Journal of Universal Computer Science, vol. 8, no. 2 (2002), 119-140
submitted: 15/9/01, accepted: 31/1/02, appeared: 28/2/02

On the Computational Complexity of Synchronized Context-Free Languages[1]

Henning Bordihn
(Institut für Informatik, Universität Potsdam,
Agust-Bebel-Str. 89, Haus 4, D-14482 Postdam, Germany
henning@cs.uni-potsdam.de)

Markus Holzer
(Institut für Informatik, Technische Universität München,
Arcisstraße 21, D-80290 München, Germany
holzer@informatik.tu-muenchen.de)

Abstract: We introduce counter synchronized context-free grammars and investigate their generative power. It turns out that the family of counter synchronized context-free languages is a proper superset of the family of context-free languages and is strictly contained in the family of synchronized context-free languages. Moreover, we establish the space and time complexity of the fixed membership, the general membership, and the non-emptiness problem for synchronized and counter synchronized context-free languages and solve the mentioned complexity questions in terms of completeness results for complexity classes. In this way we present new complete problems for LOG(**CF**), **NP**, and **PSpace**. It is worth to mention that the main theorem on the **PSpace**-completeness of the general membership problem of synchronized context-free grammars relies on a remarkable normal form for these grammars, namely for every synchronized context-free grammar one can effectively construct and equivalent grammar of same type without non-synchronizing nonterminals, except the axiom.

Key Words: Formal languages, synchronized grammars, decision problems, computational complexity.

Category: F.1.3, F.2.2, F.4.2, F.4.3

1 Introduction

One of the most thoroughly investigated language classes is the family of context-free languages. However, in the applications of formal language theory, in particular to natural and programming languages, there are a lot of aspects where context-free grammars and languages turn out to be not sufficient. For example, the language of all words of the form ww, where w is an arbitrary word over some alphabet, is important both in linguistics and in the syntax of programming languages. For an extensive discussion of that item, e.g., see [Dassow and Păun 1989].

Since context-sensitive grammars, the next level in the Chomsky hierarchy, are such powerful that they become, in fact, intractable, a series of context-free grammars with regulated rewriting has been introduced. The aim is to cover non-context-free languages which are, in principle, needed in the applications, thereby

[1] C. S. Calude, K. Salomaa, S. Yu (eds.). *Advances and Trends in Automata and Formal Languages. A Collection of Papers in Honour of the 60th Birthday of Helmut Jürgensen.*

maintaining as many as possible of the "nice" properties of the context-free languages. In grammars with regulated rewriting, certain derivations are selected to be correct whereas the other derivations are ruled out and not contributing to the language generated.

In [Jürgensen and Salomaa 1997], a new type of such extensions of context-free grammars was introduced, namely block-synchronized context-free grammars, where the mechanism ruling out some derivations is defined in a quite natural way. In a derivation, independent sub-derivations, i.e., derivations starting off from different nonterminals in a sentential form, can communicate by means of so-called synchronizing symbols which may be attached to the nonterminals. A derivation is considered to be correct if and only if the sequences of synchronizing symbols produced along any two paths are in the prefix relation, i.e., one is a prefix of the other, or, in another synchronization mode, are identical. The first restriction is referred to as prefix-mode (p-mode) of derivation and the latter as equality-mode (e-mode) of derivation.

The idea of synchronizing independent branches of a computation is inherited from several different automata models, e.g., see [Hromkovič, Karhumäki, Rovan, and Slobodová 1991; Hromkovič, Rovan, and Slobodová 1994; Salomaa 1994]. In [Jürgensen and Salomaa 1997], this concept of synchronization is extended by allowing the grammar to recursively begin, within a synchronized derivation, a sub-derivation that has to be synchronized in a similar manner, leading to the notion of *block*-synchronized context-free grammars.

In the present paper, we restrict ourselves to the case of synchronized context-free grammars, where such nesting feature is not taken into consideration. In the study of synchronized context-free grammars the question arises whether the generative power of these grammars depend on the number of synchronizing symbols. It is obvious that a synchronized grammar with an empty set of synchronizing symbols can only generate context-free languages. On the other hand two synchronizing symbols suffice to generate all synchronized context-free languages, because the synchronizing symbols of a general synchronized grammar can be encoded by strings of two synchronizing symbols. Thus, the restricted variant of *counter synchronized context-free grammars* is introduced, where, except from a possible end marker, only one synchronizing symbol can be attached to the nonterminals such that different synchronizing sequences are distinguished only by their lengths, except from a possible appearance or nonappearance of the end marker. This concept of counter synchronized context-free grammars appears to be simple, natural and, moreover, has analogues both in automata theory and in the theory of formal languages. In particular, context-free indexed counter grammars [Aho 1968] use only one index symbol and a possible end marker in order to control the derivations of the underlying context-free grammar.

The paper is organized as follows: In the next section we introduce the necessary notions. Then in Section 3 we show that counter synchronized context-free grammars can be simulated by (a restricted variant) of context-free indexed counter grammars. As a conclusion, we prove that they are strictly less powerful as their general counterparts, the synchronized context-free grammars. Moreover, besides presenting a few normal forms, we show that prefix-synchronization is exactly as powerful as equality-synchronization also in case of counter synchronized context-free grammars. In the next two sections, which are the main part of the paper, we investigate the complexity status of the fixed and the general membership as well as the non-emptiness problem for both synchronized and counter synchronized context-free grammars. Finally, we summarize our results and state some open problems.

2 Definitions

The reader is assumed to be familiar with the basic notions of formal language theory, in particular with context-free grammars, and complexity theory, as contained, e.g., in [Hopcroft and Ullman 1979] and [Balcázar, Díaz, and Gabarró 1988], respectively. This section aims to fix the notation used throughout this paper. Moreover, the definitions of synchronized and counter synchronized context-free grammars are given. We mainly follow [Jürgensen and Salomaa 1997].

First, we establish the following general conventions: Let $\subseteq$ denote inclusion, whereas $\subset$ denotes strict inclusion, and $|M|$ is the cardinality of set M. Let Σ be some alphabet, i.e., a finite non-empty set. The set of all words over Σ is denoted by Σ^* and $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$, where λ is the empty word. A word $w \in \Sigma^*$ is *prefix* of $v \in \Sigma^*$, $v \leq_{\mathrm{p}} w$, if there is a word $u \in \Sigma^*$ such that $w = vu$. Two words w_1 and w_2 are said to be in the *prefix relation*, in symbols $w_1 \simeq w_2$, if one word is prefix of the other.

A *context-free grammar* is a quadruple $G = (N, T, I, P)$, where N and T are two disjoint alphabets of nonterminal and terminal symbols, respectively, $I \in N$ is the axiom, and P is a finite set of context-free productions of the form $A \to \alpha$ with $A \in N$ and $\alpha \in (N \cup T)^*$. A word $\alpha \in (N \cup T)^*$ directly derives β, $\alpha \Rightarrow \beta$, if and only if $\alpha = \alpha_1 A \alpha_2$, $\beta = \alpha_1 \gamma \alpha_2$ with $\alpha_1, \alpha_2 \in (N \cup T)^*$, and $A \to \gamma$ in P. The language generated by G is defined by

$$L(G) = \{\, w \in T^* \mid I \overset{*}{\Rightarrow} w \,\},$$

where $\overset{*}{\Rightarrow}$ is the reflexive and transitive closure of the relation $\Rightarrow$. A language is said to be context-free if it is generated by some context-free grammar and the family of all context-free languages is denoted by $\mathcal{L}(\mathbf{CF})$.

The derivations of a context-free grammar can be represented as trees where the nodes are labelled by symbols from $(N \cup T \cup \{\lambda\})$. Such a *derivation tree* t of a context-free grammar $G = (N, T, I, P)$ satisfies the following three conditions: (1) The root of t is labelled by the axiom I, (2) the leaves of t are labelled by terminal symbols or by λ, and (3) a node labelled by nonterminal A has exactly k immediate successors which are labelled, in their natural order from left to right, by B_1, B_2, $\ldots$, B_k if and only if $(A \to B_1 B_2 \ldots B_k) \in P$ with $B_i \in (N \cup T)$, for $1 \leq i \leq k$.

We consider the concept of a tree and notions such as root, leaf, path etc. to be well-known and mention only the following. Let t be a tree the nodes of which are labelled by symbols of some set S. For some node μ, let $\mathrm{path}_t(\mu)$ denote the sequence of symbols of S occurring on the path from the root of t to the node μ. Nodes of a tree t that are not leaves are said to be *inner nodes* of t, and the tree which is obtained by cutting off all the leaves is the *inner tree* of t, denoted by $\mathrm{inner}(t)$. The *yield* of a labelled tree t, denoted by $\mathrm{yd}(t)$, is the string which results when the labels of the leaves of t are concatenated from left to right (in the natural order). Here, occurrences of λ are identified with the empty word, i.e., they are simply omitted. Then, for a context-free grammar G, we have

$$L(G) = \{ \operatorname{yd}(t) \mid t \text{ is a derivation tree of } G \ \}.$$

In the following definition we introduce synchronized and counter synchronized context-free grammars.

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

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

such that $G' = (V \times (S \cup \{\lambda\}), T, I, P)$ forms a context-free grammar. Here V is the alphabet of base nonterminals and S is the alphabet of synchronizing symbols. Elements of $V \times S$ are called synchronizing nonterminals, whereas nonterminals of the form (A, λ) are called non-synchronizing nonterminals.

A synchronized context-free grammar $G = (V, S, T, I, P)$ is a *counter synchronized context-free grammar* if and only if $S = \{f, \#\}$ and P satisfies the condition that $(A, \#) \to \alpha$ in P implies $\alpha \in T^*$, for all $A \in V$. The synchronizing symbol f is called counter symbol and $\#$ is called end marker.

In fact, if G is a counter synchronized context-free grammar, only the leaves of the inner tree of its derivation trees may be labelled by synchronizing nonterminals of the form $(A, \#)$, whereas to the other nodes only non-synchronizing nonterminals or nonterminals with the counter symbol can be assigned. But note that a leaf of the inner tree may be labelled by an arbitrary nonterminal of the counter synchronized context-free grammar.

In order to simplify the notation, we usually identify non-synchronizing nonterminals with the corresponding base nonterminal, i.e., we simply write A instead of (A, λ). Furthermore, we define the morphism

$$h_g : (V \times (S \cup \{\lambda\}))^* \to S^*$$

by the condition $h_g((A, x)) = x$ for each $A \in V$ and $x \in S \cup \{\lambda\}$.

Before we give the definitions of prefix- and equality-synchronized derivations of (counter) synchronized context-free grammars, we need to define the notions of the synchronizing sequence corresponding to a leaf of the inner tree of a derivation tree and then of p-acceptable and e-acceptable derivations. Since counter synchronized context-free grammars are particular synchronized context-free grammars, we give the formal definitions only for the general case, but they apply to counter synchronized context-free grammars as well.

Definition 2. Let $G = (V, S, T, I, P)$ be a synchronized context-free grammar and t be some derivation tree of the underlying context-free grammar $G' = (V \times (S \cup \{\lambda\}), T, I, P)$; in the forthcoming, when referring to a derivation tree of a synchronized grammar we mean the corresponding derivation tree of the underlying context-free grammar. Denote $t_1 = \operatorname{inner}(t)$ and let μ be some leaf of t_1.

1. The *synchronizing sequence* with respect to t_1 corresponding to μ is the word

$$\operatorname{seq}_{t_1}(\mu) = h_g(\operatorname{path}_{t_1}(\mu)).$$

2. The derivation tree t is referred to as p-*acceptable* if $\mathrm{seq}_{t_1}(\mu) \simeq \mathrm{seq}_{t_1}(\nu)$ for any couple of leaves (μ, ν) of the inner tree t_1.
3. The derivation tree t is referred to as e-*acceptable* if $\mathrm{seq}_{t_1}(\mu) = \mathrm{seq}_{t_1}(\nu)$ for any couple of leaves (μ, ν) of the inner tree t_1.

A derivation is said to be z-*acceptable*, for $z \in \{\mathrm{p}, \mathrm{e}\}$, if the corresponding derivation tree is so.

Observe, that in case of *counter* synchronized context-free grammars the synchronization sequence $\mathrm{seq}_{t_1}(\mu) \in f^* \cup f^*\#$ for any leaf μ of the inner tree t_1 of an arbitrary derivation tree t induced by the grammar under consideration. Although one may wish to define counter synchronized context-free grammars without end marker, i.e., only by the restriction $|S| = 1$, this would be less sensible. Clearly, under such a definition any derivation would be p-acceptable, hence those systems would describe exactly the family of context-free languages $\mathcal{L}(\mathbf{CF})$. Furthermore, it is an easy exercise to prove that allowing end markers, as in the definition given above, does not alter the generative power of counter synchronized context-free grammars considering e-acceptable derivations, only. This fact is also seen from the proof of Theorem 8 given in the next section.

Definition 3. Let $z \in \{\mathrm{p}, \mathrm{e}\}$. The language generated by the synchronized context-free grammar G working in z-mode of derivation is the set

$$L_z(G) = \{\, \mathrm{yd}(t) \mid t \text{ is a } z\text{-acceptable derivation tree of } G \,\}.$$

A language L is a z-synchronized context-free language (z-counter synchronized context-free language,respectively) if there exists a synchronized context-free grammar (a counter synchronized context-free grammar, respectively) G such that $L = L_z(G)$.

The family of z-synchronized context-free languages, for $z \in \{\mathrm{p}, \mathrm{e}\}$, is denoted by $\mathcal{L}_z(\mathbf{SCF})$. Analogously, the family of z-counter synchronized context-free languages is denoted by $\mathcal{L}_z(\mathbf{cSCF})$.

In the remainder of this section we introduce the necessary notions concerning the theory of computational complexity. We consider the following well-known sequence of containments: $\mathbf{NL} \subseteq \mathrm{LOG}(\mathbf{CF}) \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSpace}$.

Here $\mathbf{NL}$ is the set of problems accepted by nondeterministic logarithmic space bounded Turing machines, and $\mathrm{LOG}(\mathbf{CF})$ is the set of languages accepted by nondeterministic auxiliary pushdown automata with a logarithmic space bounded work tape in polynomial time. Moreover, $\mathbf{P}$ ($\mathbf{NP}$, respectively) is the set of problems accepted by deterministic (nondeterministic, respectively) polynomially time bounded Turing machines, and $\mathbf{PSpace}$ is $\bigcup_k \mathbf{DSpace}(n^k)$. Completeness and hardness are always meant with respect to deterministic many-one log-space reducibilities. If A log-space many-one reduces to B we simply write $A \leq_m^{\log} B$. A problem A is said to be log-space many-one equivalent or as hard as B, if and only if A reduces to B and B reduces to A.

In this paper we investigate the fixed and general membership problem, as well as the non-emptiness problem, for counter synchronized and synchronized context-free languages. The *fixed membership problem* for (counter) synchronized context-free languages is defined as follows:

– Fix a (counter) synchronized context-free grammar G working in z-mode of derivation, with $z \in \{\mathrm{p}, \mathrm{e}\}$. For a given word w, is $w \in L_z(G)$?

A natural generalization is the *general membership problem* which is defined as follows:

– Given a (counter) synchronized context-free grammar G working in z-mode of derivation, with $z \in \{\mathrm{p}, \mathrm{e}\}$, and a word w, i.e., an encoding $\langle G, w\rangle$, is $w \in L_z(G)$?

Finally, the *non-emptiness problem* is defined as:

– Given a (counter) synchronized context-free grammar G working in z-mode of derivation, with $z \in \{\mathrm{p}, \mathrm{e}\}$, is $L_z(G) \neq \emptyset$?

The general membership and non-emptiness problems have synchronized grammars as inputs. Therefore we need an appropriate coding function $\langle\cdot\rangle$ which maps a grammar G (and a string w) into a word $\langle G\rangle$ (word $\langle G, w\rangle$) over a fixed alphabet Σ. We do not go into the details of $\langle\cdot\rangle$, but assume it fulfills certain standard properties; for instance, that the coding of terminal and nonterminal symbols of the grammar is of logarithmic length.

3 Normal Forms and Inclusion Structure

In this section we prove normal forms for synchronized context-free grammars and investigate the inclusion structure on synchronized and counter synchronized context-free language families working in z-mode of derivation, for $z \in \{\mathrm{p}, \mathrm{e}\}$.

3.1 Normal Forms

We consider three normal forms, namely the standard normal form, the binary normal form, and the normal form without non-synchronizing nonterminals (except the axiom). All constructions in this subsection are effective and don't increase the size of the grammars to much, i.e., the constructed grammars remain polynomial in the size of the originally given grammar.

Let us start our investigations with the easiest normal form, the standard normal form. A (counter) synchronized context-free grammar $G = (V, S, T, I, P)$ is in *standard normal form*, if the right-hand sides of the productions are either a string of nonterminal symbols or a single terminal symbol. More precisely it is required that $V = V_1 \cup V_2$ with $V_1 \cap V_1 = \emptyset$, and

$$P \subseteq (N_1 \times (N_1 \cup N_2)^*) \cup (N_2 \times (T \cup \{\lambda\})),$$

where $N_i = V_i \times (S \cup \{\lambda\})$, for $1 \leq i \leq 2$. The following lemma shows that for both counter synchronized and synchronized context-free grammars such a normal form exists.

Lemma 4. *Let $G = (V, S, T, I, P)$ be a (counter) synchronized context-free grammar. Then there exists a (counter) synchronized context-free grammar G' in standard normal form with $L_z(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.*

Proof. We distinguish two cases:

1. If G is an arbitrary synchronized context-free grammar, set $V_1 = V$ and $V_2 = \{ X_a \mid a \in T \cup \{\lambda\} \}$, and replace, for all $a \in T \cup \{\lambda\}$, any occurrence of a in the rules of P by $X_a \in V_2$. The construction of P' is completed by adding the rules $X_a \to a$, for $a \in T \cup \{\lambda\}$.
2. If G is a counter synchronized context-free grammar, define the set $V_1 = V \cup \{ A_\# \mid A \in V \}$ (the union being disjoint) and $V_2 = \{ X_a \mid a \in T \cup \{\lambda\} \}$. Perform the construction as in the general case, but additionally replace all occurrences of $(A, \#)$ in the productions by $A_\#$, for all $A \in V$, and then replace all productions (which have been obtained by the replacements described above) of the form $A_\# \to X_{a_1} X_{a_2} \dots X_{a_m}$ with the production $A_\# \to (X_{a_1}, \#)(X_{a_2}, \#) \dots (X_{a_m}, \#)$, and add the rules $(X_a, \#) \to a$, for $a \in T \cup \{\lambda\}$. This completes the description of the production set P'.

Define the synchronized context-free grammar $G' = (V_1 \cup V_2, S, T, I, P')$. Obviously, grammar G' is in standard normal form and it is straightforward to prove that $\mathcal{L}_z(G') = \mathcal{L}_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$. □

Next we consider the binary normal form, which is defined as follows: A synchronized context-free grammar G is in *binary normal form* if grammar G is in standard normal form and every right-hand side of a nonterminating production is of length at most two. As in the previous lemma, such a normal form exists for both counter synchronized and synchronized context-free grammars.

Lemma 5. *Let $G = (V, S, T, I, P)$ be a (counter) synchronized context-free grammar. Then there exists a (counter) synchronized context-free grammar G' in binary normal form, such that $L_z(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.*

Proof. Let G be in standard normal form. It remains to split up the right-hand side of every nonterminating production which is of length greater than or equal to two. Let N denote the set $V \times (S \cup \{\lambda\})$. Then, every nonterminating rule $(A, s) \to Y_1 Y_2 \dots Y_m$ in P with $m \geq 3$, $s \in S \cup \{\lambda\}$, and $Y_1 Y_2 \dots Y_m \in N^+$ is replaced by

$$\begin{aligned}
(A, s) &\to Y_1 (Z_{Y_2 \dots Y_m}, \lambda) \\
(Z_{Y_2 \dots Y_m}, \lambda) &\to Y_2 (Z_{Y_3 \dots Y_m}, \lambda) \\
&\vdots \\
(Z_{Y_{m-1} Y_m}, \lambda) &\to Y_{m-1} Y_m,
\end{aligned}$$

where the Z's are new base nonterminals. Obviously, the constructed (counter) synchronized grammar G' is in binary normal form and is equivalent to the original grammar G, i.e., $L_z(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$. □

In the remainder of this subsection we show that non-synchronizing nonterminals are not essential for synchronized context-free grammars in general. A synchronized grammar $G = (V, S, T, I, P)$ is without non-synchronizing nonterminals if

$$P \subseteq (\{I\} \times (N \cup T)^*) \cup (N \times (N \cup T)^*),$$

where $N = V \times S$. Loosely speaking, there are no nonterminals of the form (A, λ), except the axiom.

Lemma 6. *Let $G = (V, S, T, I, P)$ be a synchronized context-free grammar. Then there exists a synchronized context-free grammar G' without non-synchronizing nonterminals, except the axiom, such that $L_z(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.*

Proof. Let G be in standard normal form. We eliminate non-synchronizing nonterminals, i.e., nonterminals of the form (A, λ), by introducing a new synchronizing symbol \$, which serves as a placeholder for λ. In addition the productions must be enabled to produce an appropriate number of \$ symbols along the branches in order to meat the acceptance condition of synchronized context-free grammars. In detail, the construction for a synchronized context-free grammar $\overline{G} = (\overline{V}, S \cup \{\$\}, T, I', \overline{P})$ looks as follows: Let

$$\overline{V} = V \cup \{\, A_s \mid A \in V \text{ and } s \in S \cup \{\$\}\,\} \\ \cup \{\, a' \mid a \in T \cup \{\lambda\}\,\} \cup \{\, X_a \mid a \in T \cup \{\lambda\}\,\} \cup \{I'\}$$

(the unions being disjoint) be the new base nonterminal set. In order to establish $\overline{P}$, replace any nonterminating production of P of the form

$$(A, s) \to Y_1 Y_2 \ldots Y_m,$$

where $s \in S \cup \{\lambda\}$ and $Y_1 Y_2 \ldots Y_m \in (V \times (S \cup \{\lambda\}))^+$, with the production

$$(A, t) \to Z_1 Z_2 \ldots Z_m,$$

where

$$t = \begin{cases} s \text{ if } s \in S \\ \$ \text{ otherwise} \end{cases} \quad \text{and} \quad Z_i = \begin{cases} (B_r, \$) \text{ if } Y_i = (B, r) \text{ with } r \in S \\ (B_\$, \$) \text{ if } Y_i = (B, \lambda). \end{cases}$$

Moreover, for all $A \in V$ and $s \in S \cup \{\$\}$ add the productions $(A_s, \$) \to (A_s, \$)$ and $(A_s, \$) \to (A, s)$. Finally, any terminating production of P of the form $(A, s) \to a$, for $s \in S \cup \{\lambda\}$ and $a \in T \cup \{\lambda\}$, is replaced with the production $(A, s) \to (a', \$)$ if $s \in S$ and $(A, \$) \to (a', \$)$ if $s = \lambda$. To prolong and terminate the derivation accordingly, the rules $(a', \$) \to (a', \$)$, $(a', \$) \to (X_a, \$)$, and $(X_a, \$) \to a$ for all $a \in T \cup \{\lambda\}$ are added. The use of the X_a base nonterminals ensures that the grammar is in standard normal form. Since the original axiom I is a non-synchronizing nonterminal its occurrence on the left-hand sides of productions was replaced by $(I, \$)$ according to the above given construction. Thus, we introduce a new axiom I' together with the rule $I' \to (I, \$)$ in order to start the derivation process. This completes the description of the synchronized context-free grammar $\overline{G}$.

We illustrate the idea behind the given construction by the following example. Let $(A, r) \to (B, \lambda)(C, s)$ be a rule in P with $r, s \in S$. Then by construction, we obtain the production $(A, r) \to (B_\$, \$)(C_s, \$)$. Applying this rule introduces the \$ in the synchronizing sequence along the two branches, which can be filled by further \$ symbols using the prolongation rules $(B_\$, \$) \to (B_\$, \$)$ and $(C_s, \$) \to (C_s, \$)$. The insertion of \$ symbols in both branches can be stopped by the productions $(B_\$, \$) \to (B, \$)$ and $(C_s, \$) \to (C, s)$. Therefore, it is possible to insert an arbitrary number of \$ symbols in the synchronizing sequences induced by a tree $\overline{t}$ of $\overline{G}$. This enables us to show that for every tree $\overline{t}$ of $\overline{G}$ which is z-acceptable, for $z \in \{\mathrm{p}, \mathrm{e}\}$, there is a corresponding tree t in G and *vice versa*. Thus, $L_z(\overline{G}) = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$, holds and the stated result follows. □

Combining the normal form conversions of Lemmata 5 and 6 exactly in this order, we obtain the following corollary.

Corollary 7. *Let $G = (V, S, T, I, P)$ be a synchronized context-free grammar. Then there exists a synchronized context-free grammar G' in binary normal form without non-synchronizing nonterminals, except the axiom, such that $L_z(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.* □

Finally, we want to mention that all normal form conversions of this subsection can be done by a deterministic Turing machine within logarithmic space.

3.2 Inclusion Relations

At first we recall that the family of synchronized context-free languages is the same in p- and e-mode of derivation. Then we alter this proof accordingly, such that it works also in case of counter synchronized context-free languages, thus proving also equivalence of p- and e-derivation mode. Finally, it is shown that the family of counter synchronized context-free languages is a proper subset of the family of synchronized context-free languages. To this end, we show how to simulate a counter synchronized context-free grammar by some sort of indexed grammar. Thus, the limitation to use only one synchronizing and an end marker symbol is a real restriction in the generative power of synchronized context-free grammars.

Theorem 8. $\mathcal{L}_{\mathrm{p}}(\mathbf{cSCF}) = \mathcal{L}_{\mathrm{e}}(\mathbf{cSCF})$ *and* $\mathcal{L}_{\mathrm{p}}(\mathbf{SCF}) = \mathcal{L}_{\mathrm{e}}(\mathbf{SCF})$.

Proof. The proof of $\mathcal{L}_{\mathrm{p}}(\mathbf{SCF}) = \mathcal{L}_{\mathrm{e}}(\mathbf{SCF})$ has been given in [Jürgensen and Salomaa 1997]. We only sketch the ideas, adjusting them to the purposes of the present paper by some very simple modifications, in particular concerning the size of the constructed grammars.

1. $\mathcal{L}_{\mathrm{e}}(\mathbf{SCF}) \subseteq \mathcal{L}_{\mathrm{p}}(\mathbf{SCF})$ (confer [Jürgensen and Salomaa 1997, Lemma 4.1]). Given a synchronized context-free grammar $G = (V, S, T, I, P)$, a new synchronizing symbol \$ is added to S and, in the rules, every terminal symbol a is replaced with a corresponding new nonterminal $(X_a, \$)$ that can be rewritten only by the production $(X_a, \$) \to a$. Now, p-acceptable derivations of the grammar constructed this way simulate exactly the e-acceptable derivations of G, since, for any two words $w_1, w_2 \in S^*$, we have $w_1\$ \simeq w_2\$$ if and only if $w_1 = w_2$.
2. $\mathcal{L}_{\mathrm{p}}(\mathbf{SCF}) \subseteq \mathcal{L}_{\mathrm{e}}(\mathbf{SCF})$ (confer [Jürgensen and Salomaa 1997, Lemma 4.3]). Let $G = (V, S, T, I, P)$ be a synchronized context-free grammar working in p-mode of derivation and let N denote the set $(V \times (S \cup \{\lambda\})$ of all nonterminals. By Lemma 4 we may assume that G is in standard normal form. First, for each symbol $X \in (V \cup T)$, a primed version X' is added to the alphabet of base nonterminals, and the base nonterminal of the axiom is replaced with its primed counterpart. Then, to each nonterminating production $(A, s) \to Y_1Y_2 \ldots Y_m \in P$ with $Y_1Y_2 \ldots Y_m \in N^+$, all productions of the form $(A', s) \to Z_1Z_2 \ldots Z_m$ are associated, where $Z_i = Y_i$ holds except that exactly one base nonterminal appearing on the right-hand side is replaced with its primed version. Thus, in every nonterminal sentential form, exactly

one (nondeterministically selected) symbol of N is present such that its base nonterminal is primed.
Now, each terminating production of the form $(X_a, s) \to a$, for $a \in T \cup \{\lambda\}$, is replaced with the two productions

$$(X'_a, s) \to (a', \$) \quad \text{and} \quad (X_a, s) \to a'.$$

Finally, for all $a \in T \cup \{\lambda\}$, $s \in S$, and $r \in S \cup \{\$\}$, the productions $a' \to (a', r)$, $(a', s) \to (a', r)$, and $(a', \$) \to a$ are added.
The idea is to guess (by selecting the nonterminals with primed base nonterminals) the longest path in a derivation tree t and to attach the new synchronizing symbol \$ as suffix to the synchronizing sequence corresponding to that path. Other paths of t can now be prolonged such that the resulting tree is e-acceptable if and only if t is p-acceptable. For more details, the reader may consult [Jürgensen and Salomaa 1997].

In both constructions above, a new synchronizing symbol \$ is introduced which is not present in the synchronizing alphabets of the given grammars. In order to prove the equality for counter synchronized context-free grammars, a few further modifications are needed such that we can re-use the end marker of the given grammars for the purpose of \$, keeping in mind that the same end marker may or may not occur as least synchronizing symbol in valid synchronizing sequences.

3. $\mathcal{L}_\mathrm{p}(\mathbf{cSCF}) \subseteq \mathcal{L}_\mathrm{e}(\mathbf{cSCF})$. Let $G = (V, \{f, \#\}, T, I, P)$ be a counter synchronized context-free grammar in standard normal form. We construct an equivalent counter synchronized context-free grammar $\overline{G} = (\overline{V}, \{f, \#\}, T, I, \overline{P})$ such that $L_\mathrm{e}(\overline{G}) = L_\mathrm{p}(G)$.
For all $X_a \in V_2$, add the productions $X_a \to (X_a, f)$, $X_a \to (X_a, \#)$, $(X_a, f) \to (X_a, f)$, and $(X_a, f) \to (X_a, \#)$.
Consider a p-acceptable derivation tree of G and let $t_1 = \mathrm{inner}(t)$. If there is no leaf of t_1 which is labelled by a nonterminal of the form $(A, \#)$, i.e., by a nonterminal with the end marker, then t is p-acceptable. Clearly, there is a corresponding derivation tree of $\overline{G}$ which is e-acceptable. Now, let μ be a leaf of t_1 labelled by some nonterminal with the end marker. If there is another leaf μ' of t_1 labelled by a nonterminal with the end marker, then $\mathrm{seq}_{t_1}(\mu) = \mathrm{seq}_{t_1}(\mu')$ has to hold. Therefore, such paths do not need any further treatment. If there is a leaf ν of t_1 with $\mathrm{seq}_{t_1}(\nu) = f^k$, the path from the root to ν can be prolonged with nodes labelled by nonterminals with f or $\#$ such that there is a derivation tree $\overline{t}$ of $\overline{G}$ with a corresponding path from the root to a leaf $\overline{\nu}$ of the inner tree t_2 of $\overline{t}$ such that $\mathrm{seq}_{t_2}(\overline{\nu}) = f^n\#$ and $n \geq k$. Clearly, tree $\overline{t}$ is e-acceptable if and only if, for all such leaves of the inner tree, n can be chosen such that $f^k \leq_\mathrm{p} \mathrm{seq}_{t_1}(\mu)$. Hence, $L_\mathrm{e}(\overline{G}) = L_\mathrm{p}(G)$.
4. $\mathcal{L}_\mathrm{e}(\mathbf{cSCF}) \subseteq \mathcal{L}_\mathrm{p}(\mathbf{cSCF})$. For any counter synchronized context-free grammar $G = (V, \{f, \#\}, T, I, P)$ in standard normal form, we construct an equivalent counter synchronized context-free grammar $\overline{G} = (\overline{V}, \{f, \#\}, T, I, \overline{P})$ such that $L_\mathrm{p}(\overline{G}) = L_\mathrm{e}(G)$.
Set $\overline{V} = V \cup V' \cup V'' \cup T'$, where the unions are disjoint, and we define $V' = \{\, A' \mid A \in V \,\}$, $V'' = \{\, A'' \mid A \in V \,\}$, and $T' = \{\, a' \mid a \in T \cup \{\lambda\} \,\}$.

In order to establish $\overline{P}$, replace any nonterminating production of P of the form

$$(A,s) \to Y_1Y_2\ldots Y_m,$$

where $s \in \{f,\lambda\}$ and $Y_1Y_2\ldots Y_m \in (V \times \{f,\#,\lambda\})^+$, with the production

$$(A,s) \to Z_1Z_2\ldots Z_m,$$

where

$$Z_i = \begin{cases} (B',r) & \text{if } Y_i = (B,r) \text{ with } r \in \{f,\lambda\} \\ (B'',f) & \text{if } Y_i = (B,\#). \end{cases}$$

Moreover, for all $B \in V$, add the productions

$$(B',f) \to (B,f), \quad (B',\lambda) \to (B,\lambda), \quad \text{and} \quad (B'',f) \to (B,\#).$$

Finally, any terminating production of P of the form $(X_a,s) \to a$ with $s \in \{f,\lambda\}$ and $X_a \in V_2$, is replaced with the production

$$(X_a,s) \to (a',\#),$$

and the terminating rules

$$(a',\#) \to a$$

are added for all $a \in T \cup \{\lambda\}$, whereas terminating productions of P which are of the form $(A,\#) \to w$ with $w \in T \cup \{\lambda\}$ are kept in $\overline{P}$.

To any derivation tree of G corresponds a unique derivation tree $\overline{t}$ of $\overline{G}$ such that the following holds. Let μ be a leaf of $t_1 = \text{inner}(t)$ and $\overline{\mu}$ be the corresponding leaf of $t_2 = \text{inner}(\overline{t})$. If $\text{seq}_{t_1}(\mu) = f^k$, then $\text{seq}_{t_2}(\overline{\mu}) = f^{2k}\#$, and if $\text{seq}_{t_1}(\mu) = f^k\#$, then $\text{seq}_{t_2}(\overline{\mu}) = f^{2k+1}\#$. Now, it is an easy exercise to verify that $L_{\text{p}}(\overline{G}) = L_{\text{e}}(G)$. $\square$

In each of the four constructions given in the proofs above, the equivalent (counter) synchronized context-free grammars are effectively constructed and, moreover, the size of the resulting grammar is polynomial in the size of the given grammar.

For the next lemma we need the notion of indexed grammars, which was introduced in [Aho 1968]. A *context-free indexed grammar* is a five-tuple $G = (N,T,I,P,S)$, where N, T, and I are the finite pairwise disjoint alphabets of nonterminals, terminals, and index symbols, respectively, $S \in N$ is the start symbol, and P is a finite set of productions of the form

$$A \to \alpha \quad \text{or} \quad Af \to \alpha$$

with $A \in N$, $f \in I$, and $\alpha \in (NI^* \cup T)^*$. An indexed grammar $G = (N,T,I,P,S)$ is an *indexed counter grammar*, if and only if $I = \{f,\#\}$ and the productions in P are of one of the forms

1. $S \to A\#$ or $S \to \lambda$, where S does not appear in any other production in P,
2. $A \to \alpha$ or $Af \to \alpha$ with $A \in N$, and $\alpha \in (Nf^* \cup T)^*$, or
3. $A\# \to \alpha$ with $A \in N$ and $\alpha \in (Nf^*\# \cup T)^*$.

Observe, that if G is an indexed counter grammar, the index words in the sentential forms are of the form $f^i\#$, for some $i \geq 0$. Finally, the indexed grammar G is called *restricted*, if $N = N_1 \cup N_2$ with $N_1 \cap N_2 = \emptyset$, and P contains productions of two forms:

1. $A \to \alpha$ with $A \in N_1$ and $\alpha \in (N_1 I^* \cup N_2 I^* \cup T)^*$, or
2. $Af \to \alpha$ with $A \in N_2$, $f \in I$, and $\alpha \in (N_2 I^* \cup T)^*$.

The derivation relation for indexed grammars is defined as follows: Let word $\alpha = u_1 A_1 \gamma_1 u_2 A_2 \gamma_2 \dots u_n B_n \gamma_n u_{n+1}$ with $u_i \in T^*$ for $1 \leq i \leq n+1$, $A_j \in N$, and $\gamma_j \in I^*$ for $1 \leq j \leq n$ with $n \geq 0$, be an element of $(NI^* \cup T)^*$, and let $\delta \in I^*$, we set

$$\alpha : \delta = u_1 A_1 \gamma_1 \delta u_2 A_2 \gamma_2 \delta \dots u_n B_n \gamma_n \delta u_{n+1}.$$

A sentential form $\alpha \in (NI^* \cup T)^*$ directly derives β, $\alpha \Rightarrow \beta$, if and only if $\alpha = \alpha_1 A f \delta \alpha_2$ and $\beta = \alpha_1(\gamma : \delta)\alpha_2$ with $\alpha_1, \alpha_2 \in (NI^* \cup T)^*$ and $Af \to \gamma$ in P with $f \in I \cup \{\lambda\}$.

The language generated by an indexed (counter) grammar $G = (N, T, I, P, S)$ is the set $L(G) = \{\, w \in T^* \mid S \overset{*}{\Rightarrow} w \,\}$, where $\overset{*}{\Rightarrow}$ is the reflexive transitive closure of the relation $\Rightarrow$. A language L is called an indexed (counter) language if and only if $L = L(G)$ for some indexed (counter) grammar G. Now we are ready to state the following lemma.

Lemma 9. *Let $G = (V, S, T, I, P)$ be a counter synchronized context-free grammar. Then there exists a context-free restricted indexed counter grammar G', such that $L(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.*

Proof. According to Theorem 8, it is sufficient to give the proof for $z = \mathrm{p}$. Let $G = (V, S, T, I, P)$ be a counter synchronized context-free grammar working in p-mode of derivation. Due to the construction used in the proof for $\mathcal{L}_{\mathrm{e}}(\mathbf{cSCF}) \subseteq \mathcal{L}_{\mathrm{p}}(\mathbf{cSCF})$, we can assume without loss of generality that, for any p-acceptable derivation tree t of G, we have $\mathrm{seq}_{t_1}(\mu) = f^k\#$, for $k \geq 0$, for all leaves μ of $t_1 = \mathrm{inner}(t)$. Moreover, we can assume that G is in standard normal form.

We construct an equivalent context-free restricted indexed counter grammar $G' = (N, T, J, I, P')$, where $N = N_1 \cup N_2$, as follows. Set

$$N_1 = \{I', I''\},$$

where I' and I'' are new nonterminals,

$$N_2 = (V \times (S \cup \{\lambda\})) \cup \{\, A' \mid A \in V \,\} \cup \{ A'' \mid A \in V \,\},$$

and let P' is determined as follows.

First, introduce the productions $I' \to I''\#$, $I'' \to I''f$, and $I'' \to I$. Next, any production

$$(A, s) \to Y_1 Y_2 \dots Y_m$$

in P with $Y_1 Y_2 \dots Y_m \in (V \times (S \cup \{\lambda\}))^+$ is replaced with

$$(A, s) \to Z_1 Z_2 \dots Z_m,$$

where

$$Z_i = \begin{cases} Y_i & \text{if } Y_i \text{ is non-synchronizing} \\ B' & \text{if } Y_i = (B, f) \\ B'' & \text{if } Y_i = (B, \#). \end{cases}$$

Moreover, add the rules $B'f \to (B, f)$ and $B''\# \to (B, \#)$ as well as all terminating productions from P.

With help of the rules replacing I' and I'', in a beginning phase a synchronizing sequence is guessed, which is then inherited by all nonterminals introduced in the next phase. Then, the simulation starts where an index symbol is (locally) erased whenever a synchronizing symbol is introduced in a derivation according to G, and termination is allowed at any time during the simulation. This proves $L(G') = L_{\mathrm{p}}(G)$. □

Finally, we conclude that the family of counter synchronized context-free languages is a strict superset of the family of context-free languages and a proper subset of the family of synchronized context-free languages.

Theorem 10. $\mathcal{L}(\mathbf{CF}) \subset \mathcal{L}_z(\mathbf{cSCF}) \subset \mathcal{L}_z(\mathbf{SCF})$ *for* $z \in \{\mathrm{p}, \mathrm{e}\}$.

Proof. The inclusions are obvious. For the strictness of first inclusion consider the non-context-free language

$$L = \{\, a^{2^n} \mid n \geq 1 \,\},$$

which is generated by the counter synchronized context-free grammar $G = (V, \{f, \#\}, T, I, P)$ with $V = \{I, A\}$, $T = \{a\}$, and the following set of productions:

$$\begin{aligned} I &\to (A, f) \\ (A, f) &\to (A, f)(A, f) \mid (A, \#) \\ (A, \#) &\to a, \end{aligned}$$

in either p- or e-mode of derivation.

The strictness of the second inclusion is seen as follows: Let

$$L = \{\, u\$u\$u^R \mid u \in \{a, b\}^* \,\}.$$

Language L is generated in either e- or p-mode of derivation by the synchronized context-free grammar $G = (V, S, T, I, P)$ with $V = \{I, A, B\}$, $S = \{f, g, \#\}$, $T = \{a, b, \$\}$, and the following set of productions:

$$\begin{aligned} I &\to (A, \lambda)(B, \lambda) & & \\ (A, \lambda) &\to (A, f) \mid (A, g) \mid (A, \#) & \qquad (B, \lambda) &\to (B, f) \mid (B, g) \mid (B, \#) \\ (A, f) &\to a(A, \lambda) & (B, f) &\to a(B, \lambda)a \\ (A, g) &\to b(A, \lambda) & (B, g) &\to b(B, \lambda)b \\ (A, \#) &\to \$ & (B, \#) &\to \$. \end{aligned}$$

In [Duske, Middendorf, and Parchmann 1992] it was shown that L is not an indexed counter language. Since by Lemma 9 every counter synchronized language is also a restricted indexed counter language, we conclude that L is not a counter synchronized language, too. Thus, the claim follows. □

4 Fixed Membership

We consider the fixed membership problem for counter synchronized context-free and synchronized context-free languages. It turns out that in the former case we obtain a LOG(**CF**)-complete problem, while in the latter case it becomes intractable, i.e., **NP**-complete. This shows that restricting the alphabet of synchronizing symbols to be singleton (except the end marker #) essential reduces the fixed membership complexity for synchronized context-free languages.

Theorem 11. *The fixed membership problem for counter synchronized context-free languages working in* p- *or* e-*mode of derivation is* LOG(**CF**)*-complete.*

Proof. Since $\mathcal{L}(\mathbf{CF}) \subseteq \mathcal{L}_z(\mathbf{cSCF})$, for $z \in \{\mathrm{p}, \mathrm{e}\}$, it follows that the fixed membership problem for counter synchronized context-free languages working in p- or e-mode of derivation is LOG(**CF**)-hard. Thus it remains to show that the family of languages under consideration is contained in LOG(**CF**). In Lemma 9 it was shown that for every counter synchronized context-free grammar G there exists a context-free restricted indexed counter grammar G', such that $L(G') = L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$. Observe, that without loss of generality we may restrict ourselves to leftmost derivations in indexed grammars [Aho 1968]. Next we give an algorithm on a nondeterministic auxiliary pushdown automaton which solves the membership problem for context-free (restricted) indexed counter grammars. To this end, let $G' = (N, T, I, P, S)$ be a context-free (restricted) indexed counter grammar, whose productions are in normal form, i.e., the production set contains only productions of the form

$$\begin{aligned} A &\to BC \mid a \\ A &\to Bf \\ Af &\to B \\ A\# &\to B\# \end{aligned}$$

where $A, B \in N$, $a \in T \cup \{\lambda\}$, and # denotes the index bottom symbol (see [Aho 1968]).

A nondeterministic auxiliary pushdown automaton M simulates a leftmost derivation (as in the context-free case) in a top-down manner on the pushdown. We assume that nonterminal A with index string $f^k\#$, for some k, is stored as $Af^k\#$ in the pushdown, where A is on top of the pushdown. In the initial configuration nonterminal S is stored in the pushdown. Then M starts the simulation by guessing a productions. Productions of the form $A \to a$, $A \to Bf$, $Af \to B$, and $A\# \to B\#$ can be directly applied to the sentential form stored in the pushdown; observe, that simulating $A \to a$ results in popping A and removing the index string associated with nonterminal A from the pushdown. A problem arises whenever a production of the form $A \to BC$ is guessed and should be applied, because the index string of A must be passed to B and C simultaneously. The machine overcomes this situation as follows: Symbol A is popped and the number of consecutive f's on the pushdown is counted with a binary counter implemented on the auxiliary work-tape. Then it is no problem for M to reconstruct the original index string of A for both nonterminals B and C, and to push the word $Bf^k\#Cf^k\#$ if the index string of A was exactly $f^k\#$. The automaton M accepts the input if and only if the whole input was read and the pushdown's bottom symbol is popped.

In [Duske, Middendorf, and Parchmann 1992] it was shown that for every indexed counter grammar G', there exists a constant c, such that

$$\mathrm{maxind}(w) \leq c \cdot \max\{1, |w|\}$$

for all words w that belong to $L(G')$, where

$$\mathrm{maxind}(w) := \min\{\, \mathrm{maxind}(t_w) \mid t_w \text{ is a derivation tree of } w \,\}$$

and

$$\mathrm{maxind}(t_w) := \max\{\, k \mid Af^k\# \text{ is a label of a node in } t_w \,\}.$$

Therefore, M's work-tape is logarithmically space bounded by the above given upper bound on maxind, and the polynomial time bound immediately follows form the top-down simulation. Therefore, the stated claim on the LOG(**CF**)-completeness of the fixed membership problem on counter synchronized context-free languages follows. □

In the remainder of this section we prove that for synchronized context-free languages the membership problem becomes intractable. We want to mention that the below given proof is similar to the proof of the **NP**-completeness of the fixed membership problem for ET0L (extended tabled 0L) languages, given in [van Leeuwen 1975].

Theorem 12. *The fixed membership problem for synchronized context-free languages working in* p- *or* e-*mode of derivation is* **NP**-*complete.*

Proof. Since the family of synchronized context-free languages is a subset of the **NP**-complete family of context-free indexed languages it immediately follows that **NP** is also an upper bound for the fixed membership problem for the family of languages under consideration. It remains to prove **NP**-hardness. To this end we reduce the **NP**-complete satisfiability problem for Boolean formulas in conjunctive normal form (SAT) to the fixed membership problem for synchronized context-free languages.

Let $\varphi = C_1 \wedge \cdots \wedge C_m$ be an instance of SAT consisting of m clauses and n variables. Each clause C_i, for $1 \leq i \leq m$ is a disjunction of a literal x_j or its negation $\bar{x}_j$, for $1 \leq j \leq n$. A clause C_i, with $1 \leq i \leq m$, is encoded by a word $\mathrm{code}(C_i)$ of length n in the following way: The jth letter, with $1 \leq j \leq n$, of $\mathrm{code}(C_i)$ is (i) 0, if variable x_j is not contained in the clause C_i, (ii) 1, if literal x_j is contained in C_i, and (iii) 2, if literal $\bar{x}_j$ is contained in C_i. The coding of clauses naturally extends to the coding of Boolean formulas, by defining the coding of $\varphi = C_1 \wedge \cdots \wedge C_m$ as

$$\mathrm{code}(C_1)\#\mathrm{code}(C_2)\# \ldots \#\mathrm{code}(C_m)\#.$$

Now the idea is to construct a synchronized context-free grammar which generates all satisfiable Boolean formulas in conjunctive normal form. To this end define the synchronized context-free grammar $G = (V, S, T, I, P)$, with base nonterminal set $V = \{I, F, T\}$, alphabet of synchronizing symbols $S = \{0, 1, f\}$, terminals $T = \{0, 1, 2, \#\}$, and $P = P_1 \cup P_2 \cup P_3$, where

$$\begin{aligned}
P_1 &= \{I \to (F,\lambda)I, I \to (F,\lambda)\},\\
P_2 &= \{(F,\lambda) \to (F,0), (F,\lambda) \to (F,1)\}\\
&\quad \cup \{(F,0) \to 0(F,\lambda), (F,0) \to 1(F,\lambda), (F,0) \to 2(T,\lambda)\}\\
&\quad \cup \{(F,1) \to 0(F,\lambda), (F,1) \to 1(T,\lambda), (F,1) \to 2(F,\lambda)\}
\end{aligned}$$

and

$$\begin{aligned}
P_3 &= \{(T,\lambda) \to (T,0), (T,\lambda) \to (T,1)\} \cup \{(T,\lambda) \to (T,f)\}\\
&\quad \cup \{(T,0) \to 0(T,\lambda), (T,0) \to 1(T,\lambda), (T,0) \to 2(T,\lambda)\}\\
&\quad \cup \{(T,1) \to 0(T,\lambda), (T,1) \to 1(T,\lambda), (T,1) \to 2(T,\lambda)\}\\
&\quad \cup \{(T,f) \to \#\}.
\end{aligned}$$

This completes the description of the synchronized context-free grammar.

To start the derivation process one uses rules from P_1 leading to

$$I \Rightarrow^* (F,\lambda)\dots(F,\lambda),$$

where the non-synchronizing nonterminal F appears at least once, and is responsible for producing the coding of a satisfiable clause. A single letter 0, 1, or 2 of a clause coding is produced either by rules from P_2 or P_3 in a two step process, first guessing an assignment, which is stored in the synchronizing sequence, and then to partially evaluate the clause from left to right up to the considered variable. Whether a clause is *false* or *true* is simply stored in the nonterminal F or T, and the derivation process, which produces a clause coding, can only terminate if the clause under consideration is already *true*. Each clause coding is produced independently of each other, but in the $\{0,1\}^*f$ synchronizing sequence of the right-linear subtree rooted at the appropriate nonterminal F from above, an assignment to the Boolean variables will be stored. Then the synchronization mechanism of the grammar ensures that all these stored assignments have to be equal, and thus all clauses are evaluated over the same assignment. Therefore, the coding of a Boolean formula in conjunctive normal form is satisfiable if and only if its coding belongs to $L_z(G)$, for $z \in \{\mathrm{p}, \mathrm{e}\}$.

To make the work of the grammar clearer consider the Boolean formula $\varphi = (x_1 \vee \bar{x}_3 \vee x_4) \wedge x_2$. Clause $C_1 = x_1 \vee \bar{x}_3 \vee x_4$ is coded as $\mathrm{code}(C_1) = 1021$ since the number of overall variables is 4, and $\mathrm{code}(C_2) = 0100$ because $C_2 = x_2$. Thus, the coding of the Boolean formula reads as 1021#0100#. Next consider clause C_1 in more detail. The rules from P_2 and P_3 act on a sentential form (F,λ) producing the terminal string 1021# as follows:

$$\begin{aligned}
(F,\lambda) &\Rightarrow (F,0) \Rightarrow 1(F,\lambda) \Rightarrow 1(F,1) \Rightarrow 10(F,\lambda) \Rightarrow 10(F,0)\\
&\Rightarrow 102(T,\lambda) \Rightarrow 102(T,0) \Rightarrow 1021(T,\lambda) \Rightarrow 1021(T,f) \Rightarrow 1021\#.
\end{aligned}$$

Observe, that the synchronizing sequence equals $0100f$, which tells us that variable x_1 was set to *false*, variable x_2 to *true*, variable x_3 to *false*, and variable x_4 to *false*, too. Therefore,

$$I \Rightarrow (F,\lambda)I \overset{*}{\Rightarrow} 1021\#I \Rightarrow 1021\#(F,\lambda)$$

holds, where the subtree whose yield equals $1021\#$ induces the synchronizing sequence $0100f$. This synchronizing sequence must be the same for all root-to-leaf paths, and hence also for the subtree which is responsible for the second clause $C_2 = x_2$. Therefore, clause C_2 is also evaluated under the same assignment to the variables as listed above, and a sample derivation starting at (F, λ) looks as

$$\begin{aligned}&(F,\lambda) \Rightarrow (F,0) \Rightarrow 0(F,\lambda) \Rightarrow 0(F,1) \Rightarrow 01(T,\lambda)\\ &\Rightarrow 01(T,0) \Rightarrow 010(T,\lambda) \Rightarrow 010(T,0) \Rightarrow 0100(T,\lambda) \Rightarrow 0100(T,f) \Rightarrow 0100\#.\end{aligned}$$

Thus,

$$I \Rightarrow (F,\lambda)I \stackrel{*}{\Rightarrow} 1021\#I \Rightarrow 1021\#(F,\lambda) \stackrel{*}{\Rightarrow} 1021\#0100\#$$

is a z-acceptable derivation, for $z \in \{\mathrm{p}, \mathrm{e}\}$. Since $1021\#0100\#$ is the coding of the Boolean formula φ, it is shown that φ is satisfiable.

The given construction shows that the fixed membership problem for synchronized context-free languages in p- and e-mode of derivation is **NP**-complete. □

5 General Membership and Non-emptiness

In this section we consider the computational complexity of the general membership and non-emptiness problem for counter synchronized context-free and synchronized context-free languages. For ordinary context-free languages it is well-known that general membership and non-emptiness is computational equivalent. The next lemma shows that for the language families under consideration we find the same situation. Since the result follows by standard arguments as in the case of context-free languages we omit the straight-forward proof.

Lemma 13. *For synchronized context-free languages working in* p*-mode (*e*-mode, respectively) of derivation the general membership problem is logspace many-one equivalent to the non-emptiness problem. The statement remains valid for the family of counter synchronized context-free languages.* □

The next two lemmata show that the non-emptiness problems for counter synchronized context-free and synchronized context-free grammars are somehow related to the non-emptiness problems for one-way alternating finite automata. An one-way *alternating finite automaton* [Chandra, Kozen, and Stockmeyer 1981; Chandra and Stockmeyer 1976; Kozen 1976] is a quintuple $A = (Q, \Sigma, q_0, \delta, F)$, where Q is a finite set of states, Σ is a finite input alphabet, $q_0 \in Q$ is the initial state, $F \subseteq Q$ are the accepting states, and $\delta : Q \times \Sigma \to 2^{2^Q}$ is the transition function. Here 2^Q denotes the power set of Q. If for example, $\delta(q, a) = \{\{q_1, q_2\}, \{q_2\}\}$, for some $q \in Q$ and $a \in \Sigma$, it is to be thought of as the Boolean formula $\delta(q, a) = (q_1 \vee q_2) \wedge q_2$. As the input is read (from left to right), the automaton "builds" a propositional formula, and on reading an input a, replaces every $q \in Q$ in the formula by $\delta(q, a)$. The input w is accepted if and only if the formula that is built up by starting from q_0 and reading the whole input w is *true* on substituting *true* for q if $q \in F$, and *false* otherwise. Observe, that the propositional formula that is built up by the automaton is constructed in a top-down and evaluated in a bottom-up manner.

For our further investigations we need the non-emptiness problem for alternating finite automata, which is defined as follows:

– Given an alternating finite automaton A, i.e., an encoding $\langle A\rangle$, is $L(A) \neq \emptyset$?

Now we are ready to relate non-emptiness for alternating finite automata with non-emptiness for synchronized context-free grammars. At first we consider alternating finite automata with singleton input alphabet.

Lemma 14. *The non-emptiness problem for alternating finite automata with singleton input alphabet is logspace many-one reducible to the non-emptiness problem for counter synchronized context-free grammars working z-mode of derivation, with $z \in \{\mathrm{p}, \mathrm{e}\}$.*

Proof. Let $A = (Q, \{1\}, \delta, q_0, F)$ be an alternating finite automaton. We construct a counter synchronized context-free grammar $G = (V, S, T, I, P)$, where $V = Q \cup 2^Q \cup \{I\}$, the unions being disjoint, alphabet of synchronizing symbols $S = \{f, \#\}$, and terminal set $T = \{1\}$ such that $L_z(G) \neq \emptyset$, for $z \in \{\mathrm{p}, \mathrm{e}\}$, if and only if $L(A) \neq \emptyset$. The set of productions P contains the following rules: To start the derivation

$$I \to (q_0, \lambda).$$

is used. Moreover, for every $p \in Q$ with $\delta(p, 1) = \{M_{p,1}, \ldots, M_{p,n_p}\}$, where $M_{p,i} \subseteq Q$, for $1 \leq i \leq n_p$, the following productions are in P:

$$(p, \lambda) \to (M_{p,1}, f) \ldots (M_{p,n_p}, f)$$

and

$$(M_{p,i}, f) \to (q, \lambda) \quad \text{for } 1 \leq i \leq n_p \text{ and } q \in M_{p,i}.$$

These rules simulate a single step of the alternating finite automaton A reading letter 1. Finally, to terminate the derivation process, for every $p \in F$, the rules

$$(p, \lambda) \to (p, \#) \quad \text{and} \quad (p, \#) \to \lambda$$

are in P. This completes the description of G.

By induction one verifies that $1^n \in L(A)$ if and only if there is a derivation tree t of G whose yield is λ and every leaf μ of $t_1 = \mathrm{inner}(t)$ satisfies that $\mathrm{seq}_{t_1}(\mu) = f^n\#$. This shows that $L(A) \neq \emptyset$ if and only if $L_z(G) \neq \emptyset$, with $z \in \{\mathrm{p}, \mathrm{e}\}$. Since grammar G can be constructed within logarithmic space from a suitable description of the alternating finite automaton A, the stated result follows. □

Next, we consider alternating finite automata in general, i.e., where the input alphabet size is not restricted. There we find the following situation.

Lemma 15. *The non-emptiness problem for synchronized context-free grammars working z-mode of derivation, with $z \in \{\mathrm{p}, \mathrm{e}\}$ is logspace many-one reducible to the non-emptiness problem for alternating finite automata.*

Proof. Without loss of generality we assume that the given synchronized context-free grammar $G = (V, S, T, I, P)$ is in binary normal form without non-synchronizing nonterminals and working in e-mode of derivation. If this is not the case, we first apply the conversion of p-acceptance to e-acceptance as shown in Theorem 8, followed by the normal form transformation. Since these constructions can be performed within deterministic logarithmic space as mentioned after Corollary 7 we are left with a synchronized grammar that suits our needs. Observe, that it is essential in the below given construction that the synchronized context-free grammar is in normal form without non-synchronizing nonterminals, since the synchronizing sequence induced by an e-acceptable tree will become the input of the alternating finite automaton. We construct an alternating finite automaton $A = (Q, \Sigma, \delta, q_0, F)$, where

$$Q = \{\, q_Y \mid Y \in (V \times S) \,\} \cup \{\, p_{Y_1}, p_{Y_1 Y_2} \mid Y_1, Y_2 \in (V \times S) \,\} \cup \{q_0, q_{acc}, q_{rej}\},$$

input alphabet $\Sigma = S \cup \{\$\}$, where \$ is a new symbol not contained in S, and the set of accepting states $F = \{q_{acc}\}$, such that $L(A) \neq \emptyset$ if and only if $L_{\mathrm{e}}(G) \neq \emptyset$. The transition function δ is defined as follows: Since G is in binary normal form without non-synchronizing nonterminals, all rules with left-hand side I are of the form $I \to Y$ with $Y \in (V \times S)$ due to the construction as given in the proof of Theorem 6. Let $I \to Y_1, \ldots, I \to Y_m$, for some m, be all productions in P with left-hand side I. Then we define

$$\delta(q_0, \$) = \{\{q_{Y_1}, \ldots, q_{Y_m}\}\}.$$

Assume that $(A, s) \to \alpha_1, \ldots, (A, s) \to \alpha_n$, for some n, be all productions in P with left-hand side (A, s), for $A \in V$ and $s \in S$. Because of the binary normal form $\alpha_i \in (V \times S) \cup (V \times S)^2 \cup T \cup \{\lambda\}$, for $1 \leq i \leq n$. We set

$$\delta(q_{(A,s)}, s) = \{\{p_{\alpha_1}, \ldots, p_{\alpha_n}\}\}$$

and for $1 \leq i \leq n$ define

$$\delta(p_{\alpha_i}, \$) = \begin{cases} \{\{q_{Y_1}\}, \{q_{Y_2}\}\} & \text{if } \alpha_i = Y_1 Y_2 \text{ with } Y_1 Y_2 \in (V \times S)^2 \\ \{\{q_{Y_1}\}\} & \text{if } \alpha_i = Y_1 \text{ with } Y_1 \in (V \times S) \\ \{\{q_{acc}\}\} & \text{if } \alpha_i = a \text{ for } a \in T \cup \{\lambda\}. \end{cases}$$

Finally, for $q \in \{q_{acc}, q_{rej}\}$ and $s \in S \cup \{\$\}$ let

$$\delta(q, s) = \{\{q_{rej}\}\}.$$

This completes the description of the alternating finite automaton A.

The idea behind the given construction is to use alternation to guess a possible derivation tree, and thus it is quite similar to the ordinary context-free grammar case—the application of a rule corresponds to an existential node while the check whether all nonterminals in a sentential form may terminate is done by universal nodes. The verification that the induced synchronizing sequences from the guessed derivation tree are all equal, is done by the (synchronized) work of the alternating finite automaton. To make this a little bit clearer, consider the rules $(A, s) \to (B, t_1)(C, t_2)$ and $(A, s) \to (D, t_3)(E, t_4)$. Then, by construction, $\delta(q_{(A,s)}, s\$)$ is expressed by the formula $(q_{(B,t_1)} \wedge q_{(C,t_2)}) \vee (q_{(D,t_3)} \wedge q_{(E,t_4)})$. This reflects the fact, that nonterminal (A, s) derives a word in G if and only if the sentential form $(B, t_1)(C, t_2)$ or $(D, t_3)(E, t_4)$ derives a word in G. The sentential form $(B, t_1)(C, t_2)$ ($(D, t_3)(E, t_4)$, respectively) derives a word if and only if (B, t_1) ((D, t_3), respectively) and (C, t_2) ((E, t_4), respectively) derive words in G and the induced synchronizing sequences appropriately fit together. Having this in mind one can prove the following result.

Let $h : S \to S \cup \{\$\}$ be a homomorphism defined by $h(s) = s\$$ for $s \in S$. Then, by induction, one verifies that $\$h(w) \in L(A)$ if and only if there is a derivation tree t in G whose yield is in T^* and for every leaf μ of $t_1 = \text{inner}(t)$ satisfying $\text{seq}_{t_1}(\mu) = w$. This shows that $L(A) \neq \emptyset$ if and only if $L_e(G) \neq \emptyset$. Since automaton A can be constructed with logarithmic space from a suitable description of the synchronized context-free grammar G, the stated result follows. □

Finally, we state the main theorem of this section.

Theorem 16. *The following problems are* **PSpace***-complete under logarithmic space many-one reductions:*

1. *The non-emptiness problems for both counter synchronized context-free and synchronized context-free grammars.*
2. *The general membership problems for both counter synchronized context-free and synchronized context-free grammars.*

Proof. 1. In [Jiang and Ravikumar 1991] it was shown that the non-emptiness problem for alternating finite automata is **PSpace**-complete under logspace many-one reductions. Moreover, the problem remains **PSpace**-complete if the underlying devices are restricted to have singleton input alphabet [Holzer 1996]. Thus, by Lemmata 14 and 15 it immediately follows that non-emptiness for both counter synchronized context-free and synchronized context-free grammars is **PSpace**-complete, too.
2. The **PSpace**-completeness follows from Lemma 13 and the above given result on the complexity of the non-emptiness problems. □

6 Conclusions

We have studied the computational complexity of the fixed, the general, and the non-emptiness problem for counter synchronized and synchronized context-free languages. Here a counter synchronized context-free grammar is only allowed to use one synchronizing and an end marker symbol. It is shown that the generative power of these devices induces a family of languages which lies properly in between the family of context-free and the family of synchronized context-free languages. From the computational point of view, the fixed membership problem for counter synchronized languages is easy, i.e., LOG(**CF**)-complete, while that for synchronized context-free languages becomes intractable, i.e., **NP**-complete. In the case of the general membership problem we obtain **PSpace**-completeness for both, counter synchronized and synchronized context-free grammars in general. A similar result holds for the non-emptiness problem of the grammars under consideration.

There are several directions for future research. One might be to consider a counter version of block-synchronized context-free grammars and languages from the formal language as well as the computational complexity point of view. Here it is worth to mention, that the membership and non-emptiness problem is decidable for block-synchronized context-free grammars, while it is an open problem for strong block-synchronized context-free grammars—we refer to [Jürgensen and Salomaa 1997]. Counter block-synchronized context-free grammars may be useful to attack this problem, due to the simple structure of the synchronizing sequences.

References

[Aho 1968] Aho, A. V. (1968). Indexed grammars—an extension of context-free grammars. *Journal of the ACM 15*, 647–671.

[Balcázar, Díaz, and Gabarró 1988] Balcázar, J. L., J. Díaz, and J. Gabarró (1988). *Structural Complexity I*, Volume 11 of *EATCS Monographs on Theoretical Computer Science*. Springer.

[Chandra, Kozen, and Stockmeyer 1981] Chandra, A. K., D. C. Kozen, and L. J. Stockmeyer (1981, January). Alternation. *Journal of the ACM 28*(1), 114–133.

[Chandra and Stockmeyer 1976] Chandra, A. K. and L. J. Stockmeyer (1976, October). Alternation. In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, Houston, Texas, pp. 98–108. IEEE Computer Society Press.

[Dassow and Păun 1989] Dassow, J. and G. Păun (1989). *Regulated Rewriting in Formal Language Theory*, Volume 18 of *EATCS Monographs in Theoretical Computer Science*. Springer.

[Duske, Middendorf, and Parchmann 1992] Duske, J., M. Middendorf, and R. Parchmann (1992). Indexed counter languages. *RAIRO–Informatique théorique et Applications / Theoretical Informatics and Applications 26*(1), 93–113.

[Holzer 1996] Holzer, M. (1996). On emptiness and counting for alternating finite automata. In J. Dassow, G. Rozenberg, and A. Salomaa (Eds.), *Developments in Language Theory II; at the Crossroads of Mathematics, Computer Science and Biology*, pp. 88–97. World Scientific.

[Hopcroft and Ullman 1979] Hopcroft, J. E. and J. D. Ullman (1979). *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley.

[Hromkovič, Karhumäki, Rovan, and Slobodová 1991] Hromkovič, J., J. Karhumäki, B. Rovan, and A. Slobodová (1991). On the power of synchronization in parallel computations. *Discrete Applied Mathematics 32*, 155–182.

[Hromkovič, Rovan, and Slobodová 1994] Hromkovič, J., B. Rovan, and A. Slobodová (1994). Deterministic versus nondeterministic space in terms of synchronized alternating machines. *Theoretical Computer Science 132*, 319–336.

[Jiang and Ravikumar 1991] Jiang, T. and B. Ravikumar (1991). A note on the space complexity of some decision problems for finite automata. *Information Processing Letters 40*, 25–31.

[Jürgensen and Salomaa 1997] Jürgensen, H. and K. Salomaa (1997). Block-synchronized context-free grammars. In D.-Z. Du and J.-I. Ko (Eds.), *Advances in Algorithms, Languages, and Complexity*, pp. 111–137. Kluwer.

[Kozen 1976] Kozen, D. (1976, October). On parallelism in Turing machines.

In *Proceedings of the 17th Annual Symposium on Foundations of Computer Science*, Houston, Texas, pp. 89–97. IEEE Computer Society Press.

[Salomaa 1994] Salomaa, K. (1994). Synchronized tree automata. *Theoretical Computer Science 127*, 25–51.

[van Leeuwen 1975] van Leeuwen, J. (1975, May). The membership problem for ET0L-languages is polynomially complete. *Information Processing Letters 3*(5), 138–143.