Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 387 (2007) 113–124
Theoretical Computer Science
www.elsevier.com/locate/tcs

An infinite hierarchy induced by depth synchronization☆

Franziska Biegler[a,*], Ian McQuillan[b], Kai Salomaa[c]

[a] *Department of Computer Science, University of Western Ontario, London, ON, N6A 5B7, Canada*
[b] *Department of Computer Science, University of Saskatchewan, Saskatoon, SK S7N 5A9, Canada*
[c] *School of Computing, Queen's University, Kingston, ON K7L 3N6, Canada*

Abstract

Depth-synchronization measures the number of parallel derivation steps in a synchronized context-free (SCF) grammar. When not bounded by a constant the depth-synchronization measure of an SCF grammar is at least logarithmic and at most linear with respect to the word length. Languages with linear depth-synchronization measure and languages with a depth-synchronization measure in between logarithmic and linear are proven to exist. This gives rise to a strict infinite hierarchy within the family of SCF (and ET0L) languages.

Keywords: Synchronization; ET0L languages; Regulated rewriting; Infinite hierarchy

1. Introduction

Context-free languages are among the best studied and understood families of formal languages. Unfortunately, their generative power is insufficient to model phenomena of formal and natural languages, see e.g. [1]. Context-sensitive languages, the next level in the Chomsky-hierarchy, are so powerful that they become difficult to handle. For this reason different extensions of context-free grammars have been proposed, see e.g. [1–3], in order to increase the generative capacity while maintaining as many of the desired properties of context-free languages as possible.

In [4], Jürgensen and Salomaa introduced synchronized context-free (SCF) grammars as well as block-synchronized context-free (BSCF) grammars, in which independent paths in a context-free derivation can communicate in order to synchronize by means of situation symbols. Different aspects of SCF and BSCF grammars where studied in [4–7]. The idea of synchronization as a method of communication was proposed in a similar way for automata in [8]. Measuring the amount of synchronization in SCF grammars and languages by functions was first done in [9], where the total number of situation symbols used to generate a word was used as a measure.

☆ A preliminary version of this paper appeared in DCFS 2006 [F. Biegler, I. McQuillan, K. Salomaa, An infinite hierarchy induced by depth synchronization, in: H. Leung, G. Pighizzini (Eds.), 8th International Workshop on Descriptional Complexity of Formal Systems, Las Cruces, NM, USA, 2006, pp. 82–93].

* Corresponding address: Department of Computer Science, The University of Western Ontario, Middlesex College, London, ON, N6A 5B7, Canada.
E-mail addresses: fbiegler@csd.uwo.ca (F. Biegler), mcquillan@cs.usask.ca (I. McQuillan), ksalomaa@cs.queensu.ca (K. Salomaa).

0304-3975/$ - see front matter
doi:10.1016/j.tcs.2007.07.031

In this paper we measure the amount of synchronization used in an SCF grammar by mapping an integer onto the number of symbols in the longest situation sequence needed to derive a word of this length. This definition is then extended to languages.

As our main results we show that any non-context-free language generated by an SCF grammar requires at least logarithmic and at most linear synchronization depth and that there exist languages that require linear synchronization depth. Furthermore we show that for each natural number $k \geq 1$ there are languages that, modulo a constant, require exactly synchronization depth $n^{\frac{1}{k}}$ and, thus, the depth measure gives rise to a strict infinite hierarchy of language families between logarithmic and linear depth.

2. Preliminaries

Let $\mathbb{N}_0$ and $\mathbb{N}$ be the set of non-negative and positive integers, respectively, and let $\mathbb{R}_0^+$ be the set of non-negative real numbers.

An alphabet A is a finite, non-empty set of symbols. The set of all words over A is denoted by A^*, and this set contains the empty word, λ. A language L over A is any subset of A^*. For a word $x \in A^*$ let $|x|$ denote the length of x. We let $|x|_a$ be the number of occurrences of a's in x, for $a \in A$, and we let $\text{alph}(x) = \{a \mid a \in A, |x|_a > 0\}$. We say x is a prefix (suffix, infix) of y, denoted $x \leq_\text{p} y$ ($x \leq_\text{s} y$, $x \leq_\text{i} y$), if $y = xu$ ($y = ux$, $y = uxv$) for some words $u, v \in A^*$. Also, $w_1 \simeq_\text{p} w_2$ if and only if one of w_1 or w_2 is a prefix of the other. We also say $w_1 \simeq_\text{e} w_2$ if and only if $w_1 = w_2$. For a word w with $|w| = n$ and $k \leq n$ we denote by $w[k]$ the prefix of length k of w. We use $\subseteq$, $\subset$ and $\setminus$ to denote subset, proper subset and set difference.

A *context-free grammar* is denoted by $G = (N, T, P, I)$, where N and T are disjoint alphabets of nonterminals and terminals respectively, $I \in N$ is the starting nonterminal, and P is a finite set of productions of the form $X \to w$ where $X \in N$ and $w \in (V \cup T)^*$. Derivations of context-free grammars can be represented as trees. A *tree domain* D is a non-empty finite subset of $\mathbb{N}^*$ such that

1. if $\mu \in D$, then every prefix of μ belongs to D;
2. for every $\mu \in D$ there exists $i \geq 0$ such that $\mu j \in D$ if and only if $1 \leq j \leq i$.

Let A be a set. An *A-labelled tree* is a mapping $t : D \to A$, where D is a tree domain. Elements of D are called nodes of t and D is said to be the domain of t, $\text{dom}(t)$. A node $\mu \in \text{dom}(t)$ is labelled by $t(\mu)$. A node $\lambda \in \text{dom}(t)$, denoted by $\text{root}(t)$, is called the root of t. The set of leaves of t is denoted by $\text{leaf}(t)$. The subtree of t at node μ is t/μ. The set of subtrees of t is $\text{sub}(t)$ which we extend to sets of trees T by $\text{sub}(T) = \bigcup_{t \in T} \text{sub}(t)$. When there is no confusion, we refer to a node simply by its label.

Nodes of a tree t that are not leaves are called *inner nodes* of t. The *inner tree of t,* $\text{inner}(t)$ is the tree obtained from t by cutting off all the leaves. The *yield* of an A-labelled tree t, $\text{yd}(t)$, is the word obtained by concatenating the labels of the leaves of t from left to right; the leaves are ordered by the lexicographic ordering of $\mathbb{N}^*$. For $\mu \in \text{dom}(t)$, $\text{path}_t(\mu)$ is the sequence of symbols of A occurring on the path from the root of t to the node μ. A node μ of a tree t is said to be of maximal distance from the root with respect to a certain property if for each other node ν of t which also has the property we have $|\text{path}_t(\nu)| \leq |\text{path}_t(\mu)|$.

Let $G = (N, T, P, I)$ be a CF grammar. A $(N \cup T \cup \{\lambda\})$-labelled tree t is a *derivation tree of G* if it satisfies the following conditions.

1. The root of t is labelled by the initial nonterminal, that is, $t(\lambda) = I$.
2. The leaves of t are labelled by terminals or by the symbol λ.
3. Let $\mu \in \text{dom}(t)$ have k immediate successors $\mu_1, \ldots, \mu_k$, $k \geq 1$. Then $t(\mu) \to t(\mu_1) \cdots t(\mu_k) \in P$.

The set of derivation trees of G is denoted $T(G)$. The derivation trees of G are in one-to-one correspondence with the derivations of G producing terminal words, and thus $L(G) = \{\text{yd}(t) \mid t \in T(G)\}$. Above, in the word $\text{yd}(t)$, we identify occurrences of the symbol λ with the empty word. The family of context-free languages is denoted $\mathcal{L}(\text{CF})$ (see [10,11] for more details on context-free grammars and languages).

The family of ET0L (extended tabled Lindenmayer systems without interaction) languages is denoted by $\mathcal{L}(\text{ET0L})$ (see [12] for more details on ET0L systems and languages).

We define asymptotic representations of functions as in [13]. Note that the definition of Ω given in other publications might differ. Let $f : \mathbb{N}_0 \to \mathbb{R}_0^+$ and $g : \mathbb{N}_0 \to \mathbb{R}_0^+$ be functions, then

- $g \in \mathcal{O}(f)$ if and only if there exist constants $c > 0$ and $n_0 \in \mathbb{N}_0$, such that $g(n) \leq c \cdot f(n)$ for all $n \geq n_0$;
- $g \in \Omega(f)$ if and only if there exist constants $c > 0$ and $n_0 \in \mathbb{N}_0$, such that $f(n) \leq c \cdot g(n)$ for all $n \geq n_0$;
- $g \in \Theta(f)$ if and only if there exist constants $c_1, c_2 > 0$ and $n_0 \in \mathbb{N}_0$, such that $c_1 \cdot f(n) \leq g(n) \leq c_2 \cdot f(n)$ for all $n \geq n_0$;

The function f is called an upper, lower or tight bound of g if $g \in \mathcal{O}(f)$, $g \in \Omega(f)$ or $g \in \Theta(f)$, respectively.

3. Synchronization functions

3.1. Synchronized context-free grammars

We define synchronized context-free grammars and languages as in [4].

Definition 1. A *synchronized context-free* (SCF) grammar is a five-tuple

$$G = (V, S, T, P, I)$$

such that $G' = (V \times (S \cup \{\lambda\}), T, P, I)$ is a context-free grammar and V, S and T are the alphabets of base nonterminals, situation symbols and terminals, respectively. The alphabet of nonterminals is $V \times (S \cup \{\lambda\})$, where elements of $V \times S$ are called *synchronized nonterminals* and elements of $V \times \{\lambda\}$ are called *non-synchronized nonterminals* which are usually denoted by their base nonterminals only. We define the morphism $h_G : (V \times (S \cup \{\lambda\}))^* \longrightarrow S^*$ by the condition $h_G((v, x)) = x$ for all $v \in V$ and $x \in S \cup \{\lambda\}$.

Each node in a derivation tree of an SCF grammar has a situation sequence associated with it.

Definition 2. Let G be an SCF grammar. For a derivation tree t of G, $t_1 = \text{inner}(t)$ and a node $\mu \in \text{leaf}(t_1)$, the *synchronizing sequence* (sync-sequence) corresponding to μ is $\text{seq}_{t_1}(\mu) = h_G(\text{path}_{t_1}(\mu))$. Also, define $\text{seq}_t = \{s \mid \text{seq}_{t_1}(\mu) = s, \mu \in \text{leaf}(t_1)$ and $s' \in \text{seq}_{t_1}(\mu'), \mu' \in \text{leaf}(t_1)$ implies $|s'| \leq |s|\}$. If this set is a singleton, we will use seq_t to refer to the element in the set.

Next we define which derivation trees contribute to the languages generated by an SCF grammar.

Definition 3. Let $G = (V, S, T, P, I)$ be an SCF grammar and $z \in \{\text{p}, \text{e}\}$. A derivation tree t of G is said to be z-acceptable if, for each $\mu, \nu \in \text{leaf}(\text{inner}(t))$, $\text{seq}_{\text{inner}(t)}(\mu) \simeq_z \text{seq}_{\text{inner}(t)}(\nu)$. The set of z-acceptable derivation trees of G is denoted by $T_z(G)$.

Notice that if t is an e- or p-acceptable derivation, then seq_t is a singleton.

Definition 4. For $z \in \{\text{p}, \text{e}\}$, the *$z$-synchronized language* of G is $L_z(G) = \text{yd}(T_z(G))$. The families of z-SCF languages, for $z \in \{\text{p}, \text{e}\}$, and SCF languages are denoted $\mathcal{L}_z(\text{SCF})$ and $\mathcal{L}(\text{SCF}) = \mathcal{L}_\text{e}(\text{SCF}) \cup \mathcal{L}_\text{p}(\text{SCF})$.

It was proven in [4] that p- and e-synchronization generate the same family of languages, i.e. $\mathcal{L}_\text{e}(\text{SCF}) = \mathcal{L}_\text{p}(\text{SCF}) = \mathcal{L}(\text{SCF})$. In [6] it was proven that SCF grammars generate the family of ET0L languages, i.e. $\mathcal{L}(\text{SCF}) = \mathcal{L}(\text{ET0L})$, and given an SCF grammar and a derivation mode one can effectively construct an equivalent ET0L system and vice versa. The length synchronization context-free grammars of [14] have the same generative capacity.

3.2. Basic definitions

In this section, we will recall the synchronization functions used to describe the total amount of synchronization, from [9], as well as define a new measure of synchronization, used to define the total length of the synchronization sequence.

Definition 5. The *synchronization count* of a derivation tree t of an SCF grammar $G = (V, S, T, P, I)$ is the number of situation symbols $\text{c-sit}_G(t)$ occurring in the tree. The *synchronization depth* of t is $\text{d-sit}_G(t) = |seq_t|$. The *synchronization count* (respectively *depth*) *of a word*, of G in z-mode, for $z \in \{\text{p}, \text{e}\}$ and $y \in \{\text{c}, \text{d}\}$, is

$$(y, z)\text{-sit}_G(w) = \begin{cases} \min\{y\text{-sit}_G(t) \mid t \in T_z(G), \text{yd}(t) = w\}, & w \in L_z(G) \\ 0, & w \in T^* \setminus L_z(G). \end{cases}$$

Definition 6. Let G be an SCF grammar. The *count*-(respectively *depth*-)*synchronization function* (y, z)-synch$_G : \mathbb{N} \to \mathbb{N}$, of G in z-mode, $z \in \{\mathrm{p}, \mathrm{e}\}$, $y \in \{\mathrm{c}, \mathrm{d}\}$, is

$$(y, z)\text{-synch}_G(n) = \max\{(y, z)\text{-sit}_G(w) \mid |w| \leq n\}.$$

Naturally there can be two SCF grammars for the same language with (asymptotically) different synchronization functions.

Definition 7. An SCF grammar G is said to be (y, z)-synchronizationally representative (or (y, z)-representative), $z \in \{\mathrm{p}, \mathrm{e}\}$, $y \in \{\mathrm{c}, \mathrm{d}\}$, for its generated language $L_z(G)$ if and only if , for all SCF grammars G' with $L_z(G) = L_z(G')$, $(y, z)\text{-synch}_G \in \mathcal{O}((y, z)\text{-synch}_{G'})$.

The synchronization measure for a given mode of an SCF language is bounded by the functions of the grammars generating the language in that mode.

Definition 8. Let $L \in \mathcal{L}(\mathrm{SCF})$ and $z \in \{\mathrm{e}, \mathrm{p}\}$, $y \in \{\mathrm{c}, \mathrm{d}\}$. Let $f : \mathbb{N} \to \mathbb{N}$ be a function. We say that the (y, z)-synchronization measure of L

- has upper bound f, denoted by $(y, z)\text{-synch}_L \in \mathcal{O}(f)$, if there exists a grammar G, such that $L_z(G) = L$ and $(y, z)\text{-synch}_G \in \mathcal{O}(f)$;
- has lower bound f, denoted by $(y, z)\text{-synch}_L \in \Omega(f)$, if for all grammars G with $L_z(G) = L$ we have $(y, z)\text{-synch}_G \in \Omega(f)$;

If $(y, z)\text{-synch}_L \in \mathcal{O}(f)$ and $(y, z)\text{-synch}_L \in \Omega(f)$, then f is called a (y, z)-synchronization representative of L, denoted by $(y, z)\text{-synch}_L \in \Theta(f)$.

Note that, for an SCF language L, $z \in \{\mathrm{e}, \mathrm{p}\}$ and $y \in \{\mathrm{c}, \mathrm{d}\}$, the existence of a representative grammar and a function f with $(y, z)\text{-synch}_L \in \Theta(f)$ does not follow from the definition.

Definition 9. Let $z \in \{\mathrm{e}, \mathrm{p}\}$, $y \in \{\mathrm{c}, \mathrm{d}\}$ and let $f : \mathbb{N} \to \mathbb{N}$ be a function. Let

$$\mathcal{L}^{(f)}_{(y,z)}(\mathrm{SCF}) = \{L \mid L \in \mathcal{L}(\mathrm{SCF}), (y, z)\text{-synch}_L \in \mathcal{O}(f)\}$$

be the language family of all SCF languages with the (y, z)-synchronization measure having upper bound f.

The following example should help to clarify the definitions.

Example 10. We define an SCF grammar $G = (V, S, T, P, I)$ by $V = \{I, A, B, C, A', B', C'\}$, $S = \{s_0, s_1\}$, $T = \{a, b, c\}$ and the following productions are, for $i, j \in \{0, 1\}$ and $X \in \{A, B, C\}$, in P.

$$I \to (A, s_i)(B, s_i)(C, s_i), \qquad (X, s_0) \to (X, s_i) \mid \lambda,$$
$$(X, s_1) \to (X', s_i)(X, s_i) \mid x, \qquad (X', s_i) \to (X', s_j)(X', s_j) \mid xx.$$

The tree t in Fig. 1 is an e-synchronized derivation tree of G with situation sequence $s_1 s_0 s_1$, d-sit$_G(t) = 3$, c-sit$_G(t) = 18$ and $\mathrm{yd}(t) = a^5 b^5 c^5$.

In equality mode, G generates $L_{\mathrm{e}}(G) = \{a^n b^n c^n \mid n \in \mathbb{N}_0\}$ and, as n is encoded in binary, we have $(\mathrm{d}, \mathrm{e})\text{-synch}_G \in \Theta(\log n)$.

The following normal form was originally defined in [9].

Definition 11. An SCF grammar $G = (V, S, T, P, I)$ is in λ-free normal form if all productions in P are either of the form $I \to \lambda$ or $(X, f) \to w$, where $X \in V$, $f \in S \cup \{\lambda\}$, $w \in ((V \setminus \{I\}) \times (S \cup \{\lambda\}) \cup T)^+$.

From the construction used to show that $\mathcal{L}_{\mathrm{e}}(\mathrm{SCF}) = \mathcal{L}_{\mathrm{p}}(\mathrm{SCF})$ in [4] it follows that an arbitrary e-synchronized grammar can be simulated by a p-synchronized grammar of (essentially) same synchronization depth, and vice versa. This property is stated below.

Lemma 12. *Let G be an* SCF *grammar and let $z, z' \in \{\mathrm{e}, \mathrm{p}\}$. Then there exists an* SCF *grammar G', such that $L_{z'}(G') = L_z(G)$ and $(\mathrm{d}, z')\text{-synch}_{G'} \in \Theta((\mathrm{d}, z)\text{-synch}_G)$.*

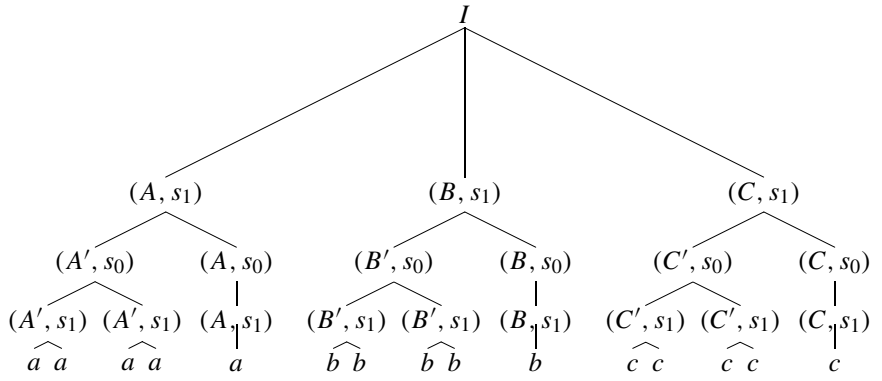

Fig. 1. e-synchronized derivation tree of G of Example 10.

It was proven in [9] that for each SCF grammar G and $z \in \{\mathrm{e}, \mathrm{p}\}$ one can effectively construct an SCF grammar G' in λ-free normal form such that $L_z(G') = L_z(G)$ and $(\mathrm{c}, z)\text{-synch}_{G'} \in \mathcal{O}((\mathrm{c}, z)\text{-synch}_G)$. From the construction of the proofs it also follows that $(\mathrm{d}, z)\text{-synch}_{G'} \in \mathcal{O}((\mathrm{d}, z)\text{-synch}_G)$. Also recall the following definition from [9]:

Definition 13. Let G be an SCF grammar, let $y \in \{\mathrm{c}, \mathrm{d}\}$, $z \in \{\mathrm{e}, \mathrm{p}\}$ and let $k \in \mathbb{N}$. The (y, z)-synchronization function of G is said to be *bounded* by k if and only if $(y, z)\text{-synch}_G(n) \leq k$ for all $n \in \mathbb{N}$. The synchronization function of G is said to be bounded if and only if there exists a constant k such that it is bounded by k. An SCF language L is called (y, z)-bounded if and only if there exists an SCF grammar G with $L_z(G) = L$ such that $(y, z)\text{-synch}_G$ is bounded. The family of all (y, z)-bounded SCF languages is denoted by $\mathcal{L}_{(y,z)}(\mathrm{bSCF})$.

Note that bounded synchronization count always implies bounded synchronization depth. The following results are immediate consequences of the proofs of the results in [9].

Lemma 14. *Let G be an* SCF *grammar, let $k \in \mathbb{N}$ and let $y \in \{\mathrm{c}, \mathrm{d}\}$, $z \in \{\mathrm{p}, \mathrm{e}\}$. If the (y, z)-synchronization function of G is bounded by k then $L_z(G) \in \mathcal{L}(\mathrm{CF})$ and a context-free grammar G' with $L(G') = L_z(G)$ can be effectively constructed.*

As every context-free language can be generated by an SCF grammar with bounded synchronization, the following is an immediate consequence.

Proposition 15. $\mathcal{L}_{(y,z)}(\mathrm{bSCF}) = \mathcal{L}(\mathrm{CF})$, *for $y \in \{\mathrm{c}, \mathrm{d}\}$, $z \in \{\mathrm{p}, \mathrm{e}\}$.*

3.3. Lower and upper bounds

In the following we establish an upper and a lower bound on depth-synchronization functions of non-context-free SCF languages, similarly to the following two bounds for count-synchronization from [9].

For count-synchronization functions the following results were proven in [9].

Lemma 16. *Given an SCF grammar G and $z \in \{\mathrm{p}, \mathrm{e}\}$, if $(\mathrm{c}, z)\text{-synch}_G$ is not bounded by a constant then $(\mathrm{c}, z)\text{-synch}_G \in \Omega(n)$.*

Lemma 17. *Let G be a synchronized context-free grammar in λ-free normal form. Then, for $z \in \{\mathrm{p}, \mathrm{e}\}$, $(\mathrm{c}, z)\text{-synch}_G \in \mathcal{O}(n^2)$.*

Theorem 18. *Let $L \in \mathcal{L}(\mathrm{SCF}) \setminus \mathcal{L}(\mathrm{CF})$ and $z \in \{\mathrm{p}, \mathrm{e}\}$. Then $(\mathrm{c}, z)\text{-synch}_L \in \Omega(n)$ and $(\mathrm{c}, z)\text{-synch}_L \in \mathcal{O}(n^2)$.*

For depth-synchronization functions we get analogous results:

Lemma 19. *Given an SCF grammar G and $z \in \{\mathrm{e}, \mathrm{p}\}$, if $(\mathrm{d}, z)\text{-synch}_G$ is not bounded by a constant then $(\mathrm{d}, z)\text{-synch}_G \in \Omega(\log n)$.*

Proof. By Lemma 16, we know that $(c, z)\text{-synch}_G \in \Omega(n)$. As the branching degree of the derivation trees is bounded by the maximal length of the right-hand sides of the productions, $(d, z)\text{-synch}_G \in \Omega(\log n)$ is an immediate consequence. □

Lemma 20. *Let G be an SCF grammar. Then, for $z \in \{e, p\}$, $(d, z)\text{-synch}_G \in \mathcal{O}(n)$.*

Proof. Let $G = (V, S, T, P, I)$, $w \in L_e(G)$, $w \neq \lambda$ and let t be an optimal, (d, e)-synchronized derivation tree of G for w, i.e. $d\text{-sit}_G(t) = (d, e)\text{-sit}_G(w)$.

As t is an optimal derivation tree, we will show next that the number of consecutive synchronized derivation steps during which no terminal symbols are produced and all paths which are branched off only generate the empty word is bounded by $(|V| \cdot |S|)! \cdot 2^{|V| \cdot |S|}$.

Let i satisfy $1 \leq i \leq |d\text{-sit}_G(t)|$ and define $f(i)$ to be the string over $V \times S$ obtained by concatenating the synchronizing nonterminals that appear in t at the ith position of the synchronizing sequence, from left to right.

Let i be some fixed number satisfying $1 \leq i \leq |d\text{-sit}_G(t)|$ and let $f(i) = A_{0,1} \cdots A_{0,n}$, where $A_{0,j} \in V \times S$ and $n > 0$. Assume that there exists l such that, for every k, $0 < k \leq l$, $f(i + k) = R_{k,0} A_{k,1} R_{k,1} \cdots R_{k,n-1} A_{k,n} R_{k,n}$ where $A_{k,j} \in V \times S$, $R_{k,j} \in (V \times S)^*$, $A_{k,j}$ is a descendant of $A_{k-1,j}$ and the yield of the subtree at each nonterminal of $R_{k,j}$ is λ, for each $0 \leq j \leq n$.

For each k, $0 < k \leq l$, let $\xi(k) = \{V \mid V \in \text{alph}(R_{k,j}), \text{ for some } 0 \leq j \leq n\}$. If there exist k_1 and k_2 such that $A_{k_1,1} = A_{k_2,1}, \ldots, A_{k_1,n} = A_{k_2,n}$ and $\xi(k_1) \subseteq \xi(k_2)$ then we can replace the subtrees at synchronization depth $i + k_1$ with those at $i + k_2$, and then the tree t were not optimal. If $l > (|V| \cdot |S|)!$, then there must exist k_1 and k_2 such that $A_{k_1,1} = A_{k_2,1}, \ldots, A_{k_1,n} = A_{k_2,n}$. If $l > 2^{|V| \cdot |S|}$, then there must exist k_1 and k_2 such that $\xi(k_1) \subseteq \xi(k_2)$. Thus, if $l > (|V| \cdot |S|)! \cdot 2^{|V| \cdot |S|}$ then there exists k_1 and k_2 such that $A_{k_1,1} = A_{k_2,1}, \ldots, A_{k_1,n} = A_{k_2,n}$ and $\xi(k_1) \subseteq \xi(k_2)$. Thus, $l \leq (|V| \cdot |S|)! \cdot 2^{|V| \cdot |S|}$.

Let m be the greatest synchronization depth at which a terminal is generated. By a similar argument, $d\text{-sit}_G(t) \leq m + (|V| \cdot |S|)! \cdot 2^{|V| \cdot |S|}$. Thus, for every $(|V| \cdot |S|)! \cdot 2^{|V| \cdot |S|}$ situation symbols in the tree there is at least one terminal symbol in the tree and hence, $(d, e)\text{-synch}_G \in \mathcal{O}(n)$. By Lemma 12 we also obtain $(d, p)\text{-synch}_G \in \mathcal{O}(n)$. □

By combining the upper and lower bounds from Lemmas 19 and 20 we obtain the following result.

Theorem 21. *Let $L \in \mathcal{L}(\text{SCF}) \setminus \mathcal{L}(\text{CF})$ and $z \in \{p, e\}$. Then $(d, z)\text{-synch}_L \in \Omega(\log n)$ and $(d, z)\text{-synch}_L \in \mathcal{O}(n)$.*

Applying a homomorphism with a bounded erasing property, as defined below, to an SCF language or intersecting it with a regular language increases the synchronization function in any mode by at most a constant factor, thus preserving the asymptotic upper bound.

Lemma 22. *Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $z \in \{e, p\}$.*

i. *Let $h : \Sigma^* \to 2^{\Gamma^*}$ be a finite substitution and $L \subseteq \Sigma^*$ such that there exists a constant $c < 1$ such that in any word $w \in L$, h erases at most $c \cdot |w|$ symbols. If G is an SCF grammar for L such that $(d, z)\text{-synch}_G \in \mathcal{O}(f)$ then $(d, z)\text{-synch}_{h(L)} \in \mathcal{O}(f)$.*

ii. *Let $L \subseteq \Sigma^*$ and assume that R is a regular language over Σ. If $(d, z)\text{-synch}_L \in \mathcal{O}(f)$ then $(d, z)\text{-synch}_{L \cap R} \in \mathcal{O}(f)$.*

Proof. i. From an arbitrary SCF grammar G for L we get an SCF grammar G' for $h(L)$ by replacing in the productions each terminal symbol b by an arbitrary word in the finite set $h(b)$. The transformation changes the synchronization depth of the grammar only by the linear factor c (it is, of course, possible that $h(L)$ may be generated by other grammars with "smaller" depth).

ii. We use the construction used in [11] to show that context-free languages are closed under intersection with regular languages. The transformation does not increase the synchronization depth of the grammar. The construction has to be modified slightly as the original construction uses Chomsky normal form. □

4. An infinite hierarchy

4.1. Languages with linear depth synchronization

In [9] the question whether or not there exists an SCF language that can only be generated by SCF grammars with quadratic count-synchronization functions remains open. We will now solve the analogous question for

depth-synchronization functions and prove that there are SCF languages that can only be generated by SCF grammars with a linear depth-synchronization function. For the remainder of this paper we fix

$$L_0 = \{w\$w \mid w \in \{0, 1, \#\}^+\}. \tag{1}$$

The next technical lemma is used to prove the main result.

Lemma 23. *For $z \in \{\mathrm{e}, \mathrm{p}\}$ and every SCF grammar G with $L_z(G) = L_0$ there exists a constant $\delta \in \mathbb{N}$ such that for any $m \geq 1$ and $0 \leq k < m$ for every word $w \in \{0, 1, \#\}^m$ there exists a derivation tree $t \in T_z(G)$ with $\mathrm{yd}(t) = w^\delta w[k]\$w^\delta w[k]$ and a subtree $t' \in \mathrm{sub}(t)$ such that $|\mathrm{yd}(t')| < \frac{1}{2}|\mathrm{yd}(t)|$ and w is a subword of $\mathrm{yd}(t')$.*

Proof. Let l be the maximum length of any right-hand side of a production of G and let $\delta = l + 2$. Let $m \geq 1$, $0 \leq k < m$ and $w \in \{0, 1, \#\}^m$ be arbitrary and let $x = w^\delta w[k]\$w^\delta w[k]$. Since $x \in L_0$, there exists a $t \in T_z(G)$ such that $x = \mathrm{yd}(t)$. Let u be a node of t of maximal distance from the root such that

$$|\mathrm{yd}(t/u)| \geq \frac{1}{2}|\mathrm{yd}(t)|. \tag{2}$$

Thus $\mathrm{yd}(t/u)$ contains entirely at least $\delta - 1$ copies of the subword w, i.e. there exist $p, q \in \mathbb{N}$, $p + q = \delta - 1$ such that $w^p w[k]\$w^q$ or $w^{\delta-1}$ is a subword of $\mathrm{yd}(t/u)$. Now $\delta - 1 = l + 1$ and by the pigeon hole principle some child u' of u contains at least one of the copies of the subword w. Also we have $|\mathrm{yd}(t/u')| < \frac{1}{2}|\mathrm{yd}(t)|$ since u was chosen to be of maximal distance from the root such that (2) holds. □

Lemma 24. *For $z \in \{\mathrm{e}, \mathrm{p}\}$, $(\mathrm{d}, z)\text{-synch}_{L_0} \in \Omega(n)$.*

Proof. Assume that $(\mathrm{d}, z)\text{-synch}_{L_0} \notin \Omega(n)$. Thus there exists an SCF grammar $G = (V, S, T, P, I)$, such that $L_0 = L_z(G)$ and $(\mathrm{d}, z)\text{-synch}_G \notin \Omega(n)$, that is

$$\forall c > 0, \forall n_0 \in \mathbb{N}, \exists n_c \geq n_0 \text{ such that } (\mathrm{d}, z)\text{-synch}_G(n_c) < c \cdot n_c. \tag{3}$$

Let δ be the constant for G obtained from Lemma 23.

There exist constants $\alpha > 0$ and $m_\alpha \geq 0$ such that for all $m \geq m_\alpha$ and $0 \leq k < m$ we have

$$|V| \cdot (|S| + 1) \cdot (|S| + 1)^{\alpha(2\delta m + 2k + 1)} < 2^{m-2}. \tag{4}$$

Let $m_0 = \max\{m_\alpha, \delta\}$. By (3) for $c = \alpha$ and a word length $n_0 = 2\delta m_0 + 2m_0 + 1$ there exists $n_c \geq n_0$ such that $(\mathrm{d}, z)\text{-synch}_G(n_c) < c \cdot n_c$ and, therefore, we have $(\mathrm{d}, z)\text{-sit}_G(n_c) < c \cdot n_c$. Without loss of generality, we can assume the constant n_c to be odd, as all words in L_0 have odd length and by Definition 6 we therefore have $(\mathrm{d}, z)\text{-synch}_G(2n) = (\mathrm{d}, z)\text{-synch}_G(2n - 1)$, for $n \geq 1$.

Furthermore there are constants $m \geq m_0$ and $0 \leq k < m$ such that $n_c = 2\delta m + 2k + 1$ and if m is assumed to be maximal, then m and k are uniquely determined by n_c and δ (as $m \geq m_0 \geq \delta$). Among the words of length n_c are the 2^{m-2} words of the form $w^\delta w[k]\$w^\delta w[k]$ where $w \in \#\{0, 1\}^{m-2}\#$ is arbitrary. By Lemma 23, we obtain for each such w a derivation tree t_w with $\mathrm{yd}(t_w) = w^\delta w[k]\$w^\delta w[k]$ such that t_w has a subtree t'_w, such that w is a subword of $\mathrm{yd}(t'_w)$ and $\mathrm{yd}(t'_w) < \frac{1}{2}|\mathrm{yd}(t_w)|$. By (3) $|\mathrm{seq}(t_w)| < c \cdot n_c$. Now (4) implies that there are words $w_1, w_2 \in \#\{0, 1\}^{m-2}\#$, $w_1 \neq w_2$, such that

$$\mathrm{seq}(t'_{w_1}) = \mathrm{seq}(t'_{w_2}) \text{ and } \mathrm{root}(t'_{w_1}) = \mathrm{root}(t'_{w_2}). \tag{5}$$

Let t_{12} be the tree obtained from t_{w_1} by replacing t'_{w_1} with t'_{w_2}. By (5), t_{12} is a synchronized derivation tree of G. We write

$$\mathrm{yd}(t_{w_i}) = w_{i,1,1}w_{i,2,1} \cdots w_{i,\delta,1}w_i[k]\$w_{i,1,2}w_{i,2,2} \cdots w_{i,\delta,2}w_i[k].$$

Here each word $w_{i,j,k}$, $1 \leq j \leq \delta$, $1 \leq k \leq 2$, is equal to w_i and we use the additional subindices only to differentiate between subword occurrences. If the marker \$ is not in $\mathrm{yd}(t'_{w_2})$, then clearly $\mathrm{yd}(t_{12}) \notin L_0$ since we replace something only on one side of the marker. Now knowing that \$ is in $\mathrm{yd}(t'_{w_2})$, it has to be also in $\mathrm{yd}(t'_{w_1})$, since otherwise the yield of t_{12} would have two symbols \$.

The word $\mathrm{yd}(t'_{w_2})$ contains at least one complete occurrence of w_2, and without loss of generality we can assume that it occurs before the middle marker \$. The other case is symmetric (and slightly easier). As shown in Fig. 2, since

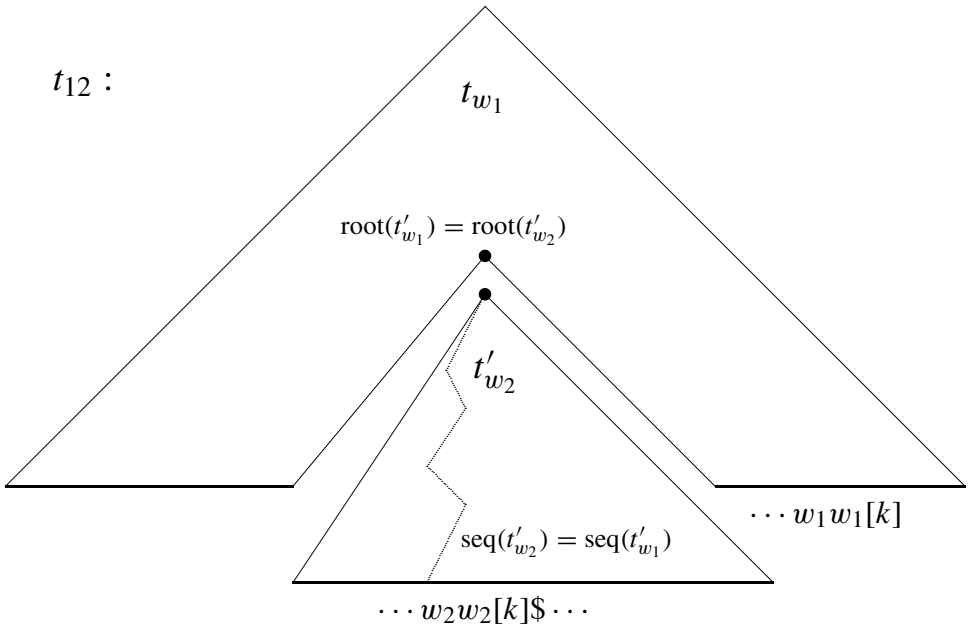

Fig. 2. The derivation tree t_{12}, obtained from t_{w_1} by replacing the subtree t'_{w_1} by the subtree t'_{w_2}, which has the same situation sequence.

$\mathrm{yd}(t'_{w_2})$ contains \$, it follows that $\mathrm{yd}(t'_{w_2})$ contains the subword occurrence $w_{2,\delta,1}$ (i.e., the last complete occurrence of w_2 before the middle marker).

If a part of $w_{1,\delta,2}$ (i.e., a part of the last complete occurrence of w_1) is in $\mathrm{yd}(t'_{w_1})$ then, since $|\mathrm{yd}(t'_{w_i})| < \frac{1}{2}|\mathrm{yd}(t_{w_i})|$, $i = 1, 2$, it is easy to see that in $\mathrm{yd}(t_{12})$ the prefix before the marker \$ must be longer than the suffix after \$.

Thus, we know that the subword occurrence $w_{1,\delta,2}$ remains in the tree t_{12} after replacing t'_{w_1} with t'_{w_2}. Hence in the yield of t_{12} the prefix before \$ ends with $w_{2,\delta,1}w_2[k] = w_2w_2[k]$ and the the suffix after \$ ends with $w_{1,\delta,2}w_1[k] = w_1w_1[k]$. Since $w_1 \neq w_2$, this means that $\mathrm{yd}(t_{12}) \notin L_0$.

As the above holds for any odd $n_c \geq 2\delta m_0 + 2m_0 + 1$, we have seen that any SCF grammar with less than linear synchronization depth cannot generate L_0. □

By Lemma 20, the following main theorem follows.

Theorem 25. *For* $z \in \{\mathrm{e}, \mathrm{p}\}$, $(\mathrm{d}, z)\text{-synch}_{L_0} \in \Theta(n)$.

The language used in Theorem 25 is over a four letter alphabet. Below we show that there is a language over a binary alphabet that has synchronization depth $\Theta(n)$.

Corollary 26. *There exists* $L \subseteq \{0, 1\}^*$ *such that* $(\mathrm{d}, z)\text{-synch}_L \in \Theta(n)$, *for* $z \in \{\mathrm{e}, \mathrm{p}\}$.

Proof. By Theorem 25 there exists a language L_0 over a four letter alphabet Σ with $(\mathrm{d}, z)\text{-synch}_{L_0} \in \Theta(n)$. Let $g : \Sigma \to \{0, 1\}^2$ be an injective mapping. By Lemma 22(i), $(\mathrm{d}, z)\text{-synch}_{g(L_0)} \in \mathcal{O}(n)$. We need to show that $(\mathrm{d}, z)\text{-synch}_{g(L_0)} \in \Omega(n)$.

We set $\Gamma = \{0, 1\}^2 \times \{1, 2\}$ and define the finite substitution $h : \{0, 1\}^* \to 2^{\Gamma^*}$ by setting for $i \in \{0, 1\}$, $h(i) = \{(j_1, j_2, k) \mid j_1, j_2 \in \{0, 1\}, j_k = i, 1 \leq k \leq 2\}$. Let $R \subseteq \Gamma^*$ be the regular language $R = \{(j_1, j_2, 1)(j_1, j_2, 2) \mid j_1, j_2 \in \{0, 1\}\}^*$.

Finally we define the morphism $s : \Gamma^* \to \Sigma^*$ by setting $s((j_1, j_2, 1)) = g^{-1}(j_1, j_2)$ and $s((j_1, j_2, 2)) = \lambda$, where $j_1, j_2 \in \{0, 1\}$.

Intuitively, s maps elements of Γ where the last component is 1, to the element of Σ represented by the first two components (if the element exists). We observe that

$$s(h(g(L_0)) \cap R) = L_0. \tag{6}$$

For the sake of contradiction assume that $(\mathrm{d}, z)\text{-synch}_{g(L_0)}$ is not in $\Omega(n)$. Thus there exists an SCF grammar G for $g(L_0)$ such that $(\mathrm{d}, z)\text{-synch}_G \notin \Omega(n)$. Now (6) and Lemma 22 imply that $(\mathrm{d}, z)\text{-synch}_{L_0} \in \mathcal{O}((\mathrm{d}, z)\text{-synch}_G))$ and, thus, $(\mathrm{d}, z)\text{-synch}_{L_0} \notin \Omega(n)$, which is a contradiction to Theorem 25. □

4.2. Depth synchronization in between logarithmic and linear

In the following we show that there are languages with depth-synchronization functions strictly in between logarithmic and linear, and that these languages form a strict infinite hierarchy. For this purpose we construct, for each natural $k \geq 1$, a language

$$L'_k = \{w\$w\$a^{f_k(|w|)} \mid w \in \{0, 1, \#\}^+\}, \tag{7}$$

where $f_k \in \Theta(n^{k+1})$, and we prove that, for $z \in \{\mathrm{e}, \mathrm{p}\}$, $(\mathrm{d}, z)\text{-synch}_{L_k} \in \Theta(n^{\frac{1}{k+1}})$.

First we show that, if such languages exist, then their lower bounds follow from Lemma 24.

Lemma 27. *Let $k \in \mathbb{N}$, $k > 1$, let $f : \mathbb{N} \to \mathbb{N}$, $f \in \Theta(n^k)$ and $L' \in \mathcal{L}(\mathrm{SCF})$ with $L' = \{w\$w\$a^{f(|w|)} \mid w \in \{0, 1, \#\}^+\}$. Then, for $z \in \{\mathrm{e}, \mathrm{p}\}$, $(\mathrm{d}, z)\text{-synch}_{L'} \in \Omega(n^{\frac{1}{k}})$.*

Proof. If $f \in \Theta(n^k)$, there exist $c_1, c_2 > 0$, $n_1 \in \mathbb{N}$, such that for all $n > n_1$, $c_1 \cdot n^k \leq f(n) \leq c_2 \cdot n^k$. Let $z \in \{\mathrm{e}, \mathrm{p}\}$ and assume that $(\mathrm{d}, z)\text{-synch}_{L'} \notin \Omega(n^{\frac{1}{k}})$. Thus there exists an SCF grammar G', such that $L' = L_z(G')$ and $(\mathrm{d}, z)\text{-synch}_{G'} \notin \Omega(n^{\frac{1}{k}})$, that is

$$\forall c > 0, \forall n_0 \in \mathbb{N}, \exists n_c \geq n_0 \text{ such that } (\mathrm{d}, z)\text{-synch}_{G'}(n_c) < c \cdot n_c^{\frac{1}{k}}. \tag{8}$$

If n_c is the length of some word in L', there exists a $n'_c \in \mathbb{N}$ with $n_c = 2n'_c + 2 + f(n'_c)$. Let G'' be the SCF grammar obtained from G' by replacing each occurrence of a in the right-hand side of a production with λ. Then for all $c, c_2 > 0$ and for all $n_0 \in \mathbb{N}$ there exists $n'_c \geq n_0$ such that

$$(\mathrm{d}, z)\text{-synch}_{G''}(2n'_c + 2) < c \cdot (2n'_c + 2 + c_2 \cdot (n'_c)^k)^{\frac{1}{k}} < c \cdot c_2 \cdot (2n'_c + 2). \tag{9}$$

Thus, $(\mathrm{d}, z)\text{-synch}_{G''} \notin \Omega(n)$, which is a contradiction as $L_z(G'') = L_0$, as defined in (1), and by Lemma 24, $(\mathrm{d}, z)\text{-synch}_{L_0} \in \Omega(n)$. □

Below we construct, for each $k \geq 1$, an SCF grammar G_k that is subsequently proven to generate a language L'_k as defined in (7).

Definition 28. For $k \in \mathbb{N}$, let $G_k = (V_k, S, T, P_k, I)$ be a synchronized context-free grammar with $V_k = \{I, A, A', B, B_1, \ldots, B_k\}$, $S = \{s_0, s_1, s_\#\}$, $T = \{0, 1, \#, \$, a\}$ and the following set of productions P_k where $i, j \in \{0, 1, \#\}$ and $1 \leq l \leq k$:

$$\begin{aligned}
I &\to (A, s_i)(A, s_i)(B_k, s_i), & (A, s_i) &\to i(A, s_j) \mid i\$,\\
(A', s_i) &\to a(A', s_j) \mid a, & (B_l, s_i) &\to (B_{l-1}, s_j)(A', s_j)(B_l, s_j) \mid a,\\
(B_1, s_i) &\to (B, s_j)(A', s_j) \mid a, & (B, s_i) &\to (B, s_j)(A', s_j) \mid a.
\end{aligned}$$

For $k \in \mathbb{N}$, let $L_k = L_{\mathrm{e}}(G_k)$ and let, for $X \in V_k$, $n \in \mathbb{N}$,

$$L_k^X(n) = \{\mathrm{yd}(t) \mid t' \in T_{\mathrm{e}}(G_k), t \in \mathrm{sub}(t'), \exists f \in S \cup \{\lambda\}, \mathrm{root}(t) = (X, f), |\mathrm{seq}_t| = n\}$$

be the set of words derivable from (X, f), for some $f \in S \cup \{\lambda\}$ with an e-synchronized tree and a situation sequence of length n, and let $f_k^X(n) = \{|w| \mid w \in L_k^X(n)\}$. Furthermore we define, for $k, n \in \mathbb{N}$, $f_k(n) = f_k^I(n)$.

We now prove some auxiliary results about the languages L_k, which lead up to the result that L_k is of the form L'_k, as defined in (7). First observe that, for $k \in \mathbb{N}$, we have

$$L_k \subseteq \{w\$w\$a^m \mid w \in \{0, 1, \#\}^+, m \in \mathbb{N}\}. \tag{10}$$

Also, for $k, k' \in \mathbb{N}$,

$$f_k^X(n) = f_{k'}^X(n) \text{ if } X \in \{A, A', B, B_l \mid l \leq \min\{k, k'\}\}. \tag{11}$$

We now show that all words generated by G_k in equality mode with situation sequences of equal length have the same length.

Lemma 29. *For $k \in \mathbb{N}$, $X \in V_k$ and $n \in \mathbb{N}$, we have $|f_k^X(n)| = 1$.*

Proof. For $X \in \{A, A'\}$ and $k, n \in \mathbb{N}$ the result trivially holds. Furthermore, for $k \in \mathbb{N}$, $|f_k^B(1)| = 1$ and by induction $|f_k^B(n+1)| = 1$, for $n \in \mathbb{N}$, if $|f_k^B(n)| = 1$, as $(B, s) \Rightarrow (B, s')(A', s')$ and $|f_k^{A'}(n)| = 1$, for $s, s' \in S$. Similarly, for $k \in \mathbb{N}$, $l \leq k$, $|f_k^{B_l}(1)| = 1$ and by induction, for $l \leq k$, $|f_k^{B_l}(n+1)| = 1$, for $n \in \mathbb{N}$, if $|f_k^{B_l}(n)| = 1$, as $(B_l, s) \Rightarrow (B_{l-1}, s')(A', s')(B_l, s')$ and $|f_k^{A'}(n)| = 1$, for $s, s' \in S$. Then, also for $k, n \in \mathbb{N}$, $|f_k^I(n)| = 1$, as $I \Rightarrow (A, s)(A, s)(B_k, s)$ and $|f_k^A(n)| = |f_k^{B_k}(n)| = 1$, for $s \in S$. □

For the remainder of this paper we write $f_k^X(n) = m$ instead of $f_k^X(n) = \{m\}$.

Lemma 30. *For $k \geq 1$ and $w_1\$w_1\$a^{m_1}, w_2\$w_2\$a^{m_2} \in L_k$, we have $|w_1| = |w_2|$ if and only if $m_1 = m_2$, and $|w_1| > |w_2|$ if and only if $m_1 > m_2$.*

Proof. For words $w_1, w_2 \in \{0, 1, \#\}^+$, we have $|w_1| = |w_2|$ if and only if $\text{d-sit}_{G_k}(w_1\$w_1\$a^{m_1}) = \text{d-sit}_{G_k}(w_2\$w_2\$a^{m_2})$, as, whenever a terminal 0, 1 or # is produced, so is exactly one synchronized nonterminal and vice versa. As the length of the situation sequence is in one-to-one correspondence with the word length by Lemma 29, $\text{d-sit}_{G_k}(w_1\$w_1\$a^{m_1}) = \text{d-sit}_{G_k}(w_2\$w_2\$a^{m_2})$ if and only if $m_1 = m_2$.

Similar to the above argument, for $w_1, w_2 \in \{0, 1, \#\}^+$, we have $|w_1| > |w_2|$ if and only if $\text{d-sit}_{G_k}(w_1\$w_1\$a^{m_1}) > \text{d-sit}_{G_k}(w_2\$w_2\$a^{m_2})$. We have $f_k^{A'}(n+1) > f_k^{A'}(n)$, for all $n \in N$ and from this it follows that, for all $n \in \mathbb{N}$, $f_k^B(n+1) > f_k^B(n)$ and $f_k^{B_k}(n+1) > f_k^{B_k}(n)$, which is the case if and only if $m_1 > m_2$. □

By the previous two lemmata we know that f_k for $k \geq 1$ is a strictly monotonically increasing function. The next proof uses the structural properties of the derivation trees, as shown in Fig. 3.

Lemma 31. *For $k \geq 1$, $f_k \in \Theta(n^{k+1})$.*

Proof. We know from Lemma 30, that the length of $w\$w\$w' \in L_k$ is in one-to-one correspondence to the length of w and that a longer w implies a longer w' and, therefore a longer word of L_k.

For $X \in \{A, A'\}$ and $k, n \in \mathbb{N}$ we obviously get $f_k^X(n) = n$, and $f_k^B(1) = 1$ and $f_k^B(n+1) = f_k^B(n) + n$ and, hence, $f_k^B \in \Theta(n^2)$.

We now show that, for $k \geq 1$, $f_k \in \Theta(n^{k+1})$. Observe first that, for $k, n \geq 1$, $f_k(n+1) = 2 \cdot f_k^A(n) + f_k^{B_k}(n)$. It is easy to verify, that $f_1 \in \Theta(n^2)$, as

$$f_1^{B_1}(n) = f_1^B(n-1) + f_1^{A'}(n-1) \in \Theta(n^2). \tag{12}$$

Now assume that $f_k^{B_k} \in \Theta(n^{k+1})$. Thus, there exist constants $c_1, c_2 > 0$ and $n_0 \in \mathbb{N}$, such that for all $n \geq n_0$ $c_1 \cdot n^{k+1} \leq f_k^{B_k}(n) \leq c_2 \cdot n^{k+1}$.

For $n \in \mathbb{N}$, we have, as depicted in Fig. 3,

$$\begin{aligned} f_{k+1}^{B_{k+1}}(n) &= f_{k+1}^{B_k}(n-1) + n + f_{k+1}^{B_{k+1}}(n-1) \overset{(11)}{=} f_k^{B_k}(n-1) + n + f_{k+1}^{B_{k+1}}(n-1) \\ &= \sum_{i=1}^{n-1} (f_k^{B_k}(n-i) + i) = \sum_{i=1}^{n-1} (f_k^{B_k}(i) + i). \end{aligned}$$

Thus, there exists $c_3 \geq 0$, such that for all $n \geq n_0$,

$$f_{k+1}^{B_{k+1}}(n) \geq \sum_{i=1}^{n-1} c_1 \cdot i^{k+1} \geq c_3 \cdot n^{k+2}, \tag{13}$$

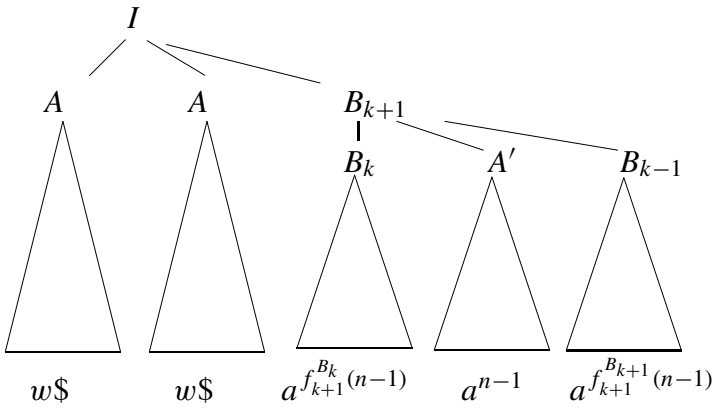

Fig. 3. Structure of derivation trees of G_{k+1} with synchronization depth n, and thus $|w| = n$.

and there exists $c_4 \geq 0$, such that for all $n \geq n_0$,

$$f_{k+1}^{B_{k+1}}(n) \leq \sum_{i=1}^{n-1}(1 + c_2) \cdot i^{k+1} \leq c_4 \cdot n^{k+2}. \tag{14}$$

Thus $f_k \in \Theta(n^{k+1})$, for $k \in \mathbb{N}$. □

We can now prove our second main theorem.

Theorem 32. *For $k \geq 1$ and $z \in \{\mathrm{e}, \mathrm{p}\}$, (d, z)-synch$_{L_k} \in \Theta(n^{\frac{1}{k+1}})$.*

Proof. By Lemma 27, for $k \geq 1$ and $z \in \{\mathrm{e}, \mathrm{p}\}$, (d, z)-synch$_{L_k} \in \Omega(n^{\frac{1}{k+1}})$, as $f_k \in \Theta(n^{k+1})$ by Lemma 31 and $L_k = \{w\$w\$a^{f_k(|w|)} \mid w \in \{0, 1, \#\}^+\}$.

Furthermore (d, z)-synch$_{(G_k)} \in \mathcal{O}(n^{\frac{1}{k+1}})$, as each word $w\$w\$a^{f(|w|)}$ is mapped to $|w|$ and $f \in \Theta(n^{k+1})$. Thus, (d, z)-synch$_{L_k} \in \Theta(n^{\frac{1}{k+1}})$. □

The languages used in Theorem 32 are over a five letter alphabet. Similar to Corollary 26, one can show the following result.

Corollary 33. *Let $k \geq 1$ and $z \in \{\mathrm{e}, \mathrm{p}\}$. There exists a language $L \subseteq \{0, 1\}^*$ such that (d, z)-synch$_L \in \Theta(n^{\frac{1}{k}})$.*

By combining Proposition 15, Lemmas 19 and 20, Theorems 25 and 32 we obtain the following strict infinite hierarchy of SCF languages.

Theorem 34. *For $k \geq 1$, $z \in \{\mathrm{e}, \mathrm{p}\}$, we have*

$$\mathcal{L}(\mathrm{CF}) = \mathcal{L}^{(1)}_{(\mathrm{d},z)}(\mathrm{SCF}) \subset \mathcal{L}^{(\log n)}_{(\mathrm{d},z)}(\mathrm{SCF}) \subset \cdots \subset \mathcal{L}^{(n^{\frac{1}{k+1}})}_{(\mathrm{d},z)}(\mathrm{SCF}) \subset \mathcal{L}^{(n^{\frac{1}{k}})}_{(\mathrm{d},z)}(\mathrm{SCF}) \subset \cdots \subset \mathcal{L}^{(n)}_{(\mathrm{d},z)}(\mathrm{SCF}) = \mathcal{L}(\mathrm{SCF}).$$

5. Conclusion

We have demonstrated that the depth-synchronization measure gives rise to a strict infinite hierarchy of language families within the family of SCF languages and is, thus, a valid way of measuring the descriptional complexity of SCF and ET0L languages.

The problem whether or not there exist languages which require synchronization depth in between logarithmic and linear other than $n^{\frac{1}{k}}$ for some $k \geq 1$ remains open. We have not been able, so far, to even find an SCF grammar with a depth-synchronization function in $\Theta(n^{\frac{2}{3}})$, which seems to be the simplest function outside of the hierarchy presented in this paper.

While we have shown the upper bound of the depth-synchronization measures for SCF languages to be tight, the analogous question still remains open for count-synchronization.

It also remains open where restricted classes of SCF languages, like the counter SCF languages defined in [5], lie within the hierarchy.

It remains to be shown that every SCF grammar has a representative grammar for count and depth synchronization. We conjecture that this is the case.

Acknowledgements

The authors' research was supported in part by grants from the Natural Sciences and Engineering Research Council of Canada. The first author's research was supported in part by the SHARCNET research chairs program.

References

[1] J. Dassow, G. Păun, Regulated Rewriting in Formal Language Theory, in: EATCS Monographs in Theoretical Computer Science, vol. 18, Springer-Verlag, Berlin, 1989.
[2] E. Csuhaj-Varjú, J. Dassow, J. Kelemen, G. Păun, Grammar Systems: A Grammatical Approach to Distribution and Cooperation, in: Topics in Computer Mathematics, vol. 5, Gordon and Breach Science Publishers, Yverdon, 1994.
[3] A. Aho, Indexed grammars — An extension of context-free grammars, Journal of the ACM 15 (1968) 647–671.
[4] H. Jürgensen, K. Salomaa, Block-synchronization context-free grammars, in: Z. Du, I. Ko (Eds.), Advances in Algorithms, Languages, and Complexity, Kluwer Academic Publishers, The Netherlands, 1997, pp. 111–137.
[5] H. Bordihn, M. Holzer, On the computational complexity of synchronized context-free languages, Journal of Universal Computer Science 8 (2) (2002) 119–140.
[6] I. McQuillan, The generative capacity of block-synchronized context-free grammars, Theoretical Computer Science 337 (2005) 119–133.
[7] I. McQuillan, Descriptional complexity of block-synchronized context-free grammars, Journal of Automata, Languages and Combinatorics 9 (2004) 317–332.
[8] J. Hromkovič, J. Karhumäki, B. Rovan, A. Slobodová, On the power of synchronization in parallel computations, Discrete and Applied Mathematics 32 (1991) 155–182.
[9] F. Biegler, Synchronization functions of synchronized context-free grammars and languages, in: C. Mereghetti, B. Palano, G. Pighizzini, D. Wotschke (Eds.), 7th International Workshop on Descriptional Complexity of Formal Systems, Como, Italy, 2005, pp. 236–244.
[10] J. Hopcroft, J. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley, Reading, MA, 1979.
[11] A. Salomaa, Formal Languages, Academic Press, New York, 1973.
[12] G. Rozenberg, A. Salomaa, The Mathematical Theory of L Systems, Academic Press, Inc., New York, 1980.
[13] T. Cormen, C. Leiserson, R. Rivest, Introduction to Algorithms, MIT Press, Cambridge, Mass., 1989.
[14] M. Madhu, K. Krithivasan, Length synchronization context-free grammars, Journal of Automata Languages and Combinatorics 9 (2004) 457–464.