On Alternating Non-Context-Free Grammars

Etsuro Moriya*
Advanced Research Institute for Science and Engineering
and
Department of Mathematics, School of Education
Waseda University, Shinjuku-ku, Tokyo, 169-8050, Japan
moriya@waseda.jp

and

Friedrich Otto
Fachbereich Elektrotechnik/Informatik, Universität Kassel
34109 Kassel, Germany
otto@theory.informatik.uni-kassel.de

October 25, 2007

Abstract

We study several extensions of the notion of alternation from context-free grammars to context-sensitive and arbitrary phrase-structure grammars. Thereby new grammatical characterizations are obtained for the class of languages that are accepted by alternating pushdown automata.

1 Introduction

Alternation is a powerful concept that was first introduced by Chandra and Stockmeyer [ChSt76, CKS81] for general Turing machines and then by Ladner, Lipton, and Stockmeyer [LLS78, LLS84] for pushdown automata. Thereafter the notion of alternation has been studied for a variety of other devices. In particular, in [Mor89] one of the authors introduced the concept of alternation for context-free grammars (ACFG for short) by distinguishing between *existential* and *universal variables* (nonterminals) with the aim of deriving a grammatical characterization for the class of languages that are accepted by alternating pushdown automata (APDA for short).

As no such characterization was obtained in [Mor89], further studies of the notion of alternation for context-free grammars and pushdown automata followed (see, e.g., [IJW92, MoNa97, MHHO, MoOt07]). In [IJW92] such a characterization was finally reached through *linear-erasing* ACFGs. Further, with the definition of the *state-alternating context-free grammar* (sACFG for short) in [MHHO], a new way of introducing alternation into context-free grammars was proposed that was inspired by the notion of *context-free grammar with states*

*Major parts of this work were done while Etsuro Moriya was visiting at the Fachbereich Elektrotechnik/Informatik, Universität Kassel.

of Kasai [Kas70]. A context-free grammar with states is a context-free grammar that is equipped with a finite set of states. Each production of the grammar is combined with a change of state, and accordingly a sentential form of such a grammar is a pair consisting of a state and a string (the proper sentential form). A production is then applicable to a sentential form only if the state of the production is identical to the state of the sentential form. Now by distinguishing between *existential* and *universal states* the sACFG is obtained. Thus, while in an ACFG the variable on the lefthand side of a production determines whether this production is to be used in an existential or a universal fashion, it is the states that make this distinction in an sACFG. For each ACFG G, an sACFG G' can be constructed such that G and G' generate the same language, but it is still open whether or not the converse is true. At least for linear context-free grammars, and therewith in particular for right-linear (that is, regular) grammars, it has been shown that the two notions of alternation yield the same expressive power. Actually, both types of alternating right-linear grammars just generate the regular languages. Further, it turned out that sACFGs working in leftmost derivation mode generate exactly those languages that are accepted by APDAs [MHHO]. In this way another grammatical characterization for this class of languages was obtained.

In [MoOt07] the authors proposed and studied a different way of defining the notion of alternation for pushdown automata. Instead of distinguishing between existential and universal states as in the original definition [LLS78, LLS84], here the pushdown symbols are used for this purpose. As it turned out these so-called *stack-alternating pushdown automata* are equivalent in expressive power to the original variant of the alternating pushdown automaton. However, the *stateless* variant of the stack-alternating pushdown automaton accepts exactly the languages that are generated by ACFGs in leftmost derivation mode [MoOt07].

It is known that the class of languages accepted by APDAs coincides with the deterministic time complexity class $\mathsf{ETIME} = \bigcup_{c>0} \mathsf{DTIME}(c^n)$ as well as with the alternating space complexity class $\mathsf{ALINSPACE}$, that is, the class of languages that are accepted by *alternating linear bounded automata* (ALBA) [CKS81, LLS84]. As in the classical (non-alternating) setting pushdown automata correspond to context-free grammars and linear bounded automata correspond to context-sensitive (or monotone) grammars, the above results raise the question about the expressive power of alternating non-context-free grammars.

In this paper we carry the notion of alternation over to general phrase structure and context-sensitive grammars. In fact, we consider both types of alternation for grammars mentioned above. By distinguishing between existential and universal variables we obtain the *alternating phrase-structure grammars* (APSG) and the *alternating context-sensitive grammars* (ACSG). By considering grammars with states, for which we distingush between existential and universal states, we obtain the *state-alternating phrase-structure grammars* (sAPSG) and the *state-alternating context-sensitive grammars* (sACSG). For state-alternating grammars it is rather straightforward to define the notion of derivation. However, for the other type of alternating grammars there are various different ways of defining the corresponding derivation relation. We will consider two such definitions, and we will prove that they are in fact equivalent in a weak sense, that is, for a fixed alternating grammar the two definitions yield different languages, but to each alternating grammar working with the one notion of derivation there is another grammar of the same type that is working with the other notion of derivation, and that generates the same language. In addition, we will consider two derivation strategies or modes: leftmost derivations and unrestricted derivations.

Actually, it will turn out that for phrase-structure grammars as well as for context-sensitive grammars, the state-alternating variant is equivalent to the alternating variant. Actually, this equivalence is valid for the leftmost derivation mode and for the unrestricted

derivation mode. With respect to the unrestricted derivation mode the APSGs just give another characterization for the class RE of recursively enumerable languages. However, with respect to the leftmost derivation mode, they have the same generative power as sACFGs. This can be interpreted as the counterpart to the corresponding result for non-alternating grammars, which states that in leftmost mode general phrase-structure grammars can only generate context-free languages [Mat64]. Our second main result states that with respect to the unrestricted derivation mode ACSGs generate exactly those languages that are accepted by alternating linear-bounded automata. As ALBAs and APDAs accept the same languages, APSGs (working in leftmost mode) and ACSGs (working in unrestricted mode) give new grammatical characterizations for the class of languages that are accepted by APDAs.

Finally, our results show that, when working in leftmost mode, ACSGs generate a subclass of this class of languages. It remains open, however, whether this is a proper subclass. These facts should be compared to the fact that no inclusion relation is known between the class of languages generated by sACFGs (or ACFGs) in leftmost mode and the class of languages generated by sACFGs (or ACFGs) in unrestricted mode.

This paper is structured as follows. In Section 2 the basic definitions of alternating grammars and state-alternating grammars are given. Also we discuss various ways of defining the notion of *derivation* for these grammars. One option consists in using a specific variable occurring in a sentential form α to determine whether α itself is existential or universal. Another option consists in using a distinguished occurrence of a variable in the lefthand side of a production to declare that production itself to be existential or universal. We will show that both these options yield the same generative power for the classes of alternating grammars considered, if in both approaches we use the leftmost variable occurrence for this choice.

Then in Section 3 we study the relationship of alternating grammars to state-alternating grammars working in leftmost mode. The main result of this section states that, with respect to leftmost derivations, APSGs (and therewith also sAPSGs) have the same expressive power as sACFGs. In Section 4 we compare alternating grammars to state-alternating grammars working in unrestricted mode. Finally, Section 5 is devoted to the study of the relationship between ACSGs on the one hand and the ALBAs on the other hand. The paper closes with Section 6, where some open problems are presented.

2 Alternating Grammars

As defined in [Mor89] an *alternating context-free grammar* is a grammar $G = (V, U, \Sigma, P, S)$, where V is a set of variables (or nonterminals), $U \subseteq V$ is a set of *universal* variables, while the variables in $V \setminus U$ are called *existential*, Σ is a set of terminals, S is the start symbol, and P is a set of context-free productions. In the derivation process an existential variable is rewritten as usual, but a universal variable is rewritten by applying all productions with that variable as left-hand side simultaneously, thus giving a finite number of successor sentential forms. In this way a derivation is not a linear chain, but it has the form of a tree. A terminal word w can be derived from G, if there exists a finite derivation tree in the above sense such that the root is labelled with the start symbol and all leaves are labelled with w. By ACFG we denote the class of alternating context-free grammars.

An *alternating phrase-structure grammar* is a general phrase-structure grammar $G = (V, U, \Sigma, P, S)$, where V is a set of variables (or nonterminals), $U \subseteq V$ is a set of *universal* variables, while the variables in $V \setminus U$ are called *existential*, Σ is a set of terminals, S is the

start symbol, and P is a set of productions, where $(\ell, r) \in P$ implies that $\ell, r \in (V \cup \Sigma)^*$, and ℓ contains at least one variable. If $|\ell| \leq |r|$ holds for all productions $(\ell, r) \in P$, then G is called an *alternating context-sensitive grammar*. By APSG we denote the class of all alternating phrase-structure grammars, while ACSG denotes the class of all alternating context-sensitive grammars.

It remains to specify the way in which derivations are performed by an alternating grammar G. In particular, we must determine a way to distinguish between existential and universal derivation steps. There are various options. First of all we can use a specific nonterminal occurring in a sentential form α to determine whether α itself is existential or universal. For example, we could use the leftmost variable occurring in α for that, that is, if $\alpha = xA\beta$, where $x \in \Sigma^*$, $A \in V$, and $\beta \in (V \cup \Sigma)^*$, then we call α an *existential sentential form* if $A \in V \setminus U$, and we call α a *universal sentential form* if $A \in U$. To apply a derivation step to α, we nondeterministically choose a substring ℓ of α that occurs as the left-hand side of one or more rules of P. Now if α is existential, then one of these rules is chosen, and $\alpha = \gamma\ell\delta$ is rewritten into $\gamma r\delta$, where $(\ell, r) \in P$ is the rule chosen. If α is universal, then let $(\ell, r_1), \ldots, (\ell, r_m)$ be those rules of P with left-hand side ℓ. Now all these productions are applied simultaneously, thus giving a finite number of successor sentential forms $\gamma r_1\delta, \ldots, \gamma r_m\delta$. In this way a derivation is not a linear chain, but it has the form of a tree. A terminal word w can be derived from G, if there exists a finite derivation tree in the above sense such that the root is labelled with the start symbol S and all leaves are labelled with w. Observe that in this way the rules themselves are neither existential nor universal, but that it purely depends on the type of the leftmost variable in the actual sentential form whether the next derivation step is existential or universal. Below we will use the notation $\Rightarrow_G^c$ to denote this derivation relation. By $L^c(G)$ we denote the language that is generated by G using this derivation relation.

As an alternative we can use a distinguished occurrence of a variable in the left-hand side of a rule to declare that rule as being existential or universal. Of course, this must be done in a consistent way, that is, for all rules with the same left-hand side, the same variable occurrence must be chosen. Then for $\alpha = \gamma\ell\delta$, if ℓ is existential, then α is rewritten into $\gamma r\delta$, where $(\ell, r) \in P$ is one of the rules with left-hand side ℓ, and if ℓ is universal, then α is rewritten simultaneously into $\gamma r_1\delta, \ldots, \gamma r_m\delta$, where $(\ell, r_1), \ldots, (\ell, r_m)$ are all the rules in P with left-hand side ℓ. For example, we could always use the leftmost variable occurrence in ℓ for this. We will use the notation $\Rightarrow_G$ to denote this derivation relation. By $L(G)$ we denote the language that is generated by G using this derivation relation.

The following example demonstrates that the derivation relations $\Rightarrow_G^c$ and $\Rightarrow_G$ will in general yield different languages.

Example 2.1. Let $G = (\{S, A, B\}, \{B\}, \{a, b, c\}, P, S)$ with $P = \{S \to AB, A \to a, A \to ab, B \to c, B \to bc\}$. Then with respect to $\Rightarrow_G^c$, G generates the language $L^c(G) = \{ac, abc, abbc\}$, while with respect to $\Rightarrow_G$, we only obtain the language $L(G) = \{abc\}$. The reason is the fact that with respect to $\Rightarrow_G^c$, the rules with left-hand side B can be applied in existential fashion as long as the variable A is still present in the actual sentential form.

However we have the following result.

Proposition 2.2. *For each alternating phrase-structure grammar G, there exists an alternating phrase-structure grammar G' such that $L(G') = L^c(G)$.*

Proof. Let $G = (V, U, \Sigma, P, S)$ be an alternating phrase-structure grammar. Then $G' :=$

$(V', U', \Sigma, P', \dot{S})$ is defined as follows:

$$\begin{aligned} V' &:= \{ A, A_e, A_u, \hat{A}_u, A_w, A_f, \dot{A} \mid A \in V \} \cup \{ \bar{a}_e, \bar{a}_u \mid a \in \Sigma \} \cup \{\dot{F}\}, \\ U' &:= \{ \hat{A}_u \mid A \in V \} \cup \{ A_w, A_f \mid A \in U \}, \end{aligned}$$

and P' consists of the following rules, where $A, B, C \in V$, $a \in \Sigma$, $x, y \in \Sigma^*$, and $\alpha, \beta, r \in (V \cup \Sigma)^*$:

(1)	$xA_e\alpha$	$\to$	r	if $(xA\alpha, r) \in P$,
(2)	$x\hat{A}_u\alpha$	$\to$	r	if $(xA\alpha, r) \in P$,
(3)	$\dot{A}$	$\to$	A_w	for all $A \in V$,
(4)	$\dot{A}$	$\to$	A_f	for all $A \in V$,
(5)	$xA_w\alpha$	$\to$	$y\dot{B}\beta$	if $(xA\alpha, yB\beta) \in P$,
(6)	$xA_f\alpha$	$\to$	$y\dot{B}\beta$	if $(xA\alpha, yB\beta) \in P$,
(7)	$xA_w\alpha$	$\to$	$y\dot{F}$	if $(xA\alpha, y) \in P$,
(8)	$xA_f\alpha$	$\to$	y	if $(xA\alpha, y) \in P$,
(9)	$\dot{F}a$	$\to$	$a\dot{F}$,	
(10)	$\dot{F}A$	$\to$	$\dot{A}$,	
(11)	$\dot{A}B$	$\to$	$\dot{A}B_e$	if $A \in V \setminus U$,
(12)	$\dot{A}B$	$\to$	$\dot{A}B_u$	if $A \in U$,
(13)	$\dot{A}B$	$\to$	$\dot{A}\hat{B}_u$	if $A \in U$,
(14)	$\dot{A}a$	$\to$	$\dot{A}\bar{a}_e$	if $A \in V \setminus U$,
(15)	$\dot{A}a$	$\to$	$\dot{A}\bar{a}_u$	if $A \in U$,
(16)	B_eC	$\to$	BC_e,	
(17)	B_uC	$\to$	BC_u,	
(18)	B_uC	$\to$	$B\hat{C}_u$,	
(19)	B_ea	$\to$	$B\bar{a}_e$,	
(20)	B_ua	$\to$	$B\bar{a}_u$,	
(21)	$\bar{a}_eC$	$\to$	aC_e,	
(22)	$\bar{a}_uC$	$\to$	aC_u,	
(23)	$\bar{a}_uC$	$\to$	$a\hat{C}_u$,	
(24)	$\bar{a}_eb$	$\to$	$a\bar{b}_e$,	
(25)	$\bar{a}_ub$	$\to$	$a\bar{b}_u$.	

The leftmost variable within a sentential form is marked by the dot. Using indices e and u the information about the type of this variable (existential or universal) is sent from left to right using the existential variables $B_e, B_u, \bar{a}_e, \bar{a}_u$ $(B \in V, a \in \Sigma)$. In the universal case at the proper place a universal variable of the form $\hat{B}_u$ is generated. Then a variant of a rule of P of type (1) (in the existential case) or (2) (in the universal case) is applied to simulate the corresponding $\Rightarrow_G^c$-step. If a rule is applied that involves the leftmost variable within the current sentential form, then the variants with indices w or f are used to nondeterministically distinguish between the case that further variables occur in the sentential form further to the right (case w) or that no further variables occur (case f). In the former case the dot is moved right to the leftmost variable in the next sentential form, using the variable $\dot{F}$ to cover the case that the rule used in the actual derivation step has no variable occurrences in its right-hand side. By induction on the number of steps in a $\Rightarrow_G^c$-derivation it can now be shown that $L^c(G) \subseteq L(G')$, and analogously the converse inclusion can be established. □

Observe that in the above construction the rules of type (10) are the only ones that are not monotone if the rules of G are monotone. In this case, however, we can use new variables

of the form $\dot{\bar{a}}$ instead of $\dot{F}$, as in a monotone grammar no ε-rules occur. Thus, we also have the following result.

Proposition 2.3. *For each alternating context-sensitive grammar G, there exists an alternating context-sensitive grammar G' such that $L(G') = L^c(G)$.*

Further, we have the following converse of Proposition 2.2.

Proposition 2.4. *For each alternating phrase-structure grammar G, there exists an alternating phrase-structure grammar G' such that $L^c(G') = L(G)$.*

Proof. Let $G = (V, U, \Sigma, P, S)$ be an alternating phrase-structure grammar. Then $G' := (V', U', \Sigma, P', \dot{S}_e)$ is defined as follows:

$$\begin{aligned} V' &:= \{ A, A_e, A_u, \dot{A}_e, \dot{A}_u, \hat{A}_e, \hat{A}_u, \bar{A}_u, A'_u, \dot{A}_e^{(w)}, \dot{A}_e^{(f)}, \dot{A}_u^{(w)}, \dot{A}_u^{(f)} \mid A \in V \} \cup \\ &\quad \{F, \hat{F}, H, \hat{H}\} \cup \{ \bar{a}_e, \bar{a}_u \mid a \in \Sigma\}, \\ U' &:= \{ \hat{A}_u, \dot{A}_u^{(w)}, \dot{A}_u^{(f)} \mid A \in U \}, \end{aligned}$$

and P' contains the following rules, where $A, B \in V$, $a, b \in \Sigma$, $x, y \in \Sigma^*$, and $\alpha, \beta, r \in (V \cup \Sigma)^*$:

$$\begin{array}{lrcll}
(1) & \dot{A}_e & \to & \dot{A}_u, & \\
(2) & \dot{A}_e B & \to & \hat{A}_e B_e, & \\
(3) & \dot{A}_e a & \to & \hat{A}_e \bar{a}_e, & \\
(4) & \dot{A}_u B & \to & A'_u B_u, & \\
(5) & \dot{A}_u B & \to & A'_u \hat{H} \bar{B}_u, & \\
(6) & \dot{A}_u a & \to & A'_u \bar{a}_u, & \\
(7) & A_e B & \to & A B_e, & \\
(8) & A_u B & \to & A B_u, & \\
(9) & A_u B & \to & A \hat{H} \bar{B}_u, & \\
(10) & \bar{a}_e B & \to & a B_e, & \\
(11) & \bar{a}_e b & \to & a \bar{b}_e, & \\
(12) & \bar{a}_u B & \to & a B_u, & \\
(13) & \bar{a}_u B & \to & a \hat{H} \bar{B}_u, & \\
(14) & \bar{a}_u b & \to & a \bar{b}_u, & \\
(15) & x A_e \alpha & \to & \hat{F} \beta & \text{if } (xA\alpha, \beta) \in P,\ A \in V \setminus U, \\
(16) & x H \bar{A}_u \alpha & \to & \hat{F} \beta & \text{if } (xA\alpha, \beta) \in P,\ A \in U, \\
(17) & C \hat{F} & \to & \hat{F} C & \text{for all } C \in V \cup \Sigma, \\
(18) & C \hat{H} & \to & \hat{H} C & \text{for all } C \in V \cup \Sigma, \\
(19) & H C & \to & C H & \text{for all } C \in V \cup \Sigma, \\
(20) & \hat{A}_e \hat{F} & \to & \dot{A}_e, & \\
(21) & \hat{A}_u \hat{F} & \to & \dot{A}_e, & \\
(22) & A'_u \hat{H} & \to & \hat{A}_u H, & \\
(23) & \dot{A}_e & \to & \dot{A}_e^{(w)}, & \\
(24) & \dot{A}_e & \to & \dot{A}_e^{(f)}, & \\
(25) & \dot{A}_u & \to & \dot{A}_u^{(w)}, & \\
(26) & \dot{A}_u & \to & \dot{A}_u^{(f)}, & \\
(27) & x \dot{A}_e^{(w)} \alpha & \to & y \dot{B}_e \beta & \text{if } (xA\alpha, yB\beta) \in P \text{ and } A \in V \setminus U, \\
(28) & x \dot{A}_e^{(f)} \alpha & \to & y \dot{B}_e \beta & \text{if } (xA\alpha, yB\beta) \in P \text{ and } A \in V \setminus U,
\end{array}$$

$$
\begin{array}{lrcll}
(29) & x\dot{A}_u^{(w)}\alpha & \to & y\dot{B}_e\beta & \text{if } (xA\alpha, yB\beta) \in P \text{ and } A \in U, \\
(30) & x\dot{A}_u^{(f)}\alpha & \to & y\dot{B}_e\beta & \text{if } (xA\alpha, yB\beta) \in P \text{ and } A \in U, \\
(31) & x\dot{A}_e^{(w)}\alpha & \to & yF & \text{if } (xA\alpha, y) \in P \text{ and } A \in V \setminus U, \\
(32) & x\dot{A}_e^{(f)}\alpha & \to & y & \text{if } (xA\alpha, y) \in P \text{ and } A \in V \setminus U, \\
(33) & x\dot{A}_u^{(w)}\alpha & \to & yF & \text{if } (xA\alpha, y) \in P \text{ and } A \in U, \\
(34) & x\dot{A}_u^{(f)}\alpha & \to & y & \text{if } (xA\alpha, y) \in P \text{ and } A \in U, \\
(35) & FB & \to & \dot{B}_e, & \\
(36) & Fa & \to & aF. &
\end{array}
$$

Consider a derivation step $yA\gamma xB\alpha\delta \Rightarrow_G yA\gamma r\delta$, where $x, y \in \Sigma^*$, $A, B \in V$, and $\alpha, \gamma, \delta, r \in (V \cup \Sigma)^*$, and $(xB\alpha \to r) \in P$ is the production being applied. This step is simulated by a sequence of $\Rightarrow_{G'}^c$-steps as follows.

Before the simulation begins the sentential form $yA\gamma xB\alpha\delta$ is encoded as $y\dot{A}_e\gamma xB\alpha\delta$, that is, the leftmost variable in a sentential form is marked by a dot. If the G-step to be simulated is universal, that is, if $B \in U$, then the variable $\dot{A}_e$ is rewritten to $\dot{A}_u$ by production (1). Observe that $\dot{A}_e$ and $\dot{A}_u$ are both existential variables of G'. Next the index e or u is moved to the right using productions (2) to (14). In this process the variable $\dot{A}_e$ or $\dot{A}_u$ is replaced by the variable $\hat{A}_e$ or A'_u, respectively, which are both existential. In this way the sentential form $y\hat{A}_e\gamma xB_e\alpha\delta$ or $yA'_u\gamma x\hat{H}\bar{B}_u\alpha\delta$ is reached. In the former case production (15) can now be used to simulate the corresponding existential step of G, which yields the sentential form $y\hat{A}_e\gamma\hat{F}r\delta$.

The universal case is more complicated. In this case the variable $\hat{H}$ is moved left by using production (18), and then the substring $A'_u\hat{H}$ is rewritten into $\hat{A}_uH$ by production (22), which yields the sentential form $y\hat{A}_uH\gamma x\bar{B}_u\alpha\delta$. Observe that the variable $\hat{A}_u$ is universal, that is, as long as this variable is the leftmost variable of the actual sentential form, all derivation steps are universal. However, to the sentential form $y\hat{A}_uH\gamma x\bar{B}_u\alpha\delta$ only production (19) is applicable, which shifts the symbol H to the right until the sentential form $y\hat{A}_u\gamma xH\bar{B}_u\alpha\delta$ is obtained. Now production (16) becomes applicable, that is, the universal step of G is being simulated, which yields the sentential forms of the form $y\hat{A}_u\gamma\hat{F}r\delta$.

In both cases only production (17) is now applicable, which moves the symbol $\hat{F}$ to the left until it meets the symbol $\hat{A}_e$ or $\hat{A}_u$, respectively, which it then converts into $\dot{A}_e$ by productions (20) or (21). In this way the sentential form $y\dot{A}_e\gamma r\delta$ is obtained, and the simulation of the next derivation step of G starts.

Productions (23) to (36) are used whenever a G-derivation step must be simulated that involves the leftmost variable occurring in the actual sentential form. This is similar to the situation in the proof of Proposition 2.2. It follows that $L^c(G') = L(G)$ holds. □

Again it is possible to adjust the construction in the proof of Proposition 2.4 to the case of context-sensitive grammars, which yields the following result.

Proposition 2.5. *For each alternating context-sensitive grammar G, there exists an alternating context-sensitive grammar G' such that $L^c(G') = L(G)$.*

Thus, we see that for context-sensitive as well as for general phrase-structure grammars, both definitions of alternation yield the same expressive power. Therefore we restrict our attention in the rest of this paper to alternating grammars for which the leftmost variable occurring in the lefthand side of a production determines whether the production itself is existential or universal.

In addition to the unrestricted derivation mode, we are also interested in the so-called *leftmost* derivation mode. A derivation step $\alpha = \gamma\ell\delta \Rightarrow_G \gamma r\delta$, respectively $\alpha = \gamma\ell\delta \Rightarrow_G (\gamma r_1\delta, \ldots, \gamma r_m\delta)$, is called *leftmost* if $\gamma \in \Sigma^*$, that is, this step involves the leftmost variable occurrence in α. By $L_{\mathsf{lm}}(G)$ we denote the language consisting of all terminal words that G generates by leftmost derivations. It is obvious that with respect to leftmost derivations the above two definitions of the derivation process of an alternating grammar coincide, if in both definitions the leftmost variable occurrence is chosen.

The following example illustrates the behaviour of leftmost derivations in non-context-free grammars.

Example 2.6. Let $G = (V, \Sigma, S, P)$ be the context-sensitive grammar that is given through $V := \{S, B\}$, $\Sigma := \{a, b, c\}$, and $P := \{S \to aSBc, S \to aBc, aB \to ab, bB \to bb, cB \to Bc\}$. Consider the following leftmost derivation mod G:

$$\begin{array}{llllll} S & \Rightarrow_G & aSBc & \Rightarrow_G & aaSBcBc & \Rightarrow_G & aaaBcBcBc \\ & \Rightarrow_G & aaabcBcBc & \Rightarrow_G & aaabBccBc & \Rightarrow_G & aaabbccBc \\ & \Rightarrow_G & aaabbcBcc & \Rightarrow_G & aaabbBccc & \Rightarrow_G & aaabbbccc. \end{array}$$

Actually, it can be shown that in leftmost mode G generates the non-context-free language $L_{\mathsf{lm}}(G) = \{\, a^n b^n c^n \mid n \geq 1 \,\} = L(G)$.

Let $G' = (V', \Sigma, S, P')$ be the context-sensitive grammar that is given through $V' := \{S, A, B\}$, $\Sigma := \{a, b, c\}$, and $P' := \{S \to ASBc, S \to ABc, AB \to Ab, bB \to bb, cB \to Bc, A \to a\}$. Consider the following leftmost derivation mod G':

$$\begin{array}{llllll} S & \Rightarrow_{G'} & ASBc & \Rightarrow_{G'} & aSBc & \Rightarrow_{G'} & aASBcBc \\ & \Rightarrow_{G'} & aaSBcBc & \Rightarrow_{G'} & aaABcBcBc. & & \end{array}$$

To the sentential form $aaABcBcBc$ the rules $A \to a$ and $AB \to Ab$ both apply in leftmost mode. The first one yields the sentential form $aaaBcBcBc$, to which no further leftmost derivation step is applicable, while the second one yields the sentential form $aaAbcBcBc$, from which we can continue with the following leftmost derivation:

$$aaAbcBcBc \;\Rightarrow_{G'}\; aaabcBcBc \;\Rightarrow^*_{G'}\; aaabbbccc.$$

Thus, even in leftmost derivation mode rules with non-identical lefthand sides may both be applicable to the same sentential form.

Finally, let $\hat{G} = (\hat{V}, \Sigma, S, \hat{P})$ be the context-sensitive grammar that is given through $\hat{V} := \{S, A, B, B', C\}$, $\Sigma := \{a, b, c\}$, and $\hat{P} := \{S \to ASBC, S \to ABC, AB \to AB', B'B \to B'B', CB \to BC, A \to a, B' \to b, C \to c\}$. Consider the following leftmost derivation mod $\hat{G}$:

$$\begin{array}{llllll} S & \Rightarrow_{\hat{G}} & ASBC & \Rightarrow_{\hat{G}} & aSBC & \Rightarrow_{\hat{G}} & aASBCBC \\ & \Rightarrow_{\hat{G}} & aaSBCBC & \Rightarrow_{\hat{G}} & aaABCBCBC & \Rightarrow_{\hat{G}} & aaAB'CBCBC \\ & \Rightarrow_{\hat{G}} & aaaB'CBCBC & \Rightarrow_{\hat{G}} & aaabCBCBC & \Rightarrow_{\hat{G}} & aaabBCCBC, \end{array}$$

to which no further leftmost derivation step applies. In fact, $L_{\mathsf{lm}}(\hat{G}) = \{abc\}$, while $L(\hat{G}) = \{\, a^n b^n c^n \mid n \geq 1 \,\}$.

Thus, although the grammars G, G', and $\hat{G}$ all generate the same language in unrestricted derivation mode, they behave very differently when restricted to leftmost mode.

In [Mat64] it is shown that the language $L_{\mathsf{lm}}(G)$ is context-free if $G = (V, \Sigma, S, P)$ is a phrase-structure grammar such that each rule $(\ell \to r) \in P$ has the structure

$$\ell = x_0 A_1 x_1 \cdots x_{n-1} A_n x_x \to x_0 \beta_1 x_2 \cdots x_{n-1} \beta_n x_n = r$$

for some $n \geq 1$, where $x_0, x_i \in \Sigma^*$, $A_i \in V$, and $\beta_i \in (V \cup \Sigma)^*$ for all $1 \leq i \leq n$ (see, e.g., [MaSa97], p. 198).

Below we will see that a corresponding result also holds in the setting of alternating grammars. To simplify the discussion we will assume in the following that, for an alternating phrase-structure grammar $G = (V, U, \Sigma, P, S)$, each rule $(\ell \to r) \in P$ has the form $\ell = xA\alpha \to x\beta = r$, where $x \in \Sigma^*$, $A \in V$, and $\alpha, \beta \in (V \cup \Sigma)^*$. Obviously, this restriction contains the above restriction as a special case. Also it is trivially satisfied by all grammars for which the lefthand side of each production begins with a nonterminal (or consists solely of nonterminals). Observe that neither the grammar G nor the grammar G' of Example 2.6 meet this restriction, but that the grammar $\hat{G}$ satisfies it.

We denote the class of languages generated by grammars of type X with respect to the leftmost derivation mode by $\mathcal{L}_{\mathsf{lm}}(\mathsf{X})$, while $\mathcal{L}(\mathsf{X})$ is used to denote the class of languages generated by grammars of type X in the unrestricted derivation mode.

In [MHHO] also a different type of alternating context-free grammars was introduced and studied. These are the so-called *state-alternating context-free grammars* (sACFG), which are obtained by combining the notion of *grammars with states* of Kasai [Kas70] with the notion of alternation. Analogously, we define the *state-alternating phrase-structure grammar* as an 8-tuple $G = (Q, U, V, \Sigma, P, S, q_0, F)$, where Q is a finite set of states, $U \subseteq Q$ is a set of *universal states*, while the states in $Q \setminus U$ are called *existential states*, V is a finite set of variables, Σ is a set of terminals, $S \in V$ is the start symbol, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is a set of final states. Finally, P is a finite set of productions of the form $(p, \ell) \to (q, r)$, where $p, q \in Q$, $\ell \in (V \cup \Sigma)^* \cdot V \cdot (V \cup \Sigma)^*$, and $r \in (V \cup \Sigma)^*$.

The *derivation relation* $\Rightarrow_G^*$ is defined on the set $Q \times (V \cup \Sigma)^*$ of *extended sentential forms*. Let $p \in Q$ and $\alpha \in (V \cup \Sigma)^*$. If p is an existential state, that is, $p \in Q \setminus U$, then $(p, \alpha) \Rightarrow_G (q, \alpha_1 r \alpha_2)$, if $\alpha = \alpha_1 \ell \alpha_2$, and there exists a production of the form $(p, \ell) \to (q, r)$. If p is a universal state, α has the factorization $\alpha = \alpha_1 \ell \alpha_2$, and $(p, \ell) \to (q_i, r_i)$ $(1 \leq i \leq k)$ are all the productions with lefthand side (p, ℓ), then

$$(p, \alpha) \Rightarrow_G ((q_1, \alpha_1 r_1 \alpha_2), \ldots, (q_k, \alpha_1 r_k \alpha_2)),$$

that is, all these productions are applied in parallel to the chosen occurrence of the substring ℓ, and following this step all these sentential forms are rewritten further, independently of each other. In this way a derivation tree is obtained.

The language $L(G)$ that is generated by G consists of all words $w \in \Sigma^*$ for which there exists a derivation tree such that the root is labelled with (q_0, S) and all leaves are labelled with pairs of the form (p, w) with $p \in F$. Note that the labels of different leaves may differ in their first components.

If $|\ell| \leq |r|$ holds for all productions $(p, \ell) \to (q, r)$ of P, then G is called a *state-alternating context-sensitive grammar*, and if $\ell \in V$ for all productions $(p, \ell) \to (q, r)$ of P, then G is a *state-alternating context-free grammar*. By sACFG, sACSG, and sAPSG we denote the classes of state-alternating context-free, context-sensitive, and general phrase-structure grammars, respectively. As before we are interested in the expressive power of these grammars with respect to the leftmost and the unrestricted derivation modes. It is known that the class

$\mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG})$ of languages that are generated by state-alternating context-free grammars in leftmost mode coincides with the class of languages that are accepted by alternating pushdown automata ([MHHO] Theorem 6.4).

3 On Alternating Grammars Working in Leftmost Mode

Here we will consider the generative power of alternating grammars with respect to the leftmost derivation mode. From the discussion on the leftmost derivation mode above we recall that we require in this section that each production of an alternating grammar is of the form $(xA\alpha \to x\beta)$, where $x \in \Sigma^*$, $A \in V$, and $\alpha, \beta \in (V \cup \Sigma)^*$. For state-alternating grammars, we require correspondingly that each production is of the form $((p, xA\alpha) \to (q, x\beta))$, where p and q are states.

Lemma 3.1. $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{sACSG})$.

Proof. Let $G = (V, U, \Sigma, P, S)$ be an ACSG that satisfies the above condition. We construct an sACSG $G' = (Q, Q^\forall, V, \Sigma, P', S, q_0, F)$ satisfying $L_{\mathsf{lm}}(G') = L_{\mathsf{lm}}(G)$. To this end we choose

$$\begin{array}{lcllcl} Q & := & \{\,[A] \mid A \in V\,\} \cup \{[?]\}, & Q^\forall & := & \{\,[A] \mid A \in U\,\}, \\ q_0 & := & [?], & F & := & \{[?]\}, \end{array}$$

and define P' as follows:

$$\begin{array}{lcll} ([?], A) & \to & ([A], A) & \text{for all } A \in V, \\ ([A], xA\alpha) & \to & ([?], x\beta) & \text{for all } (xA\alpha \to x\beta) \in P. \end{array}$$

Let $wA\gamma$ be a sentential form of G, where $w \in \Sigma^*$, $A \in V$, and $\gamma \in (V \cup \Sigma)^*$. Assume further that G contains the production $(xA\alpha \to x\beta)$, and that $w = w_0x$ and $\gamma = \alpha\delta$ hold. Then with respect to G, we have the leftmost derivation step $wA\gamma = w_0xA\alpha\delta \Rightarrow_G w_0x\beta\delta$. From the definition above we see that G' can execute the following leftmost derivation:

$$([?], wA\gamma) \Rightarrow_{G'} ([A], wA\gamma) = ([A], w_0xA\alpha\delta) \Rightarrow_{G'} ([?], w_0x\beta\delta).$$

Here the first step is existential, and the second step is existential if and only if the above step of G is existential. Thus, it follows immediately that the leftmost derivations mod G are in one-to-one correspondence to the leftmost derivations mod G'. Thus, $L_{\mathsf{lm}}(G') = L_{\mathsf{lm}}(G)$. □

By the same proof we obtain an analogous result for alternating phrase-structure grammars.

Lemma 3.2. $\mathcal{L}_{\mathsf{lm}}(\mathsf{APSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{sAPSG})$.

However, for APSGs we even have the following result.

Lemma 3.3. $\mathcal{L}_{\mathsf{lm}}(\mathsf{APSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG})$.

Proof. Let $G = (V, U, \Sigma, P, S)$ be an APSG, let $k := \max\{\,|\ell| \mid (\ell \to r) \in P\,\}$, and let $\overline{\Sigma} := \{\,\bar{a} \mid a \in \Sigma\,\}$ be a set of new variables that are in one-to-one correspondence to Σ. By $^- : (V \cup \Sigma)^* \to (V \cup \overline{\Sigma})^*$ we denote the morphism that is defined through $A \mapsto A$ for all $A \in V$

and $a \mapsto \bar{a}$ for all $a \in \Sigma$. We define an sACFG $G' := (Q', U', V', \Sigma, P', S', q_0', F')$ as follows, where $S', \$, \not\in$ are new variables, and $\hat{V} := \{\, yx\alpha \mid x, y \in \Sigma^*,\ |yx| \le k,\ \alpha \in (V \cup \Sigma)^*,\ |\alpha| \le k \,\}$:

$$
\begin{aligned}
&Q' := \{\, [yx\alpha], [y\#x\alpha] \mid yx\alpha \in \hat{V} \,\}, && V' := V \cup \overline{\Sigma} \cup \{S', \$, \not\in\}, \\
&U' := \{\, [y\#x\alpha] \mid yx\alpha \in \hat{V},\ \alpha \in U \cdot (V \cup \Sigma)^* \,\}, && F' := \{[\varepsilon]\}, \\
&q_0' := [\varepsilon], &&
\end{aligned}
$$

and P' consists of the following productions, where $a \in \Sigma$, $X \in V$, $A \in V \cup \overline{\Sigma}$, and $\alpha \in (V \cup \Sigma)^*$:

(0) $([\varepsilon], S') \to ([\varepsilon], S\not\in)$.

(1) (a) $([\alpha], X) \to ([\alpha X], \varepsilon)$, (b) $([\alpha], X) \to ([\alpha X], \$)$ for all $|\alpha| < 2k$.
(c) $([\alpha], \bar{a}) \to ([\alpha a], \varepsilon)$, (d) $([\alpha], \bar{a}) \to ([\alpha a], \$)$ for all $|\alpha| < 2k$.

(2) $([a\alpha], A) \to ([\alpha], aA)$ for all $|\alpha| < 2k$.

(3) $([y\#xA\alpha], \$) \to ([yx], \bar{\beta})$ for all $yx \in \Sigma^*$, $|yx| \le k$, if $(xA\alpha \to \beta) \in P$.

(4) $([\varepsilon], \bar{a}) \to ([\varepsilon], a)$.

(5) $([\varepsilon], \not\in) \to ([\varepsilon], \varepsilon)$.

(6) $([yx\alpha], \$) \to ([y\#x\alpha], \$)$ for all $yx\alpha \in \hat{V}$.

The states of G' of the form $[yx\alpha]$ and $[y\#x\alpha]$ are introduced to hold information on the portion $xA\alpha$ of a G-sentential form $yxA\alpha\gamma$ such that $xA\alpha$ can be the lefthand side of a G-production, where $y, x \in \Sigma^*$, $A \in V$, and $\alpha, \gamma \in (V \cup \Sigma)^*$. The first step of a G'-derivation is $([\varepsilon],\ S') \Rightarrow ([\varepsilon],\ S\not\in)$, and the $\not\in$-symbol will remain unchanged until the sentential form contains this symbol as the only variable left.

Let $xA\alpha \to x\beta$ be a production in P. Consider a leftmost G-derivation step $zyxA\alpha\gamma \Rightarrow zyx\beta\gamma$, where $zy \in \Sigma^*$, $|y| \le k - |x|$, and $\gamma \in (V \cup \Sigma)^*$. Assume that in G' we have already derived the sentential form $([yx],\ zA\bar{\alpha}\bar{\gamma}\not\in)$. Then we can execute the leftmost G'-derivation

$$
\begin{aligned}
([yx],\ zA\bar{\alpha}\bar{\gamma}\not\in) &\Rightarrow ([yxA],\ z\bar{\alpha}\bar{\gamma}\not\in) &&\Rightarrow^* ([yxA\alpha],\ z\$\bar{\gamma}\not\in) \\
&\Rightarrow ([y\#xA\alpha],\ z\$\bar{\gamma}\not\in) &&\Rightarrow ([yx],\ z\bar{\beta}\bar{\gamma}\not\in).
\end{aligned}
$$

Productions of type (1) choose nondeterministically either to further expand the substring α of the current sentential form $x\alpha\gamma$ obtained so far as a candidate for the lefthand side of a production to be applied to the sentential form or to terminate this process.

Note that $xA\alpha$ is a universal (existential, resp.) string for G if and only if $A \in U$ ($A \in V \setminus U$, resp.) if and only if $[y\#xA\alpha]$ is a universal (existential, resp.) state of G' for all $y \in \Sigma^*$ satisfying $|y| \le k - |x|$, and that by the definition of the productions of group (3), exactly one production $xA\alpha \to \beta$ of G corresponds to the productions of the form $([y\#xA\alpha], \$) \to ([yx], \bar{\beta})$ of G'. Thus, the above simulation of a G-derivation step by G' remains valid also if the G-derivation step is universal.

Next the derivation can proceed according to one of the following two cases.

Case 1: If $yx\beta\gamma = uvB\delta$ with $uv \in \Sigma^*$, $B \in V$, and $\delta \in (V \cup \Sigma)^*$, then there is a G'-derivation $([yx],\ z\bar{\beta}\bar{\gamma}\not\in) \Rightarrow^* ([v],\ zuB\bar{\delta}\not\in)$ by using productions of type (1), (2), and (4), which enables the application of a production for the next derivation step, whose lefthand side is of the form $v'B\delta_1$ for some suffix v' of v and a prefix δ_1 of δ.

Case 2: If $\beta\gamma$ is a terminal string, G' can complete the derivation by using productions of type (2) and (4) followed by production (5) to terminate the derivation: $([yx],\ z\bar{\beta}\bar{\gamma}\cent) \Rightarrow^* ([\varepsilon],\ zyx\bar{\beta}\bar{\gamma}\cent) \Rightarrow^* ([\varepsilon],\ zyx\beta\gamma\cent) \Rightarrow ([\varepsilon],\ zyx\beta\gamma)$.

By induction it follows that G' generates all the strings in $L_{\mathsf{lm}}(G)$ with respect to the leftmost derivation mode.

The converse inclusion can be proved similarly. The crucial points to note are the following:

1. The only universal states of G' are of the form $[y\#xA\alpha]$, where $A \in U$, and so $xA\alpha$ is a universal string for G if and only if $[y\#xA\alpha]$ is a universal state of G' for each $y \in \Sigma^*$ satisfying $|y| \leq k - |x|$.
2. The G'-productions $([y\#xA\alpha], \$) \to ([yx], \bar{\beta})$ of type (3), which are the only productions that rewrite sentential forms, correspond to the G-production $xA\alpha \to \beta$. □

Next we see that also the converse of Lemma 3.1 holds.

Lemma 3.4. $\mathcal{L}_{\mathsf{lm}}(\mathsf{sACSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG})$.

Proof. Let $G = (Q, U, V, \Sigma, P, q_0, S, F)$ be an sACSG. We now construct an ACSG $G' = (V', U', \Sigma, P', S')$ that generates the same language as G in leftmode mode. Let $\overline{\Sigma} := \{\, \bar{a} \mid a \in \Sigma \,\}$ be a set of new variables in one-to-one correspondence to Σ, and let $\bar{}\,: (V \cup \Sigma)^* \to (V \cup \overline{\Sigma})^*$ be the corresponding morphism. We choose

$$\begin{array}{lcl} V' & := & \{\, [q, A]_e, [q, A] \mid q \in Q,\ A \in V \cup \overline{\Sigma} \,\} \cup V \cup \overline{\Sigma}, \\ U' & := & \{\, [q, A] \mid q \in U,\ A \in V \,\}, \\ S' & := & [q_0, S], \end{array}$$

and we define P' to consist of the following productions, where $p, q \in Q$, $A, B \in V$, $a, b \in \Sigma$, $x, y \in \Sigma^*$, and $\alpha, \beta \in (V \cup \Sigma)^*$:

$$\begin{array}{lllll} (1) & [q, A]_e & \to & [q, A] & \text{for all } q \in Q \text{ and all } A \in V, \\ (2) & x[p, A]\bar{\alpha} & \to & xy[q, B]_e\bar{\beta} & \text{for all } ((p, xA\alpha) \to (q, xyB\beta)) \in P, \\ (3) & x[p, A]\bar{\alpha} & \to & x[q, \bar{a}]_e\bar{y} & \text{for all } ((p, xA\alpha) \to (q, xay)) \in P, \\ (4) & [q, \bar{a}]_e\bar{b} & \to & a[q, \bar{b}]_e & \text{for all } a, b \in \Sigma, \\ (5) & [q, \bar{a}]_e B & \to & a[q, B]_e & \text{for all } a \in \Sigma \text{ and all } B \in V, \\ (6) & [q, \bar{a}]_e & \to & a & \text{for all } q \in F \text{ and all } a \in \Sigma. \end{array}$$

As G is a context-sensitive grammar, G' is context-sensitive as well. Further, as G does not contain any ε-rules, the productions of G' of type (2) and (3) are in one-to-one correspondence to the productions of G. Actually, this correspondence respects the type of the productions of being existential or universal.

The basic idea of the simulation of G by G' is the following. The actual state of G is combined with the leftmost variable in the current sentential form. Thus, a sentential form $(p, uxA\alpha\gamma)$ of G, where $p \in Q$, $u, x \in \Sigma^*$, $A \in V$, and $\alpha, \gamma \in (V \cup \Sigma)^*$, is encoded by the sentential form $ux[p, A]_e\bar{\alpha}\bar{\gamma}$ of G'. Using the production $(p, xA\alpha) \to (q, xyB\beta)$, we can execute the leftmost derivation step $(p, uxA\alpha\gamma) \Rightarrow_G (q, uxyB\beta\gamma)$. In G' this is simulated by $ux[p, A]_e\bar{\alpha}\bar{\gamma} \Rightarrow_{G'} ux[p, A]\bar{\alpha}\bar{\gamma} \Rightarrow_{G'} uxy[q, B]_e\bar{\beta}\bar{\gamma}$. If a production of the form $(p, xA\alpha) \to (q, xay)$ is used, then we obtain the leftmost derivation step $(p, uxA\alpha\gamma) \Rightarrow_G (q, uxay\gamma)$, which is simulated in G' as follows:

$$ux[p, A]_e\bar{\alpha}\bar{\gamma} \Rightarrow_{G'} ux[p, A]\bar{\alpha}\bar{\gamma} \Rightarrow_{G'} ux[q, \bar{a}]_e\bar{y}\bar{\gamma} \Rightarrow^*_{G'} uxayz[q, B]_e\delta,$$

provided that $\gamma = zB\delta$ for some $z \in \Sigma^*$ and $B \in V$.

Note that in both cases, $(p, xA\alpha) \to (q, xyB\beta)$ (or $(p, xA\alpha) \to (q, xay)$) is a universal G-production if and only if $x[p, A]\bar{\alpha} \to xy[q, B]_e\bar{\beta}$ (or $x[p, A]\bar{\alpha} \to x[q, \bar{a}]_e\bar{y}$) is a universal G'-production Thus, the above G'-simulation of a leftmost G-derivation step is also valid, if the latter one is universal.

A successful leftmost derivation of G ends with a sentential form (q, w) for some $q \in F$ and $w \in \Sigma^+$, where in the last step the last occurring variable is replaced by a terminal string. In G' the corresponding leftmost derivation will yield a sentential form $ux[q, \bar{a}]_e\bar{y}\bar{b}$, where $w = uxayb$. However, using productions of type (4) and (6) we can complete the corresponding leftmost derivation of G' by $ux[q, \bar{a}]_e\bar{y}\bar{b} \Rightarrow^*_{G'} uxay[q, \bar{b}] \Rightarrow_{G'} uxayb = w$. It follows that $L_{\mathsf{lm}}(G') = L_{\mathsf{lm}}(G)$ holds. □

Combining Lemmas 3.1 and 3.4 we obtain the following equivalence.

Theorem 3.5. $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACSG})$.

The above proof of Lemma 3.4 can also be adapted to the case of alternating phrase-structure grammars. If G contains a production of the form $(p, xA\alpha) \to (q, x)$, then we add a new variable $[q, ?]_e$ to V' and the following non-monotone productions to P':

$$\begin{array}{lcll} x[p, A]\bar{\alpha} & \to & x[q, ?]_e, & \\ [q, ?]_e\bar{a} & \to & [q, \bar{a}]_e & \text{for all } a \in \Sigma, \\ [q, ?]_eB & \to & [q, B]_e & \text{for all } B \in V. \end{array}$$

Further, if q is a final state of G, then we also add the production $[q, ?]_e \to \varepsilon$. With these modifications we obtain the following result.

Lemma 3.6. $\mathcal{L}_{\mathsf{lm}}(\mathsf{sAPSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{APSG})$.

Combining Lemmas 3.3 and 3.6 with the trivial fact that $\mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{sAPSG})$ and the fact that $\mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG}) = \mathcal{L}(\mathsf{APDA})$ [MHHO], we have the following equivalence.

Theorem 3.7. $\mathcal{L}_{\mathsf{lm}}(\mathsf{APSG}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sAPSG}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG}) = \mathcal{L}(\mathsf{APDA})$.

As $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{APSG})$ holds, Theorem 3.5 and Theorem 3.7 yield the following consequence.

Corollary 3.8. $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACSG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG}) = \mathcal{L}(\mathsf{APDA})$.

Note that the grammar G' in the proof of Lemma 3.4 is not context-sensitive. At this moment we do not know whether or not the converse inclusion of Corollary 3.8 holds.

4 On Alternating Grammars Working in Unrestricted Mode

Above we have seen that with respect to the leftmost derivation mode ACSG and sACSG are equivalent, and also APSGand sAPSG are equivalent. Here we will prove that these equivalences also hold for the unrestricted derivation mode.

Lemma 4.1. $\mathcal{L}(\mathsf{ACSG}) \subseteq \mathcal{L}(\mathsf{sACSG})$.

Proof. Let $G = (V, U, \Sigma, P, S)$ be an ACSG. Similar to the proof of Lemma 3.1 we construct an sACSG $G' = (Q, Q^\forall, V', \Sigma, P', S', q_0, F)$ satisfying $L(G') = L(G)$. However, here we do not make any assumptions on the form of the productions of G. We take

$$\begin{array}{lcllcl} Q & := & \{\,[A] \mid A \in V\,\} \cup \{[?]\}, & Q^\forall & := & \{\,[A] \mid A \in U\,\}, \\ q_0 & := & [?], & F & := & \{[?]\}, \\ V' & := & V, & S' & := & S, \end{array}$$

and define P' as follows, where $x \in \Sigma^*$, $A \in V$, and $\alpha, \beta \in (V \cup \Sigma)^*$:

$$\begin{array}{lcll} ([?], A) & \to & ([A], A) & \text{for all } A \in V, \\ ([A], xA\alpha) & \to & ([?], \beta) & \text{for all } (xA\alpha \to \beta) \in P. \end{array}$$

Let $\omega A\gamma$ be a sentential form of G, where $A \in V$, and $\omega, \gamma \in (V \cup \Sigma)^*$. Assume further that G contains the production $(xA\alpha \to \beta)$, and that $\omega = \omega_1 x$ and $\gamma = \alpha\delta$ hold. Then with respect to G, we have the derivation step $\omega A\gamma = \omega_1 xA\alpha\delta \Rightarrow_G \omega_1\beta\delta$. From the definition above we see that G' can execute the following derivation:

$$([?], \omega A\gamma) \Rightarrow_{G'} ([A], \omega A\gamma) = ([A], \omega_1 xA\alpha\delta) \Rightarrow_{G'} ([?], \omega_1\beta\delta).$$

Here the first step is existential, and the second step is existential if and only if the above step of G is existential. Thus, it follows immediately that the derivations mod G are in one-to-one correspondence to the derivations mod G'. Thus, $L(G') = L(G)$. □

The same proof applies to alternating phrase-structure grammars, that is, we also have the following inclusion.

Lemma 4.2. $\mathcal{L}(\mathsf{APSG}) \subseteq \mathcal{L}(\mathsf{sAPSG})$.

For establishing the converse inclusions we modify the proof of Lemma 3.4.

Lemma 4.3. $\mathcal{L}(\mathsf{sACSG}) \subseteq \mathcal{L}(\mathsf{ACSG})$.

Proof. Let $G = (Q, U, V, \Sigma, P, q_0, S, F)$ be an sACSG. From G we construct an ACSG $G' = (V', U', \Sigma, P', S')$ that generates the same language as G. We choose

$$\begin{array}{lcl} V' & := & \{\,[q, A]_e, [q, A] \mid q \in Q,\ A \in V \cup \Sigma\,\} \cup V \cup \Sigma, \\ U' & := & \{\,[q, A] \mid q \in U,\ A \in V\,\}, \\ S' & := & [q_0, S], \end{array}$$

and we define P' to consist of the following productions, where $p, q \in Q$, $A \in V$, $x \in \Sigma^*$, $X \in V \cup \Sigma$, and $\alpha, \beta \in (V \cup \Sigma)^*$:

$$\begin{array}{llcll} (1) & [q, A]_e & \to & [q, A] & \text{for all } q \in Q \text{ and all } A \in V, \\ (2) & x[p, A]\alpha & \to & [q, X]_e\beta & \text{for all } ((p, xA\alpha) \to (q, X\beta)) \in P, \\ (3) & [q, X]_e Y & \to & X[q, Y]_e & \text{for all } X, Y \in V \cup \Sigma, \\ (4) & X[q, Y]_e & \to & [q, X]_e Y & \text{for all } X, Y \in V \cup \Sigma, \\ (5) & [q, a]_e & \to & a & \text{for all } q \in F \text{ and all } a \in \Sigma. \end{array}$$

As G is a context-sensitive grammar, G' is context-sensitive as well. Further, as G does not contain any ε-rules, the productions of G' of type (2) are in one-to-one correspondence to the productions of G. Actually, this correspondence respects the type of the productions of being existential or universal.

The basic idea of the simulation of G by G' is the following. The actual state of G is combined with a variable in the current sentential form. Thus, a sentential form $(p, \omega x A \alpha \gamma)$ of G, where $p \in Q$, $x \in \Sigma^*$, $A \in V$, and $\omega, \alpha, \gamma \in (V \cup \Sigma)^*$, is encoded by the sentential form $\omega x[p, A]_e \alpha\gamma$ of G'. Using the production $(p, xA\alpha) \to (q, X\beta)$, we can execute the derivation step $(p, \omega x A \alpha\gamma) \Rightarrow_G (q, \omega X \beta\gamma)$. In G' this is simulated as follows:

$$\omega x[p, A]_e \alpha\gamma \Rightarrow_{G'} \omega x[p, A]\alpha\gamma \Rightarrow_{G'} \omega[q, X]_e \beta\gamma.$$

By the productions of type (3) and (4) the new state symbol q can then be moved to the occurrence of the leftmost variable in the lefthand side of the rule that is to be applied next. A successful derivation of G ends with a sentential form (q, w) for some $q \in F$ and $w \in \Sigma^+$, where in the last step the last occurring variable is replaced by a terminal string. In G' the corresponding derivation will yield a sentential form $u[q, a]_e y$, where $w = uay$. However, using a production of type (5) we can complete the corresponding derivation of G'. It follows that $L(G') = L(G)$ holds. □

In the proof above, if G contains ε-rules, then we can modify the construction of G' accordingly to derive the following result.

Lemma 4.4. $\mathcal{L}(\mathsf{sAPSG}) \subseteq \mathcal{L}(\mathsf{APSG})$.

Thus, we have the following equalities, where RE denotes the class of recursively enumerable languages. Observe that each phrase-structure grammar can be interpreted as an alternating phrase-structure grammar without universal states. On the other hand, it is obvious that the derivation process of an alternating phrase-structure grammar can be simulated by an alternating Turing machine. Hence, we see that in unrestricted mode alternating phrase-structure grammars have exactly the same expressive power as (non-alternating) phrase-structure grammars.

Corollary 4.5. (a) $\mathcal{L}(\mathsf{ACSG}) = \mathcal{L}(\mathsf{sACSG})$.
(b) $\mathcal{L}(\mathsf{APSG}) = \mathcal{L}(\mathsf{sAPSG}) = \mathsf{RE}$.

5 ACSGs and Alternating Linear Bounded Automata

An *alternating linear bounded automaton*, ALBA for short, M is given through a 9-tuple $M = (Q, Q^\forall, \Sigma, \Gamma, ¢, \$, \delta, q_0, F)$, where Q is a finite set of states, $Q^\forall \subseteq Q$ is a set of universal states, Σ is a finite input alphabet, Γ is a finite tape alphabet that includes Σ, the symbols $¢, \$ \notin \Gamma$ are the left and right delimiters for the work space, respectively, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is a set of accepting (or final) states, and

$$\delta : Q \times (\Gamma \cup \{¢, \$\}) \to \mathcal{P}(Q \times (\Gamma \cup \{¢, \$\}) \times \{-1, 0, 1\})$$

is a transition relation.

A *configuration* of M is given through a string of the form $¢uqav\$$, where $q \in Q$, $a \in \Gamma$, and $u, v \in \Gamma^*$. Here q is the current state of M, uav is the current tape inscription, and the head of M is currently scanning the tape cell containing the distinguished occurrence of the letter a.

Assume that $\delta(q, a) = \{(q_1, b_1, \mu_1), \ldots, (q_n, b_n, \mu_n)\}$. It is required that $b_i \in \Gamma$ $(1 \leq i \leq n)$, if $a \in \Gamma$, and that $b_i = a$ $(1 \leq i \leq n)$, if $a = ¢$ or $a = \$$. Further, for all $i = 1, \ldots, n$, $\mu_i \neq -1$, if $a = ¢$, and $\mu_i \neq 1$, if $a = \$$. Thus, M's head does never leave the section of the tape containing the content $¢uav\$$, and the delimiters can neither be erased nor can new

occurrences of them be created. If state q is existential, that is, $q \in Q \setminus Q^{\forall}$, then a triple (q_i, b_i, μ_i) is chosen nondeterministically, and M executes the corresponding transformation. If state q is universal, then M applies all transformation steps given by (q_i, b_i, μ_i) , $1 \leq i \leq n$, simultaneously, which yields n successor configurations. In this way, a computation of M can be seen as a tree the nodes of which are labelled by configurations.

The initial configuration for an input $w \in \Sigma^*$ has the form $q_0 \mathbin{¢} w\$$, that is, M is in its initial state, scanning the left delimiter $¢$, with tape inscription w. The word w is *accepted* by M, if there exists a finite computation tree of M the root of which is labelled with the initial configuration $q_0 \mathbin{¢} w\$$, and all leaves are labelled with accepting configurations, that is, with configurations in which M is in an accepting state. By $L(M)$ we denote the language consisting of all words that are accepted by M.

It is known that $\mathcal{L}(\mathsf{ALBA}) = \mathcal{L}(\mathsf{APDA}) = \bigcup_{c>0} \mathsf{DTIME}(c^n)$ [CKS81, LLS84], and that $\mathcal{L}(\mathsf{ALBA}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG})$ [MHHO]. Here we will show that $\mathcal{L}(\mathsf{ALBA})$ also coincides with the class of languages that are generated by ACSGs working in the unrestricted derivation mode. The next lemma shows that ACSGs are of sufficient expressive power to generate all languages that are accepted by alternating linear bounded automata.

Lemma 5.1. $\mathcal{L}(\mathsf{ALBA}) \subseteq \mathcal{L}(\mathsf{sACSG})$.

Proof. Let $M = (Q, Q^{\forall}, \Sigma, \Gamma, ¢, \$, \delta, q_0, F)$ be an ALBA. Obviously we can assume that M halts immediately when it enters an accepting state, that is, M has no transitions that it could apply when it is in an accepting state.

We will now construct an sACSG $G = (Q^{(G)}, U, V, \Sigma, P, S, q_0^{(G)}, F^{(G)})$ that generates the language $L(M)$. In order to simplify the construction we assume that $|w| \geq 2$ holds for each word $w \in L(M)$. The finitely many short words excluded by this assumption can be handled by introducing corresponding additional start rules into G.

The idea of the construction of G is the same as that underlying the classical construction of a monotone grammar for simulating a linear bounded automaton (see, e.g., [HoUl79] Theorem 9.6). Accordingly, we choose

$$
\begin{array}{lcl}
Q^{(G)} & := & Q \cup \{\mathsf{Start}, \mathsf{Fin}\}, \\
U & := & Q^{\forall}, \\
V & := & \{S, T\} \cup \{\, [a, b], [¢a, ¢b], [a\$, b\$] \mid a \in \Sigma,\ b \in \Gamma \,\} \\
 & & \cup \{\, [a, qb], [¢a, q¢b], [¢a, ¢qb], [a\$, qb\$], [a\$, bq\$] \mid a \in \Sigma,\ b \in \Gamma,\ q \in Q \,\}, \\
q_0^{(G)} & := & \mathsf{Start}, \\
F^{(G)} & := & \{\mathsf{Fin}\}.
\end{array}
$$

Here the variables of the form $[a, b]$ will be used to describe a tape cell that contained the letter a at the start of M, but that now contains the letter b, the variables of the form $[a, qb]$ will be used analogously with the additional information that M is currently in state q scanning the symbol b. The variables of the form $[¢a, ¢b]$ and $[a\$, b\$]$ are used to describe the left and the right end of the tape inscription, respectively, and the variables of the form $[¢a, q¢b], [¢a, ¢qb], [a\$, qb\$], [a\$, bq\$]$ include in addition information on the current state and head position.

It remains to describe the productions P of G. They consist of several groups:

(I) The productions of this group are used to generate an encoding of an initial configura-

tion of M:

$$\begin{array}{lrcll}
(1) & (\mathsf{Start}, S) & \to & (\mathsf{Start}, [\mathfrak{c}a, q_0\mathfrak{c}a]T) & \text{for all } a \in \Sigma, \\
(2) & (\mathsf{Start}, T) & \to & (\mathsf{Start}, [a, a]T) & \text{for all } a \in \Sigma, \\
(3) & (\mathsf{Start}, T) & \to & (q_0, [a\$, a\$]) & \text{for all } a \in \Sigma.
\end{array}$$

Using these productions G can generate a pair of the form

$$(q_0, \alpha_0) := (q_0, [\mathfrak{c}a_1, q_0\mathfrak{c}a_1][a_2, a_2] \cdots [a_n\$, a_n\$])$$

for any $n \geq 2$ and $a_1, \ldots, a_n \in \Sigma$. Observe that $q_0\mathfrak{c}a_1 \cdots a_n\$$ is just the initial configuration of M for the input $w := a_1 \cdots a_n$.

(II) This group of rules will be used to simulate a computation of M starting from the initial configuration encoded by the string α_0, where $a_1, a_2 \in \Sigma$, $b_1, b_2, c \in \Gamma$, $p, q \in Q$, and $\alpha \in \Sigma \cup (\mathfrak{c} \cdot \Sigma)$, $\beta \in \Gamma \cup (\mathfrak{c} \cdot \Gamma)$, $\gamma \in (\Sigma \cdot \$) \cup \Sigma$, $\delta \in (\Gamma \cdot \$) \cup \Gamma$:

$$\begin{array}{lrcll}
(1) & (p, [a_1, b_1][a_2, pb_2][\gamma, \delta]) & \to & (q, [a_1, qb_1][a_2, c][\gamma, \delta]) & \text{if } (q, c, -1) \in \delta(p, b_2), \\
(2) & (p, [\mathfrak{c}a_1, \mathfrak{c}b_1][a_2, pb_2][\gamma, \delta]) & \to & (q, [\mathfrak{c}a_1, \mathfrak{c}qb_1][a_2, c][\gamma, \delta]) & \text{if } (q, c, -1) \in \delta(p, b_2), \\
(3) & (p, [\alpha, \beta][a_2, pb_2][\gamma, \delta]) & \to & (q, [\alpha, \beta][a_2, qc][\gamma, \delta]) & \text{if } (q, c, 0) \in \delta(p, b_2), \\
(4) & (p, [\alpha, \beta][a_2, pb_2][\gamma, \delta]) & \to & (q, [\alpha, \beta][a_2, c][\gamma, q\delta]) & \text{if } (q, c, 1) \in \delta(p, b_2), \\
(5) & (p, [\mathfrak{c}a_1, \mathfrak{c}pb_1][\gamma, \delta]) & \to & (q, [\mathfrak{c}a_1, q\mathfrak{c}c][\gamma, \delta]) & \text{if } (q, c, -1) \in \delta(p, b_1), \\
(6) & (p, [\mathfrak{c}a_1, \mathfrak{c}pb_1][\gamma, \delta]) & \to & (q, [\mathfrak{c}a_1, \mathfrak{c}qc][\gamma, \delta]) & \text{if } (q, c, 0) \in \delta(p, b_1), \\
(7) & (p, [\mathfrak{c}a_1, \mathfrak{c}pb_1][\gamma, \delta]) & \to & (q, [\mathfrak{c}a_1, \mathfrak{c}c][\gamma, q\delta]) & \text{if } (q, c, 1) \in \delta(p, b_1), \\
(8) & (p, [\mathfrak{c}a_1, p\mathfrak{c}b_1]) & \to & (q, [\mathfrak{c}a_1, q\mathfrak{c}b_1]) & \text{if } (q, \mathfrak{c}, 0) \in \delta(p, \mathfrak{c}), \\
(9) & (p, [\mathfrak{c}a_1, p\mathfrak{c}b_1]) & \to & (q, [\mathfrak{c}a_1, \mathfrak{c}qb_1]) & \text{if } (q, \mathfrak{c}, 1) \in \delta(p, \mathfrak{c}), \\
(10) & (p, [a_1, b_1][a_2\$, pb_2\$]) & \to & (q, [a_1, qb_1][a_2\$, c\$]) & \text{if } (q, c, -1) \in \delta(p, b_2), \\
(11) & (p, [\mathfrak{c}a_1, \mathfrak{c}b_1][a_2\$, pb_2\$]) & \to & (q, [\mathfrak{c}a_1, \mathfrak{c}qb_1][a_2\$, c\$]) & \text{if } (q, c, -1) \in \delta(p, b_2), \\
(12) & (p, [a_1, b_1][a_2\$, pb_2\$]) & \to & (q, [a_1, b_1][a_2\$, qc\$]) & \text{if } (q, c, 0) \in \delta(p, b_2), \\
(13) & (p, [a_1, b_1][a_2\$, pb_2\$]) & \to & (q, [a_1, b_1][a_2\$, cq\$]) & \text{if } (q, c, 1) \in \delta(p, b_2), \\
(14) & (p, [a_1\$, b_1p\$]) & \to & (q, [a_1\$, qb_1\$]) & \text{if } (q, \$, -1) \in \delta(p, \$), \\
(15) & (p, [a_1\$, b_1p\$]) & \to & (q, [a_1\$, b_1q\$]) & \text{if } (q, \$, 0) \in \delta(p, \$).
\end{array}$$

On the second components of the symbols in the string $[\mathfrak{c}a_1, q_0\mathfrak{c}a_1][a_2, a_2] \cdots [a_n\$, a_n\$]$ the above productions of G simulate a computation of M starting from the initial configuration $q_0\mathfrak{c}a_1 \cdots a_n\$$. Let $\mathfrak{c}uapbv\$$ ($u, v \in \Gamma^*$, $a, b \in \Gamma$, $p \in Q$) be a configuration of M during that computation, and assume that $\delta(p, b) = \{(q_1, c_1, \mu_1), \ldots, (q_m, c_m, \mu_m)\}$. If p is an existential state, and if alternative i is chosen, then

$$\mathfrak{c}uapbv\$ \vdash_M \begin{cases} \mathfrak{c}uq_iac_iv\$ & \text{if } \mu_i = -1, \\ \mathfrak{c}uaq_ic_iv\$ & \text{if } \mu_i = 0, \\ \mathfrak{c}uac_iq_iv\$ & \text{if } \mu_i = 1. \end{cases}$$

Corresponding to the configuration $\mathfrak{c}uapbv\$$ of M, we have the string

$$\alpha := [\mathfrak{c}a_1, \mathfrak{c}u_1] \cdots [a_{|u|+1}, a][a_{|u|+2}, pb][a_{|u|+3}, v_1] \cdots [a_n\$, v_{|v|}\$],$$

where u_1 is the first letter of u, and v_1 and $v_{|v|}$ are the first and the last letter of v, respectively. From the definition of the productions of G we see that

$$(p,\alpha) \Rightarrow_G \begin{cases} (q, [\mathrm{¢}a_1, \mathrm{¢}u_1]\cdots[a_{|u|+1}, q_i a][a_{|u|+2}, c_i][a_{|u|+3}, v_1]\cdots[a_n\$, v_{|v|}\$]) \text{ if } \mu_i = -1, \\ (q, [\mathrm{¢}a_1, \mathrm{¢}u_1]\cdots[a_{|u|+1}, a][a_{|u|+2}, q_i c_i][a_{|u|+3}, v_1]\cdots[a_n\$, v_{|v|}\$]) \text{ if } \mu_i = 0, \\ (q, [\mathrm{¢}a_1, \mathrm{¢}u_1]\cdots[a_{|u|+1}, a][a_{|u|+2}, c_i][a_{|u|+3}, q_i v_1]\cdots[a_n\$, v_{|v|}\$]) \text{ if } \mu_i = 1. \end{cases}$$

If p is a universal state, then the transititons $(p, b) \to (q_i, c_i, \mu_i)$, $1 \leq i \leq m$, are all applied simultaneously. From the definition of G, we see that the corresponding rules are included in P, and as the state p is also universal for G, all these rules are applied simultaneously as well. It follows by induction that corresponding to each accepting computation tree of M on input w, G generates a derivation tree the leaves of which are labelled with pairs of the form $(q, [\mathrm{¢}a_1, \mathrm{¢}b_1][a_2, b_2]\cdots[a_j, qb_j]\cdots[a_n\$, b_n\$])$ with $q \in F$.

(III) Finally we need some productions that allow us to convert an encoding of a pair of the form above into the string w. For that we introduce the following productions:

(1)	$(q, [a, qb])$	$\to$	(Fin, a)	for all $a \in \Sigma$, $b \in \Gamma$, and $q \in F$,
(2)	$(q, [\mathrm{¢}a, \mathrm{¢}qb])$	$\to$	(Fin, a)	for all $a \in \Sigma$, $b \in \Gamma$, and $q \in F$,
(3)	$(q, [\mathrm{¢}a, q\mathrm{¢}b])$	$\to$	(Fin, a)	for all $a \in \Sigma$, $b \in \Gamma$, and $q \in F$,
(4)	$(q, [a\$, qb\$])$	$\to$	(Fin, a)	for all $a \in \Sigma$, $b \in \Gamma$, and $q \in F$,
(5)	$(q, [a\$, bq\$])$	$\to$	(Fin, a)	for all $a \in \Sigma$, $b \in \Gamma$, and $q \in F$,
(6)	$(\mathsf{Fin}, c[a, b])$	$\to$	(Fin, ca)	for all $a, c \in \Sigma$, and $b \in \Gamma$,
(7)	$(\mathsf{Fin}, c[a\$, b\$])$	$\to$	(Fin, ca)	for all $a, c \in \Sigma$, and $b \in \Gamma$,
(8)	$(\mathsf{Fin}, [a, b]c)$	$\to$	(Fin, ac)	for all $a, c \in \Sigma$, and $b \in \Gamma$,
(9)	$(\mathsf{Fin}, [\mathrm{¢}a, \mathrm{¢}b]c)$	$\to$	(Fin, ac)	for all $a, c \in \Sigma$, and $b \in \Gamma$.

As M halts immediately on reaching an accepting state, we see that there are no productions in group (II) that G might apply when it has reached an accepting state. Thus, in this situation G can only continue by applying the corresponding production from group (III). But from then on, no productions from group (I) or group (II) can ever be applied again during this derivation, and we see from the form of the productions of group (III) that each pair of the form $(q, [\mathrm{¢}a_1, \mathrm{¢}b_1][a_2, b_2]\cdots[a_j, qb_j]\cdots[a_n\$, b_n\$])$ with $q \in F$ derives the pair $(\mathsf{Fin}, a_1 \cdots a_n)$.

From the discussion above it follows that $L(M) \subseteq L(G)$ holds. On the other hand, each successful derivation of G must be of the form described above, and so it generates a string that is accepted by M. Hence, we see that $L(G) = L(M)$ holds. □

Also we have the converse of Lemma 5.1.

Lemma 5.2. $\mathcal{L}(\mathsf{sACSG}) \subseteq \mathcal{L}(\mathsf{ALBA})$.

Proof. Let $G = (Q, U, V, \Sigma, P, S, q_0, F)$ be an sACSG. We can construct an ALBA M as follows. This construction is similar to the standard construction of a linear bounded automaton from a monotone grammar (see, e.g., [HoU179]).

First M divides its tape into two tracks. The given input is retained on the first track, while on the second track a G-derivation is being simulated. During this simulation M keeps

the current state of G as part of its own state. For simulating a step $(p, \alpha\ell\beta) \Rightarrow_G (q, \alpha r\beta)$, M first guesses the factor ℓ and an occurrence of ℓ that is to be replaced. Assume that $(p, \ell) \to (q_1, r_1), \ldots, (p, \ell) \to (q_m, r_m)$ are all the productions of G with lefthand side (p, ℓ). If p is an existential state, then M nondeterministically chooses an index $i \in \{1, \ldots, m\}$ and replaces the chosen occurrence of ℓ by the string r_i, changing the state of G that it stores in its finite control from p to q_i. If p is a universal state, then M executes a universal step, replacing the chosen occurrence of ℓ by all strings $r_1, \ldots, r_m$ simultaneously, for each value of i changing the state of G it stores internally to the corresponding state q_i. Of course, M halts and rejects if any of the derivation steps of G being simulated tries to exceed the space marked by the given input.

Whenever a branch of the derivation of G being simulated finishes with a pair of the form (p, u) such that $p \in F$ and $u \in \Sigma^*$, then M compares u to the input w stored on the first track. If u and w coincide, then this branch of M's computation halts and accepts; otherwise, it halts but rejects.

Now $w \in L(G)$, if and only if there is a finite derivation tree of G such that the root is labelled with the pair (q_0, S), and each leaf is labelled with a pair of the form (q, w) for some $q \in F$. From the above description of M we see that this is the case if and only if there exists a finite computation tree of M that, starting from the initial configuration on input w, accepts on all its branches. Hence, we see that $L(M) = L(G)$ holds. □

Thus, we obtain the following theorem.

Theorem 5.3. $\mathcal{L}(\mathsf{sACSG}) = \mathcal{L}(\mathsf{ALBA})$.

Because of Corollary 4.5 (a) and the fact that $\mathcal{L}(\mathsf{ALBA}) = \mathcal{L}(\mathsf{APDA}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG})$, this yields the following consequence.

Corollary 5.4. $\mathcal{L}(\mathsf{ACSG}) = \mathcal{L}(\mathsf{sACSG}) = \mathcal{L}(\mathsf{ALBA}) = \mathcal{L}(\mathsf{APDA}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG})$.

From Corollaries 3.8 and 5.4, the following inclusion follows.

Corollary 5.5. $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) \subseteq \mathcal{L}(\mathsf{ACSG})$.

It remains to consider the converse of the above inclusion. By Corollary 5.4 this is equivalent to the question of whether the inclusion $\mathcal{L}_{\mathsf{lm}}(\mathsf{sACFG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG})$ holds. As each ε-free sACFG is context-sensitive, we have at least the following special case.

Corollary 5.6. $\mathcal{L}_{\mathsf{lm}}(\varepsilon\text{-free sACFG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG})$.

Let k be a positive integer. We say that an sACFG G is *k-ε-bounded* if, for each $w \in L_{\mathsf{lm}}(G)$, there exists a leftmost G-derivation tree for w such that the number of applications of ε-productions on each path of this derivation tree is bounded from above by the number $k \cdot |w|$. G is *ε-bounded* if there exists a constant k such that G is k-ε-bounded. Can the above result be extended to ε-bounded sACFGs?

Conjecture. $\mathcal{L}_{\mathsf{lm}}(\varepsilon\text{-bounded sACFG}) \subseteq \mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG})$.

6 Concluding Remarks

We have generalized the notion of alternation from context-free to non-context-free grammars. In fact, we have considered two different types of alternation, namely alternation

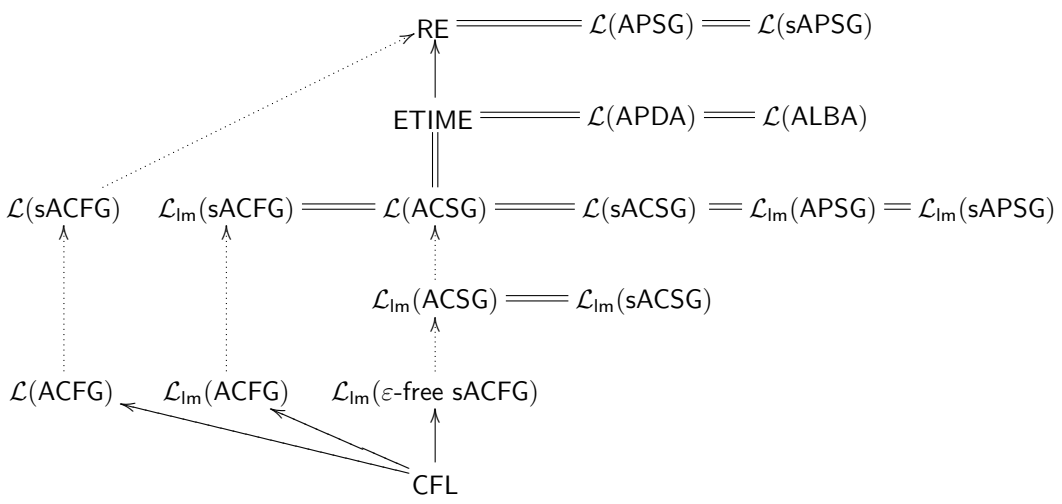

Figure 1: Inclusions relations between language classes defined by various types of alternating grammars. An arrow denotes a proper inclusion, while a dotted arrow denotes an inclusion that is not known to be proper.

based on distinguishing between existential and universal variables, and alternation based on states. In contrast to the situation for context-free grammars we have been able to show that, for general phrase-structure grammars as well as for context-sensitive grammars, both types of alternation are equivalent. This holds for the leftmost derivation mode as well as for the unrestricted derivation mode. Our main result shows that with respect to the leftmost derivation mode alternating phrase-structure grammars are just as expressive as state-alternating context-free grammars, and that alternating context-sensitive grammars working in the unrestricted derivation mode have the same expressive power, too. In this way we have obtained new grammar-based characterizations for the class of languages that are accepted by alternating pushdown automata. The diagram in Figure 1 summarizes the inclusion relations among the classes of languages we have discussed here. However, an important relation concerning alternating context-sensitive grammars remains open.

Open Problem 1. *Does* $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG}) = \mathcal{L}(\mathsf{ACSG})$ *hold?*

Observe that the corresponding problem is also still open for alternating context-free grammars. In the light of our results this problem can be expressed as follows.

Open Problem 2. *Does* $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACFG}) = \mathcal{L}_{\mathsf{lm}}(\mathsf{APSG})$ *hold?*

This can be seen as the counterpart for alternating grammars to Matthews' result that in leftmost mode phrase-structure grammars only generate context-free languages [Mat64]. Another obvious problem is the following.

Open Problem 3. *Is* $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACFG})$ *contained in* $\mathcal{L}_{\mathsf{lm}}(\mathsf{ACSG})$*?*

Finally, there are other derivation modes like the *leftish mode* that we have not considered in this paper. A derivation with respect to a context-free grammar is called *leftish* if in each step the leftmost occurrence of a lefthand side in the current sentential form is being rewritten. With respect to this mode state-alternating context-free grammars can generate all recursively enumerable languages, while for (variable-)alternating context-free grammars the

leftish derivation mode is just as expressive as the leftmost mode [MHHO]. How to define the leftish derivation mode for (state-) alternating context-sensitive and general phrase-structure grammars? What is the expressive power of these types of grammars under this mode?

Also it remains to study alternating versions of growing context-sensitive (that is, strictly monotone) grammars [DaWa86, BuOt98] and their relationship to the shrinking alternating two-pushdown automaton studied by the authors in [OtMo04].

Acknowledgement. The first author was supported in part by Waseda University Grant for Special Research Projects #2006B-073, which he gratefully acknowledges. The authors thank Hartmut Messerschmidt from Universität Kassel for fruitful discussions on the notions and results presented in this paper.

References

[BuOt98] G. Buntrock and F. Otto. Growing context-sensitive languages and Church-Rosser languages. *Information and Computation*, 141: 1–36, 1998.

[CKS81] A.K. Chandra, D.C. Kozen, and L.J. Stockmeyer. Alternation. *Journal of the Assoc. for Comput. Mach.*, 28: 114–133, 1981.

[ChSt76] A.K. Chandra and L.J. Stockmeyer. Alternation. In: *Proceedings of the 17th FOCS*, pages 98–108. IEEE Computer Society Press, 1976.

[DaWa86] E. Dahlhaus and M. Warmuth. Membership for growing context-sensitive grammars is polynomial. *Journal of Computer and System Sciences*, 33: 456–472, 1986.

[HoUl79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, Reading, M.A., 1979.

[IJW92] O.H. Ibarra, T. Jiang, and H. Wang. A characterization of exponential-time languages by alternating context-free grammars. *Theoretical Computer Science*, 99: 301–313, 1992.

[Kas70] T. Kasai. An infinite hierarchy between context-free and context-sensitive languages. *Journal of Computer and System Sciences*, 4: 492–508, 1970.

[LLS78] R.E. Ladner, R.J. Lipton, and L.J. Stockmeyer. Alternating pushdown automata. In: *Proceedings of the 19th FOCS*, pages 92–106. IEEE Computer Society Press, 1978.

[LLS84] R.E. Ladner, R.J. Lipton, and L.J. Stockmeyer. Alternating pushdown and stack automata. *SIAM Journal on Computing*, 13: 135–155, 1984.

[MaSa97] A. Mateescu and A. Salomaa. Aspects of classical language theory. In: G. Rozenberg and A. Salomaa (eds.), *Handbook of Formal Languages*, *Vol. 1: Word, Language, Grammar*, pages 175–251. Springer, Berlin, 1997.

[Mat64] G. Matthews. A note on symmetry in phrase structure grammars. *Information and Control*, 7: 360–365, 1964.

[Mor89] E. Moriya. A grammatical characterization of alternating pushdown automata. *Theoretical Computer Science*, 67: 75–85, 1989.

[MHHO] E. Moriya, D. Hofbauer, M. Huber, and F. Otto. On state-alternating context-free grammars. *Theoretical Computer Science*, 337: 183–216, 2005.

[MoNa97] E. Moriya and S. Nakayama. Grammatical characterizations of alternating pushdown automata and linear bounded automata. *Gakujutsu Kenkyu*, Series of Math, 45: 13–24, School of Education, Waseda Univ., 1997 (in Japanese).

[MoOt07] E. Moriya and F. Otto. Two ways of introducing alternation into context-free grammars and pushdown automata. *The Transactions of the IEICE*, E 90-D: 889–894, 2007.

[OtMo04] F. Otto and E. Moriya. Shrinking alternating two-pushdown automata. *The Transactions of the IEICE*, E 87-D: 959–966, 2004.