J Logic Lang Inf (2006) 15:233–250
DOI 10.1007/s10849-006-9016-z

ORIGINAL PAPER

A descriptive characterisation of linear languages

Tore Langholm

Received: 17 March 2004 / Accepted: 7 February 2006

Abstract Lautemann et al. (1995) gave a descriptive characterisation of the class of context-free languages, showing that a language is context-free iff it is definable as the set of words satisfying some sentence of a particular logic (fragment) over words. The present notes discuss how to specialise this result to the class of linear languages. Somewhat surprisingly, what would seem the most straightforward specialisation actually fails, due to the fact that linear grammars fail to admit a Greibach normal form. We identify an alternative specialisation, based on an alternative characterisation of context-free languages, also noted by Lautemann et al. (1995).

Keywords Descriptive complexity, Linear languages, Greibach normal form

1. Introduction

The model-theoretic approach to linguistic theory seeks to analyse membership conditions of formal languages using methods of mathematical logic. To this end, a finite word over a given alphabet is identified with a relational structure or "model" over a corresponding signature. Sentences of various logics appropriate for such structures then express membership conditions of words relative to languages. One aim of the programme is to compare different language classes in terms of the logics necessary and sufficient to define the languages of the classes. We refer to Thomas (1997) for a brief overview. These notes identify a logic fragment characterising the class LIN of linear context-free languages. Frequent reference will be made to the corresponding characterisation of the full class CFL of context-free languages given by Lautemann et al. (1995).

For any finite word w over a fixed, finite alphabet Σ, let the corresponding *word model* $\underline{w}$ be the relational structure

$$\big(dom(w), <^w, \big(Q_a^w\big)_{a\in\Sigma}\big),$$

T. Langholm (✉)
University of Oslo, Department of Informatics, P.O. Box 1080 Blindern, 0316 Oslo, Norway
e-mail: torel@ifi.uio.no

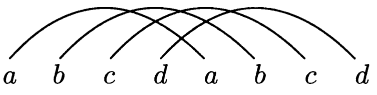

Fig. 1 Binary relation violating the axiom *UNCROSS*

where $dom(w) = \{0, \ldots, |w| - 1\}$. Here, $|w|$ is the length of w. $<^w$ is the natural order on $dom(w)$, while each Q_a^w collects the positions of w containing the symbol a, i.e., $Q_a^w = \{i \in dom(w) \mid a_i = a\}$ when $w = a_0 \ldots a_n$. Now the empty word determines a structure with empty domain, which is often disallowed in other contexts but admitted here.

To talk about such structures a first-order logic is used, containing the binary relation symbol $<$ and unary predicate symbols Q_a, in each structure $\underline{w}$ interpreted by the corresponding relations displayed above, as well as Boolean connectives and individual variables (semantically ranging over positions) with existential and universal quantifiers. Hence atomic formulas are of the forms $Q_a x$ or $x < y$, where x, y are individual variables. For further details we refer to Thomas (1997). Occasionally we shall also use the "atomic formulas" $x \leq y, x = y$ and Sxy (the latter expressing immediate succession) which are shorthand for the corresponding, more complex expressions involving $<$.[1]

A sentence φ now *defines* the language $L(\varphi) = \{w \in \Sigma^* \mid \underline{w} \models \varphi\}$. Using sentences of the first-order logic above, exactly the subclass of regular languages known as the star-free languages are definable. Passing from first-order logic to monadic second-order logic, exactly the regular languages are definable. To go beyond regular languages, one can add binary relation variables, with matching quantifiers. This, however, results in an expressive power far beyond context-free grammars. The following example is particularly revealing for present purposes.[2]

Example 1. The non-context-free[3] copy-language $\{ww \mid w \in \Sigma^*\}$ is defined by the sentence $\exists N\varphi$, where φ is the conjunction of the following.

$$\forall x \exists y(Nxy \vee Nyx)$$
$$\forall x_1 x_2 y_1 y_2(Nx_1y_1 \wedge Nx_2y_2 \rightarrow x_1 < y_2)$$
$$\forall x_1 x_2 y_1 y_2(Nx_1y_1 \wedge Nx_2y_2 \rightarrow (x_1 < x_2 \leftrightarrow y_1 < y_2))$$
$$\forall xy \bigwedge_{a \in \Sigma}(Nxy \rightarrow (Q_a x \leftrightarrow Q_a y))$$

When $w = a_0 \ldots a_{2n-1}$, then the first three conjuncts of φ uniquely identify N as the relation $\{(i, i+n) \mid 0 \leq i < n\}$ (when w is of odd length, no relation satisfies the conditions) whereas the last conjunct of φ says that each position in the left half holds the same symbol as its "copy" in the right half. Figure 1 shows the word *abcdabcd* with the binary relation from φ envisaged as a set of arcs.

[1] They are definable as $\neg y < x$, $\neg(x < y \vee y < x)$ and $x < y \wedge \neg\exists z(x < z \wedge z < y)$, respectively.

[2] It is not equally revealing for other purposes. The copy language, although not context-free, has a low recognition complexity. It is, however, possible to write a sentence of the form $\exists N\varphi$, with binary N as the only second-order variable, that defines a language with an NP-hard membership problem, see Theorem 10.1 of Eiter et al. (2000). In fact, the proof given there can easily be modified to yield such a sentence of the form $\exists N(\mathit{FORWARD} \wedge \varphi)$. *FORWARD* is a condition on N introduced shortly below.

[3] It is non-context-free provided $|\Sigma| > 1$.

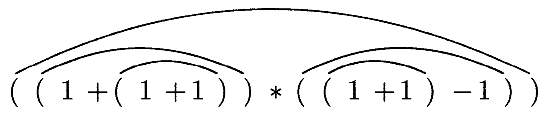

Fig. 2 Matching: relation satisfying the axioms *FORWARD*, *UNCROSS* and*STEEP*

The second conjunct of φ above implies the following somewhat weaker axiom stating that N relates "earlier" positions to "later" ones:

$$\textit{FORWARD}\quad \forall xy(Nxy \rightarrow x < y)$$

A relation satisfying this axiom can (without loss of information) be envisaged as a set of undirected arcs as in Figure 1. Crucial to the example above is the "cross-serial" behaviour of the binary relation; the N-arcs connecting each symbol in the left half to its copy in the right half all cross in a systematic manner. The axiom below expressly forbids such crossing.

$$\textit{UNCROSS}\quad \forall x_1x_2y_1y_2(Nx_1y_1 \wedge Nx_2y_2 \wedge x_1 < x_2 \wedge y_1 < y_2 \rightarrow y_1 < x_2)$$

We write *NEST* for the conjunction of *FORWARD* and *UNCROSS*, and say that the binary relation N is a *nesting* if it satisfies *NEST*.

Simple arithmetic expressions built up from 1, parentheses and binary infix operators $+, *, -, /$ form a typical context-free language. Nestings can be used to define this language, using an arc between two positions to encode that they hold the left and right parentheses of a matching pair, cf. Figure 2. The nestings used for such encodings are, however, of a particular type, being partial functions in either direction, and thus satisfying the next axiom as well.

$$\textit{STEEP}\quad \forall x_1x_2y_1y_2(Nx_1y_1 \wedge Nx_2y_2 \rightarrow (x_1 = x_2 \leftrightarrow y_1 = y_2))$$

We write *MATCH* for the conjunction of all three axioms *FORWARD*, *UNCROSS* and *STEEP*, and say (with Lautemann et al. (1995)[4]) that a binary relation N is a *matching* if it satisfies *MATCH*. Now we have the following.

Theorem 1. *(Lautemann et al. 1995). A language over a finite alphabet Σ is context-free iff it equals $L(\exists N(\textit{MATCH} \wedge \varphi))$ for some first-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

The authors remark that their proof of the above theorem can be modified to prove that CFL is also characterised by general nestings in a similar way,[5] but provide no further details. In Appendix A we show how to prove Theorem 2 directly from Theorem 1.

Theorem 2. *A language over a finite alphabet Σ is context-free iff it equals $L(\exists N(\textit{NEST} \wedge \varphi))$ for some first-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

The relation between the two theorems is linked to the concepts below.

[4] The axioms given are different, but are, in combination, equivalent to the these three.

[5] They do not use the term "nesting" and in fact do not explicitly mention anything corresponding to the axiom *FORWARD* in this context, but it is clear from what is stated that this axiom must be assumed.

Definition 1. A context-free grammar (V, Σ, P, S) is in *Greibach normal form* if the right-hand side of every production is in $\Sigma(\Sigma \cup V)^*$, i.e., if it starts with a terminal, and in *strong Greibach normal form* if each right-hand side is in ΣV^*. Moreover, a grammar is in *double Greibach normal form* if each right-hand side is in $\Sigma \cup (\Sigma(\Sigma \cup V)^* \Sigma)$, i.e., if it both starts and ends with a terminal, and in *strong double Greibach normal form* if each right-hand side is in $\Sigma \cup (\Sigma V^* \Sigma)$.

Any context-free language without ϵ can be generated by a grammar in any of these forms, see Greibach (1965) and Rosenkrantz(1967). The strong and regular versions are obviously equivalent for general context-free grammars,[6] but we shall see below that all forms are in fact non-equivalent for linear grammars.

The fact that CFL is characterised both by the more (nestings) and less (matchings) general type of relation, is intimately connected to the fact that context-free grammars admit the double Greibach normal form. Loosely put, matchings characterise the languages generated by context-free grammars in double Greibach normal form, while nestings characterise context-free languages in general. Due to the normal form theorem, the distinction disappears. It emerges, however, in the context of linear languages.

Definition 2. A context-free grammar is *linear* if the right-hand side of each production contains at most one variable. A language is *linear* if it is generated by a linear context-free grammar.

Applied to linear grammars, all four conditions listed in Definition 1 yield different language classes, and all are strict subclasses of LIN. Linear grammars in these forms shall be referred to as *Greibach linear*, etc., and generate *Greibach linear languages*, etc.

Example 2. When S is the start symbol, each set of productions below generates the corresponding language.

sGl	$\{S \to aS, S \to aI, I \to bI, I \to b\}$	$\{a^n b^m \mid n, m > 0\}$
sdGl	$\{S \to aSb, S \to ab\}$	$\{a^n b^n \mid n > 0\}$
dGl	$\{S \to aaSb, S \to aab\}$	$\{a^{2n} b^n \mid n > 0\}$
Gl	$\{S \to aS, S \to aI, I \to bIc, I \to bc\}$	$\{a^n b^m c^m \mid n, m > 0\}$
l	$\{S \to Sc, S \to Ic, I \to aIb, I \to ab\}$	$\{a^n b^n c^m \mid n, m > 0\}$

Grammars in the Greibach normal forms contain no ϵ-productions and thus generate languages without the empty word, but apart from this fact the strong Greibach linear languages are precisely the regular languages while the strong double Greibach linear languages are precisely the so-called even linear languages (Amar and Putzolu, 1964). Again allowing for this exception, it is also seen directly that the class of double Greibach linear languages is the closure of the even linear languages under ϵ-free homomorphisms.

The codes to the left in Example 2 indicate the type of the grammar. The last grammar is not Greibach linear, and $\{a^n b^n c^m \mid n, m > 0\}$ is in fact not a Greibach linear language; we refer

[6] Replace $A \to abCdE$ with $A \to aBCDE$, $B \to b$, $D \to d$ to obtain a strong Greibach grammar from a regular one. Similarly for the double-sided version. The original left-sided version of Greibach (1965) was given in the strong form, while the double-sided version of Rosenkrantz was given in the regular form. The choice of form is a matter of taste only in the general setting, but not so in the context of linear grammars.

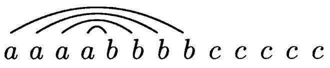

Fig. 3 Linear matching satisfying the conditions in Example 3

to Appendix B for a pumping lemma for Greibach linear languages that spoils this language. The obvious strengthening of this pumping lemma to the double Greibach linear languages moreover spoils the Greibach linear language $\{a^n b^m c^m \mid n, m > 0\}$, hence these two classes are distinct. As $\{a^{2n} b^n \mid n > 0\}$ is not an even linear language (we refer to Langholm and Bezem (2003) for a suitable pumping lemma spoiling this language), it follows that all five classes are distinct.

For the characterisation of linear languages we introduce the axiom below, and say that a nesting N is *linear* if it satisfies this axiom.

$$\textit{LINEAR} \quad \forall x_1 x_2 y_1 y_2 (N x_1 y_1 \wedge N x_2 y_2 \rightarrow x_1 < y_2)$$

The axiom says that all start points precede all endpoints, hence in particular no arc fully precedes another arc. Given the condition *UNCROSS*, this means that all arcs are wrapped inside each other in a single "well" structure. Hence we use the term *well* to refer to linear nestings, i.e., relations satisfying the three[7] axioms *FORWARD, UNCROSS* and *LINEAR*.

Example 3. The last language of Example 2, $\{a^n b^n c^m \mid n, m > 0\}$, is defined by the sentence $\exists N(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi)$, where φ is the conjunction of the four sentences below.

$$\exists xyz(Q_a x \wedge Q_b y \wedge Q_c z)$$
$$\forall xyz(Q_a x \wedge Q_b y \wedge Q_c z \rightarrow x < y \wedge y < z)$$
$$\forall xy(Nxy \rightarrow Q_a x \wedge Q_b y)$$
$$\forall x(Q_a x \vee Q_b x \rightarrow \exists y(Nxy \vee Nyx))$$

An example word model (and matching) is seen in Figure 3.
This example cannot be generalised directly to a theorem, however; the class LIN is *not* characterised by linear matchings in the way that CFL is characterised by matchings in Theorem 1. In Appendix B we shall see that the linear language of the next example (like its close relative last in Example 2) is not Greibach linear, and prove moreover that it is not defined by any sentence $\exists N(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi)$, where φ is a first-order sentence over signature $(N, <, (Q_a)_{a \in \Sigma})$.

Example 4. The language $\{a^n b^n c^{2m} \mid n, m > 0\}$ over the alphabet $\Sigma = \{a, b, c\}$ is linear, and defined by $\exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi)$, where φ is the conjunction of the first-order[8] sentences below.

$$\exists xyz(Q_a x \wedge Q_b y \wedge Q_c z)$$

[7] *LINEAR* of course implies *FORWARD*, so in fact the two axioms *UNCROSS* and *LINEAR* suffice.

[8] The unique existential quantifier $\exists!$ is defined from ordinary $\exists$ and $\forall$, Boolean connectives and $=$ in the usual way.

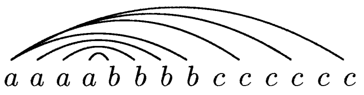

Fig. 4 Linear nesting satisfying the conditions in Example 4

$$\forall xyz(Q_a x \wedge Q_b y \wedge Q_c z \to x < y \wedge y < z)$$

$$\forall x(Q_a x \to \exists! y(Q_b y \wedge Nxy))$$

$$\forall y(Q_b y \to \exists! x(Q_a x \wedge Nxy))$$

$$\forall xy(Sxy \wedge Q_c x \wedge Q_c y \to ((\exists z Nzx) \leftrightarrow \neg(\exists z Nzy)))$$

$$\forall xyz(Nxz \wedge Syz \wedge Q_c z \to Q_c y)$$

$$\forall x \exists y(x < y \vee Nyx)$$

These sentences say that all *as*, of which there is at least one, come first, all *bs* (also at least one) come in the middle and all *cs* (again at least one) last, that N connects each a to exactly one b and vice versa, and finally (in the last three sentences) that every second c, excluding the first and including the last, are endpoints of N-arcs.

The corresponding start point will, by necessity, in all cases be the first a, hence this N will not be a matching, cf. Figure 4. This definition of the language in terms of linear nestings thus makes essential use of the fact that linear nestings in general need not be linear matchings. An attempt to define the same language in terms of linear matchings immediately runs into trouble: one now seems to need the *entire* linear matching in order to ensure an equal number of *as* and *bs*, and hence is left with only first-order logic to ensure an even number of *cs*. This latter task is impossible, as the language $\{c^{2n} \mid n > 0\}$ is not star-free, and thus not definable by a first-order sentence over the signature $(Q_c, <)$. Appendix B contains a proof along these lines.

In Sections 3 and 4 we prove the general result that LIN is characterised by linear nestings in the same way that CFL, in Theorem 2, is characterised by nestings. Section 2 describes a correspondence between linear nestings (i.e., "wells") and "caterpillars," which are essentially the shape of the parse trees for linear grammars. This correspondence plays a central role in Sections 3 and 4.

2. Wells and dense wells

Say that one pair (i, j) in a nesting N *wraps around* another pair (k, l) if the two are distinct (i.e., either $i \neq k$ or $j \neq l$, or both) and $i \leq k < l \leq j$. In this case we write $(i, j) \prec_N (k, l)$, and say that the wider pair *precedes* the narrower. By the non-crossing condition on nestings, if $(i, j) \prec_N (k, l)$ then (k, l) has a unique immediate predecessor, i.e., a unique narrowest pair in the nesting that wraps around it.

Say that a position i *occurs inside* the pair (j, k) if $j \leq i \leq k$, and that i *belongs to* (j, k) if (j, k) is the last (narrowest) pair in the nesting inside which i occurs. Then every pair in the nesting will have at least one position that belongs to it, and every position will belong to at most one pair in the nesting. A *wide nesting* on $w = a_0 \ldots a_n$ is a nesting that contains $(0, n)$, i.e., a nesting satisfying the next axiom.

$$\textit{WIDE} \quad \exists xy(Nxy \wedge \neg\exists z(z < x \vee y < z))$$

Hence in wide nestings every position will belong to exactly one pair.

In general, each pair in a nesting may have several immediate successors, thus in general $\prec_N$ is not a linear order. In fact, the relation $\prec_N$ of a nesting N is a linear order precisely iff N is linear.

Definition 3. Let w be a word; a *caterpillar* on w is an ordered tree with yield w, in which every internal node has at least two children and no internal node has more than one child that is also internal.

We shall remain at this level of formalisation, and will not spell things out further in terms of tree domains or the like. Note that the terminal nodes of a caterpillar are decorated with elements of Σ, while internal nodes are not decorated. Observe also that for any word $w \in \Sigma^*$ of length at least 2 there is a 1-1-correspondence between wide wells and caterpillars on w:

– When the wide well N on $w = a_0 \dots a_n$ is given, the tree (in fact caterpillar) t_N is constructed as follows: the leaves are $0, \dots, n$, and the internal nodes are the members of N. Each leaf i is decorated with a_i.$(0, n)$ is the root. The children of a pair will be the positions that belong to it as well as the successor pair, if there is one, in the obvious sequence which respects the ordering of positions.
– Conversely, given a caterpillar t on w, let N_t be the wide well

$$\{(ll_t(z), rl_t(z)) \mid z \text{ is an internal node}\}, \tag{1}$$

where $ll_t(z)$ is the leftmost leaf dominated by z (i.e., the leaf at the end of the path from z that repeatedly selects leftmost children) and $rl_t(z)$ similarly is the rightmost leaf dominated by z.

It is easy to check that the two mappings are inverses of each other, i.e., that $N_{t_N} = N$ and that t_{N_t} is isomorphic to t.

Definition 4. A *dense well* on w is a wide well on w that satisfies the following additional requirement: for any positions i, j of w, if $N(i, j)$ and $i + 1 < j$, then either $N(i + 1, j)$ or $N(i, j - 1)$.

Hence a well is dense iff it satisfies the axiom *WIDE* and the next axiom.

$$\textit{DENSE} \quad \forall x_1 x_2 y_1 y_2 (N x_1 y_2 \wedge S x_1 x_2 \wedge S y_1 y_2 \rightarrow x_1 = y_1 \vee N x_1 y_1 \vee N x_2 y_2)$$

Note that by *UNCROSS* we never have *both* $N(i + 1, j)$ and $N(i, j - 1)$, hence in a dense well *exactly* one of the two will hold whenever $N(i, j)$ and $i + 1 < j$.

Lemma 1. *Let w be a word of length at least 2. Every well on w can be extended to a dense well on w.*

Proof: Assume that $w = a_0 \dots a_n$, where $n > 0$. We show (1) that $N \cup \{(0, n)\}$ is a well if N is, and (2) that for any positions i, j of w, if $i + 1 < j$, N is a well and $N(i, j)$, then either

$N \cup \{(i+1, j)\}$ or $N \cup \{(i, j-1)\}$ is a well also. As w is finite, this gives a process that eventually stops.

(1) is immediate and is left to the reader. To prove (2), suppose $N(i, j)$, where $i + 1 < j$. If (i, j) has no successor in N, then $N \cup \{(i+1, j)\}$ (the other choice is equally good) is a well. If (i, j) has the successor (k, l), then either $k = i$, in which case $N \cup \{(i, j-1)\}$ is a well, or $k > i$, in which case $N \cup \{(i+1, j)\}$ is a well. □

Definition 5. A binary caterpillar on w is a caterpillar on w in which every internal node has exactly two children.

Observe that if N is a wide well then N is dense iff t_N is a binary caterpillar. Conversely, if t is a caterpillar with at least one internal node, then N_t is dense iff t is binary.

Finally we also consider binary caterpillars with decorations on the internal nodes. Decorations taken from a set of two possible choices will suffice. The two choices are denoted $\otimes$ and $\odot$; this notation is borrowed from Lautemann et al. (1995), who use it for similar purposes.

Definition 6. A *Boolean binary caterpillar* on w is a binary caterpillar on w in which each internal node is decorated with either $\otimes$ or $\odot$. A *complemental well-pair* on w is a pair (N, M) of wells on w, such that $N \cap M = \emptyset$, and $N \cup M$ is a dense well.

Note that an alternative formulation of Lemma 1 would say that every well is the first component of some complemental well-pair.

We now observe that for any word $w \in \Sigma^*$ of length at least 2 there is a 1-1-correspondence between Boolean binary caterpillars and complemental well-pairs on w.

– When the complemental well-pair (N, M) is given, the Boolean binary caterpillar $T_{N,M}$ is constructed by taking the binary caterpillar $t_{N \cup M}$ and decorating each internal node (i, j) with $\otimes$ if $N(i, j)$, and with $\odot$ if $M(i, j)$.
– Conversely, given a Boolean binary caterpillar T let N_T and M_T of the corresponding complemental well-pair (N_T, M_T) be

$$\{(ll_T(z), rl_T(z)) \mid z \text{ is a node decorated with } \otimes\}$$

and

$$\{(ll_T(z), rl_T(z)) \mid z \text{ is a node decorated with } \odot\},$$

respectively, where $ll_T(z)$ and $rl_T(z)$ are defined exactly as $ll_t(z)$ and $rl_t(z)$. The two mappings are easily seen to be inverses of each other.

3. Encoding of grammars

Say that a context-free grammar (V, Σ, P, S) is in *weak Greibach normal form* if the right-hand sides of all productions are in $(\Sigma(V \cup \Sigma)^*) \cup ((V \cup \Sigma)^*\Sigma)$, i.e., in every case there is a terminal symbol either first or last. Hence a linear grammar is in weak Greibach normal form if it contains no ϵ-productions and no productions with a single variable on the right-hand side. It is straightforward to see that any linear language without ϵ can be generated by a linear grammar in weak Greibach normal form. Hence the following can also be obtained.

Lemma 2. *Let L be a linear language without ϵ. Then L is generated by a linear grammar in weak Greibach normal form, with the following property. For any two nonterminal productions $A \to uBv$ and $A' \to u'B'v'$, if $u = u'$ and $v = v'$ then $A = A'$.*

For a proof, we refer to the construction used to prove Lemma 2.1.2 of Lautemann et al. (1995), and note that this construction preserves linear grammars. For languages containing only words of length at least 2, we also note the following useful variation.

Lemma 3. *Let L be a linear language containing only words of length at least 2. Then L is generated by a linear grammar with the following properties. The right-hand side of each production has length at least 2, and for any two nonterminal productions $A \to uBv$ and $A' \to u'B'v'$, if $u = u'$ and $v = v'$ then $A = A'$.*

Proof: Suppose $L \subseteq \Sigma\Sigma\Sigma^*$, and apply Lemma 2. We need to eliminate productions with right-hand sides of length 1. As the grammar is in weak Greibach normal form and $L \subseteq \Sigma\Sigma\Sigma^*$, such productions are of the type $A \to a$, with a a terminal and A distinct from the start symbol. Remove each such production $A \to a$, simultaneously adding $B \to uav$ for each production $B \to uAv$. As no new nonterminal productions are added, the uniqueness property of the lemma is preserved. $\square$

Before stating the main lemma of the section, we introduce some useful notation and terminology. First define

$$\overline{N}xy \quad \text{as} \quad x \leq y \wedge \neg\exists zw(x < z \wedge z < y \wedge (Nzw \vee Nwz)),$$

$$succ(x, x', y', y) \quad \text{as} \quad Nxy \wedge Nx'y' \wedge \overline{N}xx' \wedge \overline{N}y'y \wedge \neg(x = x' \wedge y = y'),$$

$$last(x, y) \quad \text{as} \quad Nxy \wedge \neg\exists x'y' succ(x, x', y', y).$$

Hence $\overline{N}xy$ says that $x \leq y$, and that no position strictly between the two is involved in N. Assuming N is a well, $succ(x, x', y'.y)$ says that the pair (x', y') is the successor to the pair (x, y), while $last(x, y)$ says that (x, y) is the last pair of N. Furthermore, for any $w \in \Sigma^*$ let the two formulas $\overrightarrow{\psi_w}(x, y)$ and $\overleftarrow{\psi_w}(y, x)$ be defined by the following recursions.

$$\begin{aligned}
\overrightarrow{\psi_\epsilon}(x, y) &= \overleftarrow{\psi_\epsilon}(y, x) = (x = y) \\
\overrightarrow{\psi_{aw}}(x, y) &= Q_a(x) \wedge \exists z(Sxz \wedge \overrightarrow{\psi_w}(z, y)) \\
\overleftarrow{\psi_{wa}}(y, x) &= Q_a(x) \wedge \exists z(Szx \wedge \overleftarrow{\psi_w}(y, z)) \\
\psi_{ua}(x, y) &= \overrightarrow{\psi_u}(x, y) \wedge Q_a(y)
\end{aligned}$$

Hence $\overrightarrow{\psi_w}(x, y)$ says that $x \leq y$, and that the positions from and including x until and not including y contain the word w. Similarly, $\overleftarrow{\psi_w}(x, y)$ says that $x \leq y$, and that the positions from and not including x until and including y contain the word w. The last equation defines, for any $w \in \Sigma^+$, the formula $\psi_w(x, y)$, saying that $x \leq y$ and that the positions from and including x until and including y contain w.

Lemma 4. *Every linear language L over alphabet Σ is definable as $L(\exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi_L))$ for some first-order sentence φ_L over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

Proof: Without loss of generality, we may assume L to contain only words of length at least 2.[9] Now $L = L(G)$ for some linear grammar $G = (V, \Sigma, P, S)$ of the form described in Lemma 3, and let P_0 and P_1 be the subsets of P containing the productions without and with a variable occurring on the right-hand side, respectively. Then for any productions $p_0 = A \to u \in P_0$ and $p_1 = A \to uBv \in P_1$ define

$$\psi_{p_0}(x, y) \text{ as } last(x, y) \wedge \psi_u(x, y), \quad \text{and}$$
$$\psi_{p_1}(x, y, z, w) \text{ as } succ(x, y, z, w) \wedge \overrightarrow{\psi_u}(x, y) \wedge \overleftarrow{\psi_v}(z, w).$$

Next for any $A \in V$ let $\psi_A(x, y)$ be

$$\bigvee_{A\to\alpha\in P_0} \psi_{A\to\alpha}(x, y) \vee \bigvee_{A\to\alpha\in P_1} \exists x'y'\psi_{A\to\alpha}(x, x', y', y),$$

and finally for any production $p = A \to uBv \in P_1$ let $\overline{\psi}_p(x, y, z, w)$ be $\psi_p(x, y, z, w) \wedge \psi_B(y, z)$. Then let φ_G be the conjunction of the two sentences below.

$$\exists xy(\psi_S(x, y) \wedge \neg\exists z(z < x \vee y < z))$$
$$\forall xyzw(succ(x, y, z, w) \to \bigvee_{p\in P_1} \bar{\psi}_p(x, y, z, w)).$$

We argue that $L(G) = \{w \in \Sigma^* \mid \underline{w} \models \exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi_G)\}$.

First suppose $w \in L(G)$, i.e., there is some legal parse tree T according to G, with yield w. By the form of G (i.e., the facts that G is linear and that every right-hand side has length at least 2) T defines a caterpillar t on w when decorations on internal nodes are ignored. As $|\, w\,| \geq 2$, N_t is a wide well, and it is easily verified that $(\underline{w}, N_t) \models \varphi_G$ holds.

Conversely, suppose $(\underline{w}, N) \models \varphi_G$ for $w = a_0 \ldots a_n$, where N is a well on w. φ_G implies that N is wide, so there exists a corresponding caterpillar t_N. From this we define a legal parse tree according to G by the addition of decorations to internal nodes of t_N. This is done by an induction going from top to bottom, which in each step decorates the next node (i, j) with some A for which $(\underline{w}, N, i, j) \models \psi_A(x, y)$ holds.

First, $(0, n)$ is decorated with S. By the first conjunct of φ_G we have $(\underline{w}, N, 0, n) \models \psi_S(x, y)$.

Now suppose $(i, j) \in N$ has already been decorated with A, that $(\underline{w}, N, i, j) \models \psi_A(x, y)$ holds, and that (k, l) is the successor to (i, j). By the definition of ψ_A we know that

$$(\underline{w}, N, i, k, l, j) \models \psi_{A\to uBv}(x, z, w, y)$$

[9] Any linear language L over Σ can be formed as the union of the star-free (as it is finite) language $L_0 = L \cap (\epsilon \cup \Sigma)$ and the linear (as LIN is a full trio, see Hopcroft and Ullman (1979)) language $L_1 = L \cap (\Sigma\Sigma\Sigma^*)$, which contains only words of length at least 2. Now by the characterisation of star-free languages by first-order logic (see Theorem 4.4 of Thomas (1997)) $L_0 = \{w \mid \underline{w} \models \varphi_0\}$ for some first-order sentence φ_0 over signature $(<, (Q_a)_{a\in\Sigma})$, and assuming $L_1 = \{w \mid \underline{w} \models \exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi_1)\}$ for a first-order sentence φ_1 over signature $(N, <, (Q_a)_{a\in\Sigma})$, the identity $L = \{w \mid \underline{w} \models \exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge (\varphi_0 \vee \varphi_1))\}$ follows by prenex laws.

holds for some $A \to uBv \in P_1$, and by the second conjunct of φ_G we also know that

$$(\underline{w}, N, i, k, l, j) \models \overline{\psi}_{A' \to u'B'v'}(x, z, w, y)$$

holds for some $A' \to u'B'v' \in P_1$. Now this is only possible if $u = u'$ and $v = v'$, and hence, by the form of G, if $A = A'$. Then decorate (k, l) with B'. The local tree involving (i, j) and (k, l) is then sanctioned by the production $A \to uB'v$, and from the definition of $\overline{\psi}_p$ we know that $(\underline{w}, N, k, l) \models \psi_{B'}(x, y)$ holds.

Finally, if (i, j) has been decorated with A and it has no successor, and $(\underline{w}, N, i, j) \models \psi_A(x, y)$ holds, then clearly $(\underline{w}, N, i, j) \models \psi_{A \to u}(x, y)$ for some $A \to u \in P_0$, which now sanctions the local tree rooted in (i, j). □

4. Encoding by grammars

To prove the reverse of Lemma 4 we need to consider a monadic second-order logic over trees, e.g. as described by Gécseg and Steinby (1997). For this purpose we declare $\otimes$ and $\odot$ to be symbols of rank 2; this means that for any Σ the $\{\otimes, \odot\}\Sigma$-trees are the ordered, binary trees with leaves decorated by elements of Σ and internal nodes decorated by $\otimes$ or $\odot$. The monadic second-order logic described by Gécseg and Steinby (1997) contains primitives which (using the monadic second-order logic machinery) allows the definition of the following relations:

$Q_a(x)$ saying that node x is a leaf decorated with a
$Q_\otimes(x)$ saying that x is an internal node decorated with $\otimes$
$Q_\odot(x)$ saying that x is an internal node decorated with $\odot$
$c(x, y)$ saying that y is a child of x
$lc(x, y)$ saying that y is the left child of x
$rc(x, y)$ saying that y is the right child of x
$leaf(x)$ saying that x is a leaf
$ll(x, y)$ saying that y is the leftmost leaf dominated by x
$rl(x, y)$ saying that y is the rightmost leaf dominated by x
$x < y$ saying that x is left of y in the inorder traversal of the tree.[10]

Now let *CAT* be the axiom

$$\forall xyz(c(x, y) \wedge c(x, z) \wedge y \neq z \to (leaf(y) \vee leaf(z)))\ ,$$

saying that at most one child of any internal node is itself internal. Any $\{\otimes, \odot\}\Sigma$-tree satisfying this is a Boolean binary caterpillar.

Lemma 5. *Let Σ be finite, and let φ be a monadic second-order sentence over signature $(N, <, (Q_a)_{a \in \Sigma})$. Then $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$ is a linear language.*

[10] Note that the primitive symbol < provided by Gécseg and Steinby (1997) denotes an entirely different ordering relation, but the one described here is definable from (this and) other primitives in the logic.

Proof: Let φ be as in the lemma. Without loss of generality we may assume that $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$ only contains words of length at least 2.[11]

Now let the sentence φ' of the monadic second-order tree logic be obtained from φ by restricting all first-order quantification to leaves and all second-order quantification to sets of leaves, and by replacing $x < y$ with its definition in the tree-logic, and finally by replacing $N(x, y)$ with $\exists z(Q_{\otimes}(z) \wedge ll(z, x) \wedge rl(z, y))$. Then $(\underline{w}, N) \models \varphi$ iff N is the first component of some complemental well-pair (N, M) on w such that $T_{N,M} \models \varphi'$. Hence the following are all equal.

$L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$
$\{w \mid T_{N,M} \models \varphi'$ for some complemental well pair (N, M) on $w\}$
$\{w \mid T \models \varphi'$ for some Boolean binary caterpillar T on $w\}$
$yield[\{T \mid T$ is a Boolean binary caterpillar such that $T \models \varphi'\}]$
$yield[\{T \in \{\otimes, \odot\}\Sigma \mid T \models \varphi' \wedge \textsc{cat}\}]$.

Now by Propositions 12.2 and 6.2 of Gécseg and Steinby (1997) it follows that $\{T \in \{\otimes, \odot\}\Sigma \mid T \models \varphi' \wedge \textsc{cat}\}$ is a regular tree language.
Proposition 14.3 of Gécseg and Steinby (1997) furthermore states that the yield of any regular tree language is a context-free language. This is not helpful for our purposes, but an inspection of the proof reveals that the following strengthening is true as well.

Proposition 1. *Let Σ be an alphabet and let Π be a ranked alphabet such that $\Sigma \cap \Pi = \emptyset$. If L is a regular tree language over $\Pi\Sigma$ then there exists a finite number of context-free grammars $G_1, \ldots, G_n$, where $G_i = (V_i, \Sigma, P_i, S_i)$, and mappings $h_i : V_i \to \Pi$ such that $T \in L$ iff there is an i and a legal parse tree T_i according to G_i, such that* T *is the result of applying h_i to the decorations of nonterminal nodes*[12] *of T_i.*

Now substituting $\{\otimes, \odot\}$ for Π and $\{T \in \{\otimes, \odot\}\Sigma | T \models \varphi' \wedge \textsc{cat}\}$ for L one obtains context-free grammars G_i as described. All legal parse trees according to any of the G_i are (binary) caterpillars with decorations on the internal nodes, hence the G_i can be assumed to be linear grammars. As the mappings h_i do not affect the decorations on leaves, it follows that the yield of $\{T \in \{\otimes, \odot\}\Sigma \mid T \models \varphi' \wedge \textsc{cat}\}$, i.e., $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$, is the union of finitely many linear languages, and thus itself a linear language. $\square$

Combining Lemmas 4 and 5, we obtain the following characterisations.

Theorem 3. *A language over a finite alphabet Σ is linear iff it equals $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$ for some first-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

Theorem 4. *A language over a finite alphabet Σ is linear iff it equals $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$ for some monadic second-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

[11] $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi))$ is always the union of $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi \wedge \exists xy\ x < y))$ and $L(\exists N(\textsc{nest} \wedge \textsc{linear} \wedge \varphi \wedge \neg\exists xy\ x < y))$, of which the latter is finite and thus linear, whereas the former is of the requisite type and thus by assumption also linear. As the union of two linear languages is itself linear, this would also follow for the original language.

[12] A nonterminal node is, in this case, a node decorated with a member of Π. When Π contains no symbols of rank 0 – as it will throughout these notes – a node is nonterminal iff it is internal.

Finally, we take the next theorem to be the result suggested by Lautemann et al. (1995, top of page 213). In Appendix B we show that the version using first-order logic would be false.

Theorem 5. *A language over a finite alphabet* Σ *is linear iff it equals* $L(\exists N(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi))$ *for some monadic second-order sentence* φ *over signature* $(N, <, (Q_a)_{a\in\Sigma})$.

Proof: One direction follows from Lemma 5. The other direction follows from Lemma 4 and the fact that any well is representable by a linear matching and two unary predicates. We sketch the representation, for simplicity dividing it up in two nearly identical steps; first one step representing a well by one unary predicate and what we shall call a functional well, i.e., a well satisfying the rightwards half of *STEEP*, and then a second step representing a functional well by another unary predicate and a linear matching.

By a functional well we mean a well satisfying the axiom below.

$$FUNC \quad \forall x_1x_2y_1y_2(Nx_1y_1 \wedge Nx_2y_2 \rightarrow (x_1 = x_2 \rightarrow y_1 = y_2))$$

Any well can be "pruned" down to a functional well by deleting all but the outermost arc emanating from any given start point. Deleted arcs are "remembered" by assigning a predicate E to the endpoints of all deleted arcs, and the original well N is recoverable from E and the functional, "pruned" well P by the equivalence between Nxy and the disjunction of Pxy and the following.[13]

$$Ey \wedge \exists y'(y < y' \wedge Pxy' \wedge \forall zw(y < z < y' \rightarrow \neg P_W z) \wedge \forall x'(Px'y' \rightarrow x' \leq x))$$

Note that any E and P derived in this way will satisfy the condition $\forall xyz(Pxy \wedge Ez \rightarrow x < z)$. Conversely, any E and functional well P satisfying this latter condition defines a well N by the above equivalence. The second half of the representation is nearly identical, and left to the reader. Since an existential quantification over a well is replaced by existential quantifications over a matching and two unary predicates, the φ in Theorem 5 will be monadic second-order. □

5. Subclasses of the linear languages

Recall that a linear grammar is even linear (Amar and Putzolu, 1964) if $|u| = |v|$ in any production $A \rightarrow uBv$, and that an even linear language is a language generated by an even linear grammar. Langholm and Bezem (2003) gave a characterisation of the class ELL of even linear languages as those definable by monadic second-order sentences over signature $(\rightleftharpoons, <, (Q_a)_{a\in\Sigma})$, where $\rightleftharpoons$ is a binary relation symbol which on any $w = a_0 \ldots a_n$ denotes the relation $\{(i, j) | i \leq j, i + j = n\}$.

Except for the fact that $\rightleftharpoons$ relates any middle element to itself, $\rightleftharpoons$ is essentially a particular linear matching, in fact the unique linear matching violating GAP_1 of the definition below.

[13] Intuitively, any "E-position" y is the endpoint of a deleted arc. (But only one; if two N-arcs end in y then the innermost would (by non-crossing) be the outermost arc emanating from its start point.) Now where did this arc start? To find the start point, first move right from y to the next position y' which is the endpoint of some P-arc. Then find the *innermost* P-arc ending in y'. The start point of this arc is the position we are looking for.

Definition 7. For any $n > 0$ let $\textit{GAP}_n$ be the sentence

$$\exists x_1 \ldots x_n (Sx_1x_2 \wedge \ldots \wedge Sx_{n-1}x_n \wedge \neg\exists z(Nx_1z \vee Nzx_1 \vee \ldots \vee Nx_nz \vee Nzx_n)) \ ,$$

declaring the existence of n consecutive positions not involved in N.
$\textit{GAP}_n$ in general can be used to characterise the class of double Greibach linear languages, as seen from the following results.

Theorem 6. *An ϵ-free language over a finite alphabet Σ is double Greibach linear iff for some $n > 0$ it equals*

$$L(\exists N(\textit{MATCH} \ \wedge \ \textit{LINEAR} \ \wedge \ \neg \ \textit{GAP}_n \wedge \varphi))$$

for a monadic second-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.

Theorem 7. *An ϵ-free language over a finite alphabet Σ is double Greibach linear iff for some $n > 0$ it equals*

$$L(\exists N(\textit{MATCH} \ \wedge \ \textit{LINEAR} \wedge \neg \ \textit{GAP}_n \wedge \varphi))$$

for some first-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.

Proof: The "only if" direction of Theorem 7 (and thus of Theorem 6) follows by inspection of the proof of Lemma 4: if the given grammar G is double Greibach linear, then φ_G implies $\textit{STEEP}$ as well as $\neg\textit{GAP}_n$ for some n. The "if" direction of Theorem 6 (and thus of Theorem 7) is expressed by Lemma 7 of Appendix B. □

Appendix A: Nestings characterise CFL

This appendix sketches a proof of Theorem 2 from Theorem 1. The crucial idea is to represent nestings with matchings; the following concepts are involved: when M is a matching on the word $w \in (\Sigma \cup \{\ ),(\ \})^*$, where), ($\notin \Sigma$, then define the nesting N on w by the equivalence below.

$$\forall xy(Nxy \leftrightarrow x < y \wedge Q_\Sigma x \wedge Q_\Sigma y \wedge \exists x'y'(x' \dashleftarrow x \wedge y \dashrightarrow y' \wedge Mx'y')) \ ,$$

where for variables x, y we define

$Q_\Sigma x$ as $\vee_{a\in\Sigma} \ Q_a x$,

$x \dashrightarrow y$ as $x \leq y \wedge \forall z(x < z \wedge z \leq y \rightarrow Q(z)$, and

$y \dashleftarrow x$ as $y \leq x \wedge \forall z(y \leq z \wedge z < x \rightarrow Q(z)$.

Hence N connects two positions iff the first is to the left of the second and they both hold non-parentheses, and moreover the first possibly has a block of left parentheses immediately to its left and the second possibly a block of right parentheses immediately to its right, and some M-arc starts either in the first of the two positions or somewhere inside the block of left parentheses to its left, and ends either in the second position or somewhere inside the block of right parentheses to its right. It is straightforward to see that N contains no crossing arcs if M doesn't, and in particular that N is a nesting if M is a matching. Now from this nesting N on $w \in (\Sigma \cup \{\,)(\,\})^*$ derive a new nesting N_0 on w_0 simply by deleting parentheses, i.e., w_0 is the result of deleting all parentheses in w, while N_0 connects precisely the same occurrences of symbols from Σ as N does. Now say that the pair (w, M) *represents* the pair (w_0, N_0) if (w_0, N_0) is derived from (w, M) in the way described. Note that such derivation always succeeds, hence every pair (w, M) represents some pair (w_0, N). It is also not very difficult to see that this representation relation is surjective; that every pair (w, N), where N is a nesting on $w \in \Sigma^*$, is represented by some pair (w', M), where M is a matching on $w' \in (\Sigma \cup \{), (\})^{*}$[14]

Now from any first-order sentence φ on signature $(N, <, (Q_a)_{a\in\Sigma})$ define the sentence φ' on signature $(M, <, (Q_a)_{a\in\Sigma}, Q_), Q_()$, by restricting all quantification to positions holding members of Σ, and by replacing any occurrence of N by its definition involving M. Then $(M, \underline{w'}) \models \varphi'$ *iff* $(N, \underline{w}) \models \varphi$ whenever (M, w') represents (N, w).

Let h be the homomorphism from $\Sigma \cup \{), (\}$ to Σ that maps every member of Σ to itself and both parentheses to the empty string. Then, since every pair (N, w) is represented by some pair (M, w') and vice versa, and since $h(w') = w$,

$$\{w \in \Sigma^* | \underline{w} \models \exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi)\}$$

is the image of[15]

$$\{w' \in (\Sigma \cup \{\,), (\,\})^* | \underline{w'} \models \exists M(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi')\}$$

under h. Since the latter is context-free by Theorem 1, and CFL is closed under homomorphism, the former is context-free as well.

This proves one direction of Theorem 2. The other direction is a trivial consequence of Theorem 1, since *STEEP* is a first-order sentence.

Appendix B: Linear matchings do not characterise LIN

This appendix sketches a proof that linear matchings do not characterise LIN in the same strong sense that nestings characterise LIN, i.e., we show that the qualifier "monadic second-order" cannot be replaced by "first-order" in Theorem 5. This negative result is intimately

[14] Essentially, just add parentheses and "move" arcs "out" to separate the start points and endpoints of distinct arcs. A more rigorous argument might employ the notions of Section 2, suitably generalised from wide wells to nestings in general, to show that any nesting N on w defines an ordered forest on w. Then furnish every internal node in this forest with two new children, one to the left decorated with a left parenthesis and one to the right decorated with a right parenthesis. The modified forest would then correspond directly to a pair (w', M) representing (w, N).

[15] We should rather write something like $\exists M(\textit{MATCH}(M/N) \wedge \textit{LINEAR}(M/N) \wedge \varphi')$, to indicate that *MATCH* and *LINEAR* now talk about M rather than N.

connected to the fact that linear grammars fail to admit the Greibach normal form. To see this, we first note the following pumping lemma, which is proved using the standard construction in the proof of the pumping lemma for general context-free languages, and then making a few extra observations.

Lemma 6. *Let L be a Greibach linear language. Then there is some $N > 0$ such that any word of L longer than N can be written in the form $uvwxy$, where $|v| > 0$, $|uvxy| < N$, and where $uv^n wx^n y \in L$ for any $n \geq 0$.*

This is easily seen to spoil the linear language last in Example 2, as well as the one in Example 4, hence these are not Greibach linear.

Lemma 7. *Let Σ be finite, and let φ be a monadic second-order sentence over signature $(N, <, (Q_a)_{a\in\Sigma})$. Then for any $n > 0$ the language $L(\exists N(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi \wedge \neg\textit{GAP}_n))$ is double Greibach linear if it is ϵ-free.*

Proof: Recall that *MATCH* is defined as the conjunction *NEST* and *STEEP*. From the proof of Lemma 5 we see that

$$\{w \mid \underline{w} \models \exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \varphi \wedge \textit{STEEP} \wedge \neg\textit{GAP}_n\}$$

is the yield of the tree language

$$\{T \in \{\otimes, \odot\}\Sigma | T \models \varphi' \wedge \textit{STEEP}' \wedge \neg\textit{GAP}'_n \wedge \textit{CAT}\},$$

where φ', $\textit{STEEP}'$ and $\textit{GAP}'_n$ are defined by the general translation described in the proof of Lemma 5. $\textit{STEEP}'$ says that $ll_T(z_1)$ and $ll_T(z_2)$ are distinct if z_1, z_2 are distinct $\otimes$-nodes, and likewise that $rl_T(z_1)$ and $rl_T(z_2)$ are distinct under this condition. Moreover, $\neg\textit{GAP}'_n$ implies that there cannot be more than $2n\odot$-nodes directly over each other, without $\otimes$-nodes in between.

Now by Proposition 1 there exist context-free grammars $G_1, \ldots, G_m$, where $G_i = (V_i, \Sigma, P_i, S_i)$, and mappings $h_i : V_i \rightarrow \{\otimes, \odot\}$ such that a $\{\otimes, \odot\}\Sigma$-tree T is included in the above tree language iff there is an i and a legal parse tree T_i according to G_i, such that T is the result of applying h_i to the decorations of internal nodes of T_i.

Say that a variable $A \in V_i$ is proper if $h_i(A) = \otimes$, and auxiliary otherwise. The following can now be assumed about all G_i.

- They are linear.
- If A, A' are proper and $A \overset{+}{\Rightarrow} uA'v$ then $|u| > 0$ and $|v| > 0$.
- If $A_0 \Rightarrow u_1A_1v_1 \Rightarrow \ldots \Rightarrow u_1 \ldots u_{2n}A_{2n}v_{2n} \ldots v_1$, then at least one of $A_0, \ldots, A_{2n}$ is proper.

Hence it is not difficult to see that each G_i is equivalent to a linear grammar in double Greibach normal form (essentially by eliminating all auxiliary variables (except possibly the start symbol, which needs special treatment) and adding finitely many new “shortcut” productions) and that they can all be combined into a single grammar of this type. □

Note that a stronger version of Lemma 7 without the clause $\neg$GAP$_n$ in the definition of the language would yield Proposition 2 below directly, since not all linear languages are Greibach linear. Such a stronger version of Lemma 7 is not provable, however, since the language $\{a^n b^n c^m | n, m > 0\}$ is in fact first-order definable using matchings, as was seen in Example 3, and still fails the Pumping Lemma and is thus not Greibach linear. To obtain the proposition we shall therefore have to consider instead a somewhat more involved argument using the language $\{a^n b^n c^{2m} | n, m > 0\}$ of Example 4.

Proposition 2. *Let Σ be an alphabet containing at least two symbols. Then there is a linear language over Σ which is not of the form $L(\exists N(\textit{MATCH} \wedge \textit{LINEAR} \wedge \varphi))$ for any first-order sentence φ over signature $(N, <, (Q_a)_{a\in\Sigma})$.*

Proof: To simplify details we consider a three-symbol alphabet, but an argument using only two symbols would be quite similar.[16]

Let L be $\{a^n b^n c^{2m} | n, m > 0\}$ and suppose, for a *reductio ad absurdum* argument, that $L = L(\exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \textit{STEEP} \wedge \varphi))$ for some first-order sentence φ over the signature $(N, <, Q_a, Q_b, Q_c)$.

By Lemmas 6 and 7, L cannot, however, for any $n > 0$, equal $L(\exists N(\textit{NEST} \wedge \textit{LINEAR} \wedge \textit{STEEP} \wedge \varphi \wedge \neg \textit{GAP}_n))$.

Now let m be the quantifier-depth of φ and let n be $3 \cdot 2^m$; from the above we know that for some $w = a^k b^k c^{2l}$ and some linear matching N on w, we have both $(\underline{w}, N) \models \varphi$ and $(\underline{w}, N) \models \textit{GAP}_n$. Hence w contains a block of 2^m consecutive positions that are not involved in N, and which are all decorated with the same terminal symbol.

Now by an argument that in all relevant respects is identical to the one given in Example 4.3 of Thomas (1997) (which is there used to prove that the language $\{a^{2n} | n > 0\}$ is not first-order definable), it can be shown that $(\underline{w}, N) \cong_m (\underline{w}', N')$, where w' is a word identical to w but in which an extra position, decorated with the same terminal symbol, has been inserted into the block mentioned above, and N' is exactly as N, i.e., relating the "old" positions as N does.

But then by the Ehrenfeucht–Fraïssé Theorem (for a suitably formulated reference see Theorem 4.1 of Thomas (1997)) it follows that also $(\underline{w}', N') \models \varphi$, hence $w' \in L$, which is false as w' is either $a^{k+1} b^k c^{2l}$ or $a^k b^{k+1} c^{2l}$ or $a^k b^k c^{2l+1}$. □

References

Amar, V., & Putzolu, G. (1964). On a family of linear grammars. *Information and Control*, *7*, 283–291.

Eiter, T., Gottlob, G., & Gurevich, Y. (2000). Existential second-order logic over strings. *Journal of the ACM*, *47*, 77–131.

Gécseg, F., & Steinby, M. (1997). Tree languages. In Rozenberg, G. and A. Salomaa, editors, *Handbook of Formal Languages*, vol. 3, Springer-Verlag.

Greibach, S. (1965). A new normal-form theorem for context-free phrase structure grammars. *Journal of the ACM*, *12*, 42–52.

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

Langholm, T., & Bezem, M. (2003). A descriptive characterisation of even linear languages. *Grammars 6*, 169–181.

[16] Using instead the language $\{a^n b^n a^{2m} | n, m > 0\}$.

Lautemann, C., Schwentick, T., & Thérien, D. (1995). Logics for context-free languages. In Pacholsky, L. and J. Tiuryn, editors, *Computer Science Logic*, *LNCS 933*, 205–216. Springer-Verlag.

Rosenkrantz, D.J. (1967). Matrix equations and normal forms for context-free grammars. *Journal of the ACM*, *14*, 501–507.

Thomas, W. (1997). Languages, Automata and Logic. In Rozenberg, G. and A. Salomaa, editors, *Handbook of Formal Languages*, Vol. 3, Springer-Verlag.