \leq t\}$, i.e., the future light cone of $(0, 0)$ which encompasses the set of points impacted by $(0, 0)$. It leaves the line $t = c$ stable and scales by the factor $1/\alpha$ in the up-left to down-right direction according to the map $M_g = \begin{pmatrix} (1+\alpha)/(2\alpha) & (\alpha-1)/(2\alpha) \\ (\alpha-1)/(2\alpha) & (1+\alpha)/(2\alpha) \end{pmatrix}$. Observe that the transformation is workable. Inside the cone, the neighborhood constraints

are satisfied. The right side of the cone is stable. And the computation on the subword a_1^+ occurring below the left side can easily be scaled.

The second transformation is symmetric. Initiated from the unique symbol a_4, it starts at point $(2j+2, 0)$ and applies on its cone of consequences $\{(c,t)\colon |c-2j-2| \leq t\}$. It leaves the line $t+c = 2j+2$ stable and scales by the factor $1/\beta$ in the up-right to down-left direction according to the map $M_d = \left(\begin{smallmatrix} (1+\beta)/(2\beta) & (1-\beta)/(2\beta) \\ (1-\beta)/(2\beta) & (1+\beta)/(2\beta) \end{smallmatrix}\right)$.

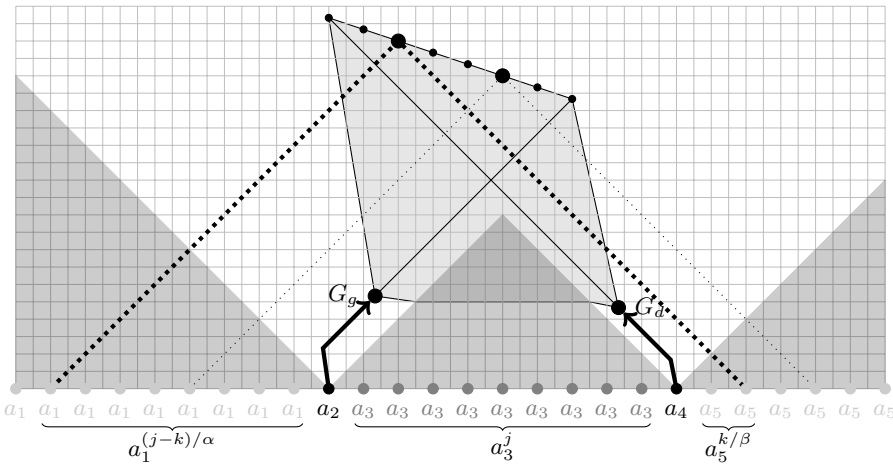


Fig. 5. Composition of two scalings

Now, the composition of these two scalings applies inside the cone of consequences of $(j+1, j+1)$: $\{c,t)\colon |c-j-1| \leq t-j-1\}$, intersection of the two cones issued from a_2 and a_4. See Fig. 5. It corresponds to the affine transformation with origin $(j+1, j+1)$ and matrix $M = M_g \times M_d = \left(\begin{smallmatrix} (\alpha+\beta)/(2\alpha\beta) & (\alpha-\beta)/(2\alpha\beta) \\ (\alpha-\beta)/(2\alpha\beta) & (\alpha+\beta)/(2\alpha\beta) \end{smallmatrix}\right)$. Thus the firing points $\{(1+2k+\gamma, 1+2j-\gamma)\colon k \in \mathbb{N}, 0 < k < j-\gamma\}$ being inside the cone of $(j+1, j+1)$ are mapped to the points $\{(1+j-(j-k-\gamma)/\alpha + k/\beta,$ $1+j+(j-k-\gamma)/\alpha+k/\beta)\colon k \in \mathbb{N}, 0 < k < j-\gamma\}$. Among these points, some of them match OCA's sites providing k/β and $(j-k-\gamma)/\alpha$ are integers. Moreover such a point, with $(j-k-\gamma)/\alpha, k/\beta \in \mathbb{N}$, corresponds to the output of the input subword that begins at $(2(j-k-\gamma)/\alpha, 0)$ and ends at $(2j+2+2k/\beta, 0)$, i.e., the subword $a_1^n a_2 a_3^j a_4 a_5^m$ with $n = (j-k-\gamma)/\alpha$, $m = k/\beta$. To conclude just note that $\alpha n + \beta m + \gamma = j$. □

Proposition 2. *For any homomorphism h on $\mathcal{A}$ such that neither* $\texttt{link}(h(a_1), h(a_2), h(a_3))$ *nor* $\texttt{link}(h(a_3), h(a_4), h(a_5))$ *holds, $h(L_{\alpha,\beta,\gamma})$ is a real time OCA language.*

Proof. One has to be careful because real time OCA is not closed under morphism: each recursively enumerable language can be written as the image of a real time OCA language by a morphism [4]. But here we play with the image of languages with very simple structure.

First observe that $h(\{a_1^n a_2 a_3^{\alpha n+\beta m+\gamma} a_4 a_5^m : n < N_0 \text{ or } m < N_0\})$ is a linear CFL, so to show that $h(\{a_1^n a_2 a_3^{\alpha n+\beta m+\gamma} a_4 a_5^m : n, m \geq N_0\})$ is a real time OCA language will suffice. According to Proposition 1, there exists a real time OCA $(Q, \mathcal{A}, Q_{accept}, \delta)$ which recognizes the language $L = \{a_1^n a_2 a_3^{\alpha n+\beta m+\gamma} a_4 a_5^m : n, m \geq N_0\}$. Following Okhotin [10], L is defined by the linear conjunctive grammar $\mathcal{G} = (\mathcal{A}, \{S\} \cup \{A_q : q \in Q\}, S, \mathcal{R})$ where $\mathcal{R}$ contains the following rules:

$$S \to A_q \text{ for all } q \in Q_{accept}$$
$$A_a \to a \text{ for all } a \in \mathcal{A}$$
$$A_{\delta(g,d)} \to A_g b \,\&\, c A_d \text{ for all } g, d \in Q \text{ and all } b, c \in \mathcal{A}$$

Now let us make explicit, in the grammar rules, the bounded feature of the language. All words are of shape $a_1^+ a_2 a_3^+ a_4 a_5^+$. We denote by $\texttt{Follow}(i)$ the set of j such that a_j follows immediately a_i in the expression $a_1^+ a_2 a_3^+ a_4 a_5^+$: $\texttt{Follow}(1) = \{1, 2\}$, $\texttt{Follow}(2) = \{3\}$, $\texttt{Follow}(3) = \{3, 4\}$, $\texttt{Follow}(4) = \{5\}$, $\texttt{Follow}(5) = \{5\}$. Then we rewrite $\mathcal{G}$ to $\mathcal{G}' = (\mathcal{A}, \{S\} \cup \{(A_q, 1, 5) : q \in Q_{accept}\} \cup \{(A_q, i, j) : q \in Q \backslash Q_{accept}, 1 \leq i \leq j \leq 5\}, S, \mathcal{R}')$ where $\mathcal{R}'$ contains the following rules:

$$S \to (A_q, 1, 5) \text{ for all } q \in Q_{accept}$$
$$(A_{a_i}, i, i) \to a_i \text{ for all } i = 1, \cdots, 5$$
$$(A_{\delta(g,d)}, i, j) \to (A_g, i, s) a_j \,\&\, a_i (A_d, r, j) \text{ for all } g, d \in Q \text{ and all } i, j, r, s$$
$$\text{with } 1 \leq i \leq r \leq s \leq j \leq 5,\ r \in \texttt{Follow}(i),\ j \in \texttt{Follow}(s)$$

To gain a better understanding of the last derivation rule, see Fig. 6, it depicts the corresponding OCA transition. With such refinements, we get that $L_{\mathcal{G}'}(A_q, i, j)$ is the subset of $L_{\mathcal{G}}(A_q)$ whose words begin with a_i, ends with a_j and which are factors of $a_1^+ a_2 a_3^+ a_4 a_5^+$.

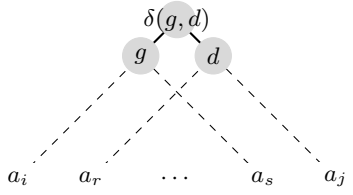


Fig. 6. Interpretation of the rule $(A_{\delta(g,d)}, i, j) \to (A_g, i, s) a_j \,\&\, a_i (A_d, r, j)$ with $i \leq r \leq s \leq j$, $r \in \texttt{Follow}(i)$, $j \in \texttt{Follow}(s)$

In addition, to meet later requirements, we replace all rules $(A_{a_i}, i, i) \to a_i$ with the rules $(A_{\delta(a_{i_1}, \cdots, a_{i_{N_0}})}, i_1, i_{N_0}) \to a_{i_1} \cdots a_{i_{N_0}}$ for all words $a_{i_1} \cdots a_{i_{N_0}}$ of

length N_0 which are factors of $a_1^{N_0}a_2a_3^{N_0}a_4a_5^{N_0}$. It does not alter the expressiveness of the grammar.

Finally, in replacing each symbol a_i with its image $h(a_i)$, $\mathcal{G}$ is rewritten to $\mathcal{H} = (\Sigma, \{S\} \cup \{(A_q, i, j) \colon q \in Q, 1 \leq i \leq j \leq 5\}, S, \mathcal{R}'')$ where $\mathcal{R}''$ contains the following rules:

$$
\begin{aligned}
&S \to (A_q, 1, 5) \text{ for all } q \in Q_{accept}\\
&(A_{\delta(a_{i_1}, \cdots, a_{i_{N_0}})}, i_1, i_{N_0}) \to h(a_{i_1} \cdots a_{i_{N_0}}) \text{ for all words } a_{i_1} \cdots a_{i_{N_0}} \text{ of}\\
&\qquad\qquad \text{length } N_0 \text{ which are factors of } a_1^{N_0}a_2a_3^{N_0}a_4a_5^{N_0}\\
&(A_{\delta(g,d)}, i, j) \to (A_g, i, s)h(a_j) \,\&\, h(a_i)(A_d, r, j) \text{ for all } g, d \in Q \text{ and all}\\
&\qquad i, j, r, s \text{ with } 1 \leq i \leq r \leq s \leq j \leq 5,\ r \in \texttt{Follow}(i),\ j \in \texttt{Follow}(s)
\end{aligned}
$$

Clearly, $h(L_{\mathcal{G}}(S)) \subseteq L_{\mathcal{H}}(S)$. Let us ensure that they are equal and, more specifically, that $L_{\mathcal{H}}(A, i, j) \subseteq h(L_{\mathcal{G}}(A, i, j))$ for every variable (A, i, j). The proof is done by induction on the height of the parse trees. The inductive assumption is that all words generated within $\mathcal{H}$ by a tree of height h and root node (A, i, j) are images by h of words generated within $\mathcal{G}$ by a tree of height h and root node (A, i, j).

The base case. The trees of height 1 within $\mathcal{H}$ display the derivations $(A_q, i_1, i_{N_0}) \to h(a_{i_1} \cdots a_{i_{N_0}})$ whereas their counterparts within $\mathcal{G}$ display the derivations $(A_q, i_1, i_{N_0}) \to a_{i_1} \cdots a_{i_{N_0}}$.

The inductive step. We focus on the trees with root node $(A_q, 1, 5)$ and omit the ones with root (A_q, i, j) when $i > 1$ or $j < 5$ that are easier to handle. Given any tree T within $\mathcal{H}$ of height $h+1 > 1$ and root $(A_q, 1, 5)$. The root node expands into two subtrees of height at most h according to some rule $(A_q, 1, 5) \to (A_g, 1, s)h(a_5) \,\&\, h(a_1)(A_d, r, 5)$ where $\delta(g, d) = q$, s is 4 or 5 and r is 1 or 2. By assumption, $(A_g, 1, s)$ produces the image by h of a word $a_1^{n_1}a_2a_3^{n_2}a_4a_5^{n_3}$ with $n_1 > 0, n_2 > N_0, n_3 \geq 0$ and $(A_d, r, 5)$ produces the image by h of a word $a_1^{m_1}a_2a_3^{m_2}a_4a_5^{m_3}$ with $m_1 \geq 0, m_2 > N_0, m_3 > 0$. Hence the root node produces $h(a_1^{n_1}a_2a_3^{n_2}a_4a_5^{n_3+1}) = h(a_1^{m_1+1}a_2a_3^{m_2}a_4a_5^{m_3})$. Next we may notice that, in the proof of Lemma 1, the hypothesis that the first exponents n_1 and m_1 are greater than N_0 is not used, as well, (in handling the word backward) for the last exponents n_r and m_r. Bearing in mind that neither $\texttt{link}(h(a_1), h(a_2), h(a_3))$ nor $\texttt{link}(h(a_3), h(a_4), h(a_5))$ holds, all prerequisites to apply Lemma 1 are satisfied. So $n_1 = m_1 + 1, n_2 = m_2, n_3 + 1 = m_3$. It follows that $a_1^{n_1}a_2a_3^{n_2}a_4a_5^{n_3+1}$ is indeed represented by a tree within $\mathcal{G}$ structured as T. □

4 Conclusion

Based on the precise characterization of the poly-slender CFL in terms of Dyck loops given by Ilie, Rozenberg and Salomaa, we have shown that linear-slender CFL are real time OCA languages. More than the result in itself, the approach appears interesting. It mixes algorithmic constructions with grammar tools and combinatoric properties: Starting from a real time OCA recognizing a specific language, here a variant of Čulík's one, we make use of Okhotin's result to

translate it in terms of a linear conjunctive grammar. Then in modifying the grammar, we capture a wider set of real time OCA languages structured like the starting one.

Now the challenge would be to show that poly-slender CFL are also real time OCA languages, in other words, that all Dyck loops are real time OCA languages. For that purpose, a key step would be to exhibit a generalization of Čulík's construction to handle languages of shape $a_0 b_1^{n_1} a_1 b_2^{n_2} a_2 \cdots b_r^{n_r} a_r$ where the integers $n_1, \cdots, n_r$ are linear combinations of k integers and the symbols a_i, b_i are distinct.

References

1. Christian Choffrut and Karel Čulík II. On real-time cellular automata and trellis automata. *Acta Informatica*, 21(4):393–407, November 1984.
2. Karel Čulík II. Variations of the firing squad problem and applications. *Information Processing Letters*, 30(3):152 – 157, 1989.
3. Karel Čulík II, Jozef Gruska, and Arto Salomaa. Systolic trellis automata. I, II. *International Journal Computer Mathematics*, 16:3–22, 1984.
4. Karel Čulík II, Jozef Gruska, and Arto Salomaa. Systolic trellis automata: Stability, decidability and complexity. *Information and Control*, 71(3):218–230, 1986.
5. Charles R. Dyer. One-way bounded cellular automata. *Information and Control*, 44(3):261–281, 1980.
6. Seymour Ginsburg and Edwin H. Spanier. Bounded algol–like languages. *Transactions of the American Mathematical Society*, 113:333–368, 1964.
7. Lucian Ilie, Grzegorz Rozenberg, and Arto Salomaa. A characterization of poly-slender context-free languages. *Theoretical Informatics and Applications*, 34(1):77–86, 2000.
8. Michel Latteux and Gabriel Thierrin. On bounded context-free languages. *Elektronische Informationsverarbeitung und Kybernetik*, 20(1):3–8, 1984.
9. Mohamed Naji. Ambiguity of context-free languages as a function of the word length. FB Informatik, Goethe Universität, Frankfurt am Main, 1998.
10. Alexander Okhotin. On the equivalence of linear conjunctive grammars and trellis automata. *RAIRO Informatique Théorique et Applications*, 38(1):69–88, 2004.
11. Alexander Okhotin. Comparing linear conjunctive languages to subfamilies of the context-free languages. In *SOFSEM 2011: Theory and Practice of Computer Science, Nový Smokovec, Slovakia, LNCS 6543*, pages 431–443, 2011.
12. Véronique Terrier. On real time one-way cellular array. *Theoretical Computer Science*, 141(1–2):331–335, 1995.
13. Véronique Terrier. Language not recognizable in real time by one-way cellular automata. *Theoretical Computer Science*, 156(1–2):281–287, 1996.
14. Véronique Terrier. Closure properties of cellular automata. *Theoretical Computer Science*, 352(1-3):97–107, 2006.