JMLR: Workshop and Conference Proceedings 21:84–96, 2012 The 11th ICGI

Beyond Semilinearity: Distributional Learning of Parallel Multiple Context-free Grammars

Alexander Clark ALEXC@CS.RHUL.AC.UK
Department of Computer Science Royal Holloway, University of London

Ryo Yoshinaka RY@I.KYOTO-U.AC.JP
Graduate School of Informatics, Kyoto University

Editors: Jeffrey Heinz, Colin de la Higuera and Tim Oates

Abstract

Semilinearity is widely held to be a linguistic invariant but, controversially, some linguistic phenomena in languages like Old Georgian and Yoruba seem to violate this constraint. In this paper we extend distributional learning to the class of parallel multiple context-free grammars, a class which as far as is known includes all attested natural languages, even taking an extreme view on these examples. These grammars may have a copying operation that can recursively copy constituents, allowing them to generate non-semilinear languages. We generalise the notion of a context to a class of functions that include copying operations. The congruential approach is ineffective at this level of the hierarchy; accordingly we extend this using dual approaches, defining nonterminals using sets of these generalised contexts. As a corollary we also extend the multiple context free grammars using the lattice based approaches.

1. Introduction and Motivation

There are two fundamental language theoretic boundaries that are closely related: the first is the boundary between regular languages and non-regular languages, the second is between semilinear languages and non-semilinear languages. Semilinear languages are, roughly speaking, those where the lengths of the strings in the language are linear combinations of a finite set of fixed lengths.[1] Joshi et al. (1991) say that semilinearity:

> is intended to be an approximate characterization of the linguistic intuition that sentences of a natural language are built from a finite set of clauses of bounded structures using certain simple linear operations.

These two boundaries are clearly related because of the following theorem: a language is semilinear iff it is letter equivalent to a regular language. Clearly the class of semilinear languages is not directly useful as it is uncountable and thus contains undecidable languages, but it serves to help demarcate the class(es) of mildly context-sensitive (MCS) languages. All standardly used grammatical formalisms are semilinear – regular grammars, context-free grammars, multiple context-free grammars, tree adjoining grammars and so on all define subsets of the class of semilinear languages. Examples of non-semilinear languages include

1. See Michaelis and Kracht (1997) for a precise definition.

$\{ a^{2^n} \mid n > 0 \}$ and $\{ a^{n^2} \mid n > 0 \}$, which can be parsed in linear time, yet cannot be expressed by MCS formalisms. Formalisms that can define non-semilinear languages include range concatenation grammars and the representation we will use in this paper, parallel multiple context-free grammars (PMCFGs), that we define in Section 2.

Recent work in grammatical inference has made significant progress in learning semilinear, non-regular languages using representations such as context-free grammars (Clark and Eyraud, 2007) and multiple context-free grammars (Yoshinaka, 2011b). Crucially, these representations just use concatenation – substrings are combined, but never copied. The richer operations used by MCFGs are just generalisations of concatenation to tuples of strings.

There is a broad consensus that natural language string sets are *semilinear*, and so attention has focused largely on properties of formalisms that generate semilinear languages. However there are a number of cases where linguistic data suggest that there are richer processes involved, processes that either require or might benefit from a more powerful formalism. These data, which we examine in detail in Section 3, are still controversial. However, regardless of what the final determinations on these examples are, it is still useful to have richer learning algorithms available since even if these formalisms are not strictly speaking necessary, the additional descriptive power that they give us may allow for a more compact and succinct grammar than we could obtain with a semilinear formalism.

In this paper we extend distributional learning to the inference of non-semilinear languages; the major technical detail is the extension of the notion of context. We give an intuitive explanation of this in Section 4, and present the technical details of the learning target and algorithm together with the proof of its correctness in Section 5.

2. Preliminaries

The sets of non-negative and strictly positive integers are denoted by $\mathbb{N}$ and $\mathbb{N}^+$, respectively. A sequence over an alphabet Σ is called a *word*. The empty word is denoted by λ. Σ^* denotes the set of all words and $\Sigma^+ = \Sigma^* - \{\lambda\}$. Any subset of Σ^* is called a *language (over Σ)*. An *m-word* is an m-tuple of words and we denote the set of m-words by $\mathcal{S}_m$. Any m-word is a *multiword*. Similarly we define $\mathcal{S}_{\leq m} = \bigcup_{i \leq m} \mathcal{S}_i$ and $\mathcal{S}_* = \bigcup_{i \in \mathbb{N}} \mathcal{S}_i$.

We fix a countably infinite set X of *variables* $x_1, x_2, \ldots$. We use y, y', y_i etc. as meta variables for variables in X. A *pattern* is a string over $\Sigma \cup X$. For a pattern π, we denote by X_π the set of variables that occur in π. An *m-pattern* π is a pattern such that $|X_\pi| = m$. Hence a 0-pattern is a synonym of a word. A pattern is said to be *n-copying* if each variable occurs at most n times in it. An *m-context* is an m-pattern π such that $X_\pi = \{x_1, \ldots, x_m\}$. We denote the set of m-contexts by $\mathcal{C}_m$ and that of n-copying m-contexts by $\mathcal{C}_{m,n}$. Note that every element of $\mathcal{C}_{m,n}$ contains *exactly* m variables, each of which occurs *at most* n times. In preceding papers on distributional learning algorithms (e.g. Clark and Eyraud (2007)), a context is defined to be a pair (l, r) of words. Those correspond to lx_1r in our notation and particularly the empty context (λ, λ) is denoted by x_1 in this paper. One can see an m-context as a function that takes an m-word as an argument and returns a word formed from the components of the m-word and other words attached to the m-context. For example, from a 2-context $ax_1bx_2x_1$ and a 2-word (c, d), one will get a word $acbdc$. We formally define the composition of an m-context and an m-word through a substitution. A *substitution* θ is a finite partial function from X to Σ^*, which is extended to the homomorphism $\hat{\theta}$ from $(\Sigma \cup X)^*$

to $(\Sigma \cup X)^*$ such that $\hat{\theta}(y) = \theta(y)$ if y is in the domain of θ, and $\hat{\theta}(y) = y$ otherwise. We identify $\hat{\theta}$ and θ if no confusion arises. A substitution θ is often denoted as a suffix operator $[y_1 \mapsto \theta(y_1), \dots, y_k \mapsto \theta(y_k)]$ where $\{y_1, \dots, y_k\}$ is the domain of θ. Particularly when the domain is $\{x_1, \dots, x_k\}$, it is denoted by $[\theta(x_1), \dots, \theta(x_k)]$ or $[\theta(x_1, \dots, x_k)]$ omitting the domain. E.g., $ax_1bx_2x_1[c, d] = acbdc$. We naturally adapt those notations to tuples. For a tuple of variables $\boldsymbol{y} = (y_1, \dots, y_k)$ and a multiword $\boldsymbol{v} = (v_1, \dots, v_k)$, $\theta(\boldsymbol{y}) = \boldsymbol{v}$ means that $\theta(y_i) = v_i$ for each $i = 1, \dots, k$, which is also denoted by $[\boldsymbol{y} \mapsto \boldsymbol{v}]$. Particularly when $\boldsymbol{y} = (x_1, \dots, x_k)$, it is simply denoted as $[\boldsymbol{v}]$. (The homomorphic extension of) a substitution operation is naturally generalized for sets $C \subseteq \mathcal{C}_k$ and $K_i \subseteq \Sigma^*$ for $i = 1, \dots, k$ as $C[K_1, \dots, K_k] = \{\, w[v_1, \dots, v_k] \mid w \in C,\ v_i \in K_i \text{ for } i = 1, \dots, k \,\}$.

In what follows, we will consider a fixed language L and we denote the set of m-words that every r-copying m-context in a set $C \subseteq \mathcal{C}_{m,r}$ accepts with respect to a language $L \subseteq \Sigma^*$ by

$$C^\dagger = \{\, \boldsymbol{v} \in \mathcal{S}_m \mid \pi[\boldsymbol{v}] \in L \text{ for all } \pi \in C \,\}.$$

By definition, $C[K] \subseteq L$ iff $K \subseteq C^\dagger$. Note that if $C = \{x_1\}$ then $C^\dagger = L$. For a language $L \subseteq \Sigma^*$, we let

$$\mathrm{Sub}_{\leq p}(L) = \{\, \boldsymbol{v} \in \mathcal{S}_m \mid \pi[\boldsymbol{v}] \in L \text{ for some } \pi \in \mathcal{C}_{m,1} \text{ with } m \leq p \,\},$$
$$\mathrm{Con}_{\leq p,r}(L) = \{\, \pi \in \mathcal{C}_{m,r} \mid \pi[\boldsymbol{v}] \in L \text{ for some } \boldsymbol{v} \in \mathcal{S}_m \text{ with } m \leq p \,\}.$$

Note that $\mathrm{Con}_{\leq 0,r}(L) = L$ and that replacing $\mathcal{C}_{m,1}$ in the definition of $\mathrm{Sub}_{\leq p}(L)$ by $\mathcal{C}_m$ gives an equivalent definition.

2.1. Parallel multiple context-free grammars

A *ranked alphabet* is a pair $\langle N, \dim \rangle$ of an alphabet N and a function $\dim : N \to \mathbb{N}^+$. The number $\dim(A)$ is called the *dimension* of A. We often simply express a ranked alphabet $\langle N, \dim \rangle$ by N if no confusion arises. By N_d we denote the subset of N whose elements have dimension d.

Seki et al. (1991) introduced *parallel multiple context-free grammars* (PMCFGs) as a generalization of context-free grammars. The formalism is essentially equivalent to linear context-free rewriting systems (Vijay-Shanker et al., 1987). A PMCFG is a tuple $G = \langle \Sigma, N_{\dim}, S, P \rangle$ where Σ is an alphabet whose letters are called *terminals*, $N_{\dim}$ is a ranked alphabet whose elements are called *nonterminals*, $S \in N$ is a special symbol of dimension 1, and P is a set of *production rules*.

Production rules in P have the following form:[2]

$$B_0(\pi_1, \dots, \pi_{d_0}) :\!- B_1(y_{1,1}, \dots, y_{1,d_1}), \dots, B_k(y_{k,1}, \dots, y_{k,d_k})$$

where $B_0, B_1, \dots, B_k \in N$ for some $k \geq 0$, $d_i = \dim(B_i)$ for each $i \in \{0, \dots, k\}$, variables $y_{1,1}, \dots, y_{k,d_k}$ are distinct, and each π_j for $j = 1, \dots, d_0$ is a pattern such that

$$\bigcup_{1 \leq j \leq d_0} X_{\pi_j} = \{\, y_{i,j} \mid 1 \leq i \leq k,\ 1 \leq j \leq d_i \,\}.$$

2. The notation adopted in this paper follows Smullyan's elementary formal systems (1961) rather than Seki et al. (1991). We only consider non-deleting productions in this paper.

If $k = 0$ then the right-hand side is empty, and the production is of the form $B(\boldsymbol{v}) :-$ where $\boldsymbol{v} \in \mathcal{S}_{\dim(B)}$. We say that a rule is *r-copying* if the concatenation of the patterns on its left-hand side is r-copying, that is, no variables occur more than r times on the left-hand side.

Example 1 *A tuple $G_1 = \langle \{a\}, N_{\dim}, S, P \rangle$ where $N_{\dim} = N_1 = \{S\}$ and P consists of the two rules*

$$S(x_1 x_1) :- S(x_1); \quad S(a) :-$$

is a PMCFG.

A tuple $G_2 = \langle \{a\}, N_{\dim}, S, P \rangle$ where $N_{\dim} = \{S, A\}$ with $N_1 = \{S\}$ and $N_2 = \{A\}$ and P consists of the three rules

$$S(x_1 x_2 a) :- A(x_1, x_2); \quad A(x_1 x_2 a, x_2 a a) :- A(x_1, x_2); \quad A(\lambda, \lambda) :-$$

is a PMCFG.

The derivation process of a PMCFG is given as follows. We write $\vdash_G A(\boldsymbol{v})$ if $A(\boldsymbol{v}) :-$ is a rule in P. If we have $\vdash_G B_i(\boldsymbol{v}_i)$ for all $i = 1, \ldots, k$ and P has a rule $B_0(\pi_1, \ldots, \pi_{d_0}) :- B_1(\boldsymbol{y_1}), \ldots, B_k(\boldsymbol{y}_k)$, then we deduce

$$\vdash_G B_0(\theta(\pi_1, \ldots, \pi_{d_0}))$$

where $\theta(\boldsymbol{y_i}) = \boldsymbol{v}_i$ for all $i = 1, \ldots, k$. We will abbreviate this substitution θ as $[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]$. The *language of A* is defined by

$$\mathcal{L}(G, A) = \{\, \boldsymbol{v} \in \mathcal{S}_{\dim(A)} \mid \vdash_G A(\boldsymbol{v}) \,\}.$$

The *language of G* is $\mathcal{L}(G) = \mathcal{L}(G, S)$.

For the grammar G_1 in Example 1, clearly we have $\vdash_{G_1} S(a^{2^n})$ for all $n \in \mathbb{N}$ and in fact $\mathcal{L}(G_1) = \{\, a^{2^n} \mid n \in \mathbb{N} \,\}$. For the grammar G_2 in Example 1, it is easy to see that we have $\vdash_{G_2} A(a^{n^2}, a^{2n})$ for all $n \in \mathbb{N}$ and in fact $\mathcal{L}(G_2) = \{\, a^{n^2} \mid n \in \mathbb{N}^+ \,\}$. See Section 3 for some more examples together with some derivation trees.

We denote by $\mathbb{G}(p, q, r)$ the class of PMCFGs such that the dimension of every nonterminal is at most p and every production rule has at most q nonterminals on the right-hand side and is r-copying. For the grammars in Example 1, we have $G_1 \in \mathbb{G}(1, 1, 2)$ and $G_2 \in \mathbb{G}(2, 1, 2)$.

Theorem 1 (Seki et al. (1991)) *The uniform membership problem for $\mathbb{G}(p, q, r)$ is solvable in polynomial time whose degree is linear in pq.*

3. Linguistic Examples

While semilinearity is generally considered to be a property that holds for all natural languages, there is an increasing amount of evidence that there are some phenomena that take natural languages out of the class of semilinear languages. None of the arguments here are as conclusive as Shieber's argument that natural languages are not weakly context-free (Shieber, 1985) but are nonetheless suggestive. In what follows we will describe fragments of languages that are not semilinear and will provide toy PMCFGs for these fragments. Figure 1

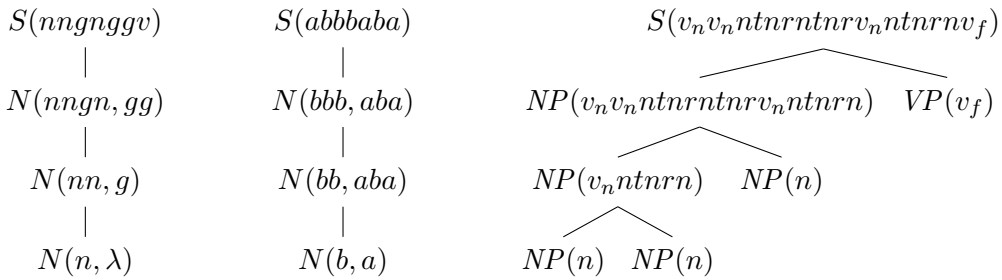

Figure 1: Derivation trees for these examples. On the left, Old Georgian; middle Chinese numbers; on the right Yoruba.

shows an example derivation tree for each of the three examples here. Another important area where (non-recursive) copying operations may occur is in morphology and phonology. For reasons of space we cannot discuss these here; see Inkelas and Zoll (2005) for a wide range of examples of reduplication in many languages. Some of these involve copying of entire stems which can in principle be unbounded in length.

3.1. Suffixaufnahme in Old Georgian

Old Georgian is a now dead language that has a particularly extreme form of suffix stacking (*Suffixaufnahme*); a phenomenon that is fairly widespread in European languages and Australian languages such as Dyirbal and Kayardild. The exact status is controversial (Michaelis and Kracht, 1997; Bhatt and Joshi, 2004); here we assume that the arguments are valid. For reasons of space we will just describe the string set concerned, rather than attempt to describe the linguistic data: n can be thought of as a noun, g a genitive suffix and v a verb. We have a string set over three letters $\{n, g, v\}$ where the language is: $\{nv, nngv, nngnggv, nngnggngggv, \dots\}$. More formally, defining $u_i = ng^i$ this is the language: $\{\, nu_1 \dots u_k v \mid k \geq 0 \,\}$. This is not semilinear, since the total number of occurrences of g will be a quadratic function of the number of ns in the string. We can describe this string set with the grammar:

$$\begin{aligned} S(x_1x_2v) &:\!- N(x_1, x_2)\ ; \\ N(x_1x_2n, x_2g) &:\!- N(x_1, x_2)\ ; \\ N(n, \lambda) &:\!-\ . \end{aligned}$$

3.2. Yoruba

Kobele (2006) argues that Yoruba, a Nigerian language, has a certain type of recursive copying in relative clauses. Yoruba can form relative clauses by copying entire verb phrases — the verb phrases can have nouns which can have relative clauses; the end result of this is a language which under intersection with a suitable regular language, and after homomorphism gives the language $\{\, a^{2^n} \mid n \geq 0 \,\}$. Yoruba has noun phrases of the form

(Example 4.48 of Kobele (2006)) 'rira NP ti Ade ra NP' (the fact that Ade bought NP) where NP is a noun phrase which must be copied; the two occurrences of NP must be identical.[3]

Noun phrases can also be formed into sentences like 'Ade ra NP' (Ade bought NP) or 'NP ko da' (NP is not good).

We can represent this tiny non-semilinear fragment of Yoruba with the grammar:

$$\begin{aligned} S(x_1x_2) &:\!- NP(x_1), VP(x_2) \ ; \\ NP(n) &:\!- \ ; \\ NP(v_n x_1 t x_2 r x_1) &:\!- NP(x_1), NP(x_2) \ ; \\ VP(v_f) &:\!- \ , \end{aligned}$$

where we write n for nouns like 'adiẹ' (chicken) and proper nouns like 'Ade', v_n for nonfinite verbs like 'rira' (buying), v_f for finite verb phrases, and t, r for 'ti' and 'ra'. We can verify that this language is not semilinear by counting the number of occurrences of t in grammatical sentences. Similar phenomena occur widely in West African languages, though Yoruba has perhaps the most complex system of this type.

3.3. Chinese number names

In Mandarin Chinese, a certain subset of number names can be formed from 'wu' (5) and 'zhao' (10^{12}). Here we will write a for 'wu' and b for 'zhao'. The wellformed expressions intersected with a suitable regular language form the language $L_{CN} = \{\, ab^{k_1}ab^{k_2}\dots ab^{k_n}a \mid k_1 > k_2 \cdots > k_n > 0 \,\}$. This data is controversial as it is not clear whether the wellformedness of number expressions should form part of the syntax of the language. Here we assume that it does, in which case the language is not semilinear (Radzinski, 1991). A grammar for this is:

$$\begin{aligned} S(ax_1x_2) &:\!- N(x_1, x_2) \ ; \\ N(bx_1, x_2) &:\!- N(x_1, x_2) \ ; \\ N(bx_1, ax_1x_2) &:\!- N(x_1, x_2) \ ; \\ N(\lambda, \lambda) &:\!- \ . \end{aligned}$$

4. Informal Explanation of the Approach

We will now give an informal introduction to the extension of distributional learning to these formalisms; the basic idea is quite natural but may be obscured by the unavoidable complexity of the notation.

In distributional learning we typically consider a context (l, r) which we can wrap around a substring u to give a complete string lur. Consider this context rather as a function f from a substring to a full sentence. $u \mapsto lur$, which in our notation is represented by what we call a 1-copying 1-context lx_1r, an element of $\mathcal{C}_{1,1}$.

In the derivation of a string wrt a CFG, these functions correspond to the operation that takes the yield of a nonterminal and integrates into the rest of the sentence: given a

3. There can apparently be slight differences in the NPs which we neglect here.

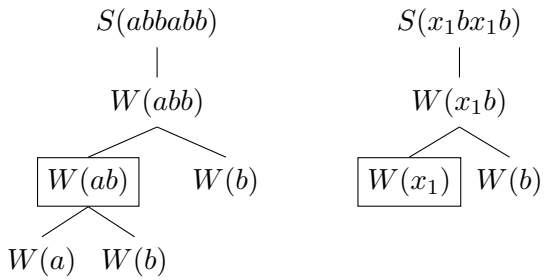

Figure 2: Example of a context as a function from strings to strings.

derivation like $S \overset{*}{\Rightarrow} lNr \overset{*}{\Rightarrow} lur$, we can consider the derivation $S \overset{*}{\Rightarrow} lNr$ to be applying a function $f \in \mathcal{C}_{1,1}$ to the yield of N.

In a parallel CFG, we might again have a nonterminal N that derives a string u. However, the part of the derivation that produces the whole sentence from u may include rules that copy u.

Example 2 *Consider for example the language* $\{\, ww \mid w \in \{a,b\}^+ \,\}$. *This could be defined as a parallel CFG with two nonterminals* S *and* W, *together with productions:*

$$S(x_1x_1) :- W(x_1);\;\; W(a) :- ;\;\; W(b) :- ;\;\; W(x_1x_2) :- W(x_1), W(x_2).$$

Figure 2 shows a derivation tree of *abbabb*; we can pick one node in the tree (marked with a box). If we consider the "context" of the node $W(ab)$, then this is not a simple context, but rather the function $x_1 \mapsto x_1bx_1b$. We can see this by replacing this node in the tree with a variable x_1, as on the right hand side of Figure 2.

Therefore with this richer class of grammars we need to consider a larger class of functions that correspond to q-copying contexts, and when we consider tuples of strings in the full PMCFG formalism, to q-copying r-contexts: the class $\mathcal{C}_{r,q}$. Given such a set of functions we can consider the 'distribution' in this extended sense of a substring in a language to be the set of functions that when applied to that substring give an element of the language.

In this paper we use a dual approach – the nonterminals are defined by small finite sets of patterns/functions, and incorrect rules will be eliminated by strings or tuples of strings.

In Example 2, we can see how the nonterminals that we need can be picked out. The symbol S will correspond as usual to the single simple pattern x_1 – the identity function which corresponds to the empty context (λ, λ) in distributional learning of CFGs (e.g. Clark and Eyraud (2007)). The symbol W corresponds to the 2-copying context x_1x_1. It is easy to see that the set of strings generated by W is exactly the same as the set of strings which can occur in the context x_1x_1. In the notation we defined earlier we have a singleton set $C_W = \{x_1x_1\}$ such that $\mathcal{L}(G, W) = \{\, v \in \Sigma^* \mid C_W[v] \subseteq \mathcal{L}(G) \,\}$. Note that for any grammar, the start symbol S will be characterised by the single context x_1. We shall show that languages that have this nice property – that the languages defined by each nonterminal can be picked out by a small set of contexts – are learnable by a straightforward algorithm, very similar to ones that have been used before for distributional learning of CFGs.

5. Learning Target and Algorithm

Definition 2 *We say that a* PMCFG G *has* the (r,s)-finite context property *((r,s)-FCP) if each nonterminal $A \in N_d$ admits a nonempty set $C_A \subseteq \mathcal{C}_{d,r}$ of r-copying d-contexts such that $|C_A| \leq s$ and*

$$\mathcal{L}(G,A) = \{\, \boldsymbol{v} \in \mathcal{S}_d \mid C_A[\boldsymbol{v}] \subseteq \mathcal{L}(G) \,\}.$$

Such a set C_A is called a characterising set *of A.*

By $\mathbb{G}(p,q,r,s)$ we denote the subclass of $\mathbb{G}(p,q,r)$ where grammars have the (r,s)-FCP. The class of languages generated by grammars in $\mathbb{G}(p,q,r,s)$ is denoted by $\mathbb{L}(p,q,r,s)$.

Clearly the above definition is a generalisation of the s-FCP (Clark, 2010). All regular languages are in $\mathbb{L}(1,1,1,1)$ and the Dyck language is in $\mathbb{L}(1,2,1,1)$.

Example 3 $\{\, a^{2^n} \mid n \geq 0 \,\} \in \mathbb{L}(1,1,2,1)$ *since the 2-copying 1-context x_1x_1 characterises the nonterminal S of G_1 in Example 1. Similarly $\{\, vv \mid v \in \Sigma^* \,\} \in \mathbb{L}(1,1,2,1)$, since the context x_1x_1 again characterises the relevant set of strings.*

Consider the examples in Section 3: in the Old Georgian case, we can use the single context $x_1x_2nx_2gv$ to pick out the multiwords generated by the nonterminal N. In the Yoruba case, the NP class is picked out by the context x_1v_f and the VP class (just the symbol v_f in this trivial example) by the context nx_1. Finally in the Chinese number example, the context $abx_1ax_1x_2$ defines the set of multiwords $b^k, ab^{k_1} \dots ab^{k_n}a$ where $k > k_1 > \cdots > k_n$ with $n \geq 0$, which is what is required. Proving these requires some analysis of particular cases, which we omit for reasons of space.

5.1. Hypothesis

Hereafter we arbitrarily fix rather small natural numbers $p,q,r,s \geq 1$ and a target language $L_* \in \mathbb{L}(p,q,r,s)$ to be learnt. The learner receives a presentation of positive data in the identification in the limit paradigm; we assume that our learner has in addition access to an oracle which answers *membership queries* (MQs), which says whether an arbitrary string u belongs to the learning target L_*. See for example Yoshinaka (2010) for details.

Our learner is a straightforward generalisation of the one for CFGs with the s-FCP given in Yoshinaka (2011a). The learner constructs its conjecture $\hat{G} = \mathcal{G}(K,F)$ from finite sets of multiwords $K \subseteq \mathrm{Sub}_{\leq p}(D)$ and r-copying contexts $F \subseteq \mathrm{Con}_{\leq p,r}(D)$ where D is a set of positive examples. We let $K_i = K \cap \mathcal{S}_i$ for $i = 1, \dots, p$ and $F_j = F \cap \mathcal{C}_{j,r}$ for $j = 0, \dots, p$. We assume that $\{x_1\} \in F_1$. For $C \subseteq \mathcal{C}_{i,r}$, we define

$$C^{(K)} = C^\dagger \cap K_i = \{\, \boldsymbol{v} \in K_i \mid C[\boldsymbol{v}] \in L_* \,\}.$$

The nonterminal set $\hat{N} = \bigcup_{1 \leq i \leq p} \hat{N}_i$ of $\hat{G}$ is given by

$$\hat{N}_i = \{\, [\![C]\!] \mid C \subseteq F_i \text{ with } 1 \leq |C| \leq s \,\},$$

where $[\![C]\!]$ simply means a symbol indexed with C. The start symbol is $[\![\{x_1\}]\!]$.

Our grammar $\mathcal{G}(K,F)$ has rules of the form

$$[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \dots, [\![C_k]\!](\boldsymbol{x}_k)$$

if and only if the following conditions hold:

- $0 \leq k \leq q$,
- $\boldsymbol{\pi}$ is a d_0-tuple of patterns whose concatenation is an r-copying d-context,
- $|\boldsymbol{x}_i| = d_i$ for each $i = 1, \ldots, k$ and the variables from $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$ constitute $X_{\boldsymbol{\pi}}$,
- there are $\boldsymbol{v}_i \in \mathcal{S}_{d_i}$ for $i = 1, \ldots, k$ and $\pi_0 \in \mathcal{C}_{d_0,1}$ such that $\pi_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \in F_0$,
- $C_0[\boldsymbol{\pi}[C_1^{(K)}, \ldots, C_k^{(K)}]] \subseteq L_*$,

where d_i are such that $[\![C_i]\!] \in N_{d_i}$ for $i = 0, \ldots, k$ and $d = \sum_{1 \leq i \leq k} d_i$.

Lemma 3 *One can compute $\mathcal{G}(K, F)$ in polynomial time in $\|D\|$.*

Proof By $F \subseteq \mathrm{Con}_{\leq p,r}(D)$ and $K \subseteq \mathrm{Sub}_{\leq p}(D)$, $\|F\|$ and $\|K\|$ are bounded by a polynomial in $\|D\|$ with a degree linear in pr and p, respectively. For each $[\![C]\!] \in \hat{N}$, the fact $|C| \leq s$ implies $|\hat{N}| \leq |F|^s$.

We estimate the number of rules of the form

$$[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)\,. \tag{1}$$

By $k \leq q$, at most $|\hat{N}|^{q+1}$ combinations of nonterminals are possible. There exist $u \in F_0 \subseteq D$, $\boldsymbol{v}_i \in \mathrm{Sub}_{\leq p}(u)$ for $i = 1, \ldots, k$ and $\pi_0 \in \mathrm{Con}_{\leq p,1}(u)$ such that $u = \pi_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]]$. Determining $\boldsymbol{\pi}$ can be seen as determining the left and right positions in u to which each occurrence of a variable corresponds. Note that π_0 and $\boldsymbol{\pi}$ contain at most p and pqr occurrences of variables, respectively. Thus we extract at most $|u|^{2(p+pqr)}$ variants of $\boldsymbol{\pi}$ from a string $u \in F_0$. Therefore, we have at most $|\hat{N}|^{q+1}|F_0|\ell^{2(p+pqr)}$ production rules in $\mathcal{G}(K, F)$ where ℓ is the length of a longest word in F_0.

Among those rules, the algorithm rejects some incorrect rules. To compute $C_i^{(K)}$, we call the membership oracle on at most $|C_i||K|$ words. To see whether

$$C_0[\boldsymbol{\pi}[C_1^{(K)}, \ldots, C_k^{(K)}]] \subseteq L_*$$

concerning a rule of the form (1), it is enough to check the membership on at most $|C_0| \prod_{1 \leq i \leq k} |C_k^{(K)}| \leq s|K|^q$ words.

For checking whether $D \subseteq \mathcal{L}(\hat{G})$, it takes polynomially many steps by Theorem 1.

All in all, one can compute $\mathcal{G}(K, F)$ in polynomial time in $\|D\|$, where the degree of the polynomial linearly depends on $pqrs$. ■

Just like the algorithms based on syntactic concept lattices (e.g. Clark (2010)), we establish the following monotonicity lemma: Expansion of F expands the hypothesised language while expansion of K shrinks the hypothesised language.

Lemma 4 (Monotonicity) *Let $\hat{G} = \mathcal{G}(K, F)$ and $\hat{G}' = \mathcal{G}(K', F')$.*

1. *If $K \subseteq K'$ and $F = F'$, then $\mathcal{L}(\hat{G}) \supseteq \mathcal{L}(\hat{G}')$.*
2. *If $K = K'$ and $F \subseteq F'$, then $\mathcal{L}(\hat{G}) \subseteq \mathcal{L}(\hat{G}')$.*

Proof (1) Every rule of $\hat{G}'$ is also a rule of $\hat{G}$. (2) Every rule of $\hat{G}$ is also a rule of $\hat{G}'$. ■

We say that a rule of the form $[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)$ is *correct* if it holds that $C_0[\boldsymbol{\pi}[C_1^\dagger, \ldots, C_k^\dagger]] \subseteq L_*$, that is, for any $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in \mathcal{S}_*$,

$$C_i[\boldsymbol{v}_i] \subseteq L_* \text{ for all } i \in \{1, \ldots, k\} \implies C_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \subseteq L_* \,.$$

If a rule is not correct, it is *incorrect*.

Lemma 5 *Every context set F admits a multiword set K of a polynomial cardinality in $\|F\|$ such that $\hat{G} = \mathcal{G}(K, F)$ has no incorrect rules.*

Proof Suppose that a rule $[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)$ is incorrect. There exist multiwords $\boldsymbol{v}_i \in C_i^\dagger$ for $i = 1, \ldots, k$ such that $C_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \nsubseteq L_*$. If $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k \in K$, such a rule is suppressed. This proves the lemma together with the facts that $k \leq q$ and that the number of production rules is polynomially bounded by $|F|\ell$ with $\ell = \max\{\, |u| \mid u \in F_0 \,\}$ by the proof of Lemma 3. ■

We say that K is *fiducial on* F if $\mathcal{G}(K, F)$ has no incorrect rules.

Lemma 6 *If $\hat{G} = \mathcal{G}(K, F)$ has no incorrect rules, then $\mathcal{L}(\hat{G}) \subseteq L_*$.*

Proof We show by induction that $\vdash_{\hat{G}} [\![C]\!](\boldsymbol{v})$ implies $C[\boldsymbol{v}] \subseteq L_*$. This implies particularly for $\vdash_{\hat{G}} [\![\{x_1\}]\!](v)$, where $[\![\{x_1\}]\!]$ is the start symbol of $\hat{G}$, we have $v \in L_*$.

Suppose that we have $\vdash_{\hat{G}} [\![C_0]\!]([\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]])$ by the rule $[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)$ and $\vdash_{\hat{G}} [\![C_i]\!](\boldsymbol{v}_i)$ for $i = 1, \ldots, k$. By the induction hypothesis we have $C_i[\boldsymbol{v}_i] \subseteq L_*$ for $i = 1, \ldots, k$. (When $k = 0$, it is the base case.) Since the rule is correct, we have $C_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \subseteq L_*$. ■

Lemma 7 *Let $G_* = \langle \Sigma, N_*, P_*, S_* \rangle \in \mathbb{G}$ generate L_*. Suppose that F includes a characterising set C_A for every nonterminal $A \in N_*$ and that F_0 contains a string $v_\rho \in \mathcal{L}(G_*)$ derived by using ρ for every rule $\rho \in P_*$. Then $L_* \subseteq \mathcal{L}(\mathcal{G}(K, F))$ for any K.*

Proof Let $\hat{G} = \mathcal{G}(K, F)$. For a rule $A_0(\boldsymbol{\pi}) :- A_1(\boldsymbol{x}_1), \ldots, A_k(\boldsymbol{x}_k)$ of G_*, let $C_i \subseteq F_{\dim(A_i)}$ be a characterising set of A_i for $i = 0, \ldots, k$. By the assumption, there are $\boldsymbol{v}_i \in \mathcal{L}(G_*, A_i)$ for $i = 1, \ldots, k$ and a $\dim(A_0)$-pattern π_0 such that $\pi_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \in \mathcal{L}(G) \cap F_0$. We show that $\hat{G}$ has the corresponding rule $[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)$ whatever K is. For any $\boldsymbol{u}_i \in C_i^{(K)} = \mathcal{L}(G_*, A_i) \cap K$, we have $\boldsymbol{\pi}[\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k] \in \mathcal{L}(G_*, A_0) = C_0^\dagger$. That is, $C_0[\boldsymbol{\pi}[\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k]] \subseteq L_*$. Hence the rule is present in $\hat{G}$. ■

Lemmas 3, 5–7 mean that one can construct a right grammar from a small amount of data efficiently.

5.2. Learning algorithm

Our learner for $\mathbb{L}(p, q, r, s)$ is shown as Algorithm 1.

Algorithm 1 $\mathcal{A}(p,q,r,s)$

Data: A sequence of strings $w_1, w_2, \cdots \in L_*$; membership oracle $\mathcal{O}$
Result: A sequence of PMCFGs $G_1, G_2, \cdots \in \mathbb{G}(p,q,r)$
let $D := K := F := \varnothing$; $\hat{G} := \mathcal{G}(K,F)$;
for $n = 1, 2, \ldots$ **do**
  let $D := D \cup \{w_n\}$; $K := \mathrm{Sub}_{\leq p}(D)$;
  if $D \nsubseteq \mathcal{L}(\hat{G})$ **then**
    let $F := \mathrm{Con}_{\leq p,r}(D)$;
  end if
  output $\hat{G} = \mathcal{G}(K,F)$ as G_n;
end for

Lemma 8 *If the current conjecture $\hat{G}$ is such that $L_* \nsubseteq \mathcal{L}(\hat{G})$, then the learner will discard $\hat{G}$ at some point.*

Proof At some point, some element $u \in L_* - \mathcal{L}(\hat{G})$ is given to the learner. The rule $[\![\{x_1\}]\!](u) :-$ is correct but not present in $\hat{G}$. Once the learner gets u, we obtain this rule by $u \in F_0$. ■

Lemma 9 *If $\mathcal{L}(\hat{G}) \nsubseteq L_*$, then the learner will discard $\hat{G}$ at some point.*

Proof The fact $\mathcal{L}(\hat{G}) \nsubseteq L_*$ implies that K is not fiducial on F by Lemma 6. That is, $\hat{G}$ has an incorrect rule $[\![C_0]\!](\boldsymbol{\pi}) :- [\![C_1]\!](\boldsymbol{x}_1), \ldots, [\![C_k]\!](\boldsymbol{x}_k)$ such that $C_0[\boldsymbol{\pi}[C_1^\dagger, \ldots, C_k^\dagger]] \nsubseteq L_*$. At some point the learner will have $D \subseteq L_*$ such that $C_i[\boldsymbol{v}_i] \in D$ and $C_0[\boldsymbol{\pi}[\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k]] \nsubseteq L_*$ for some $\boldsymbol{v}_i \in \mathrm{Sub}_{\leq p}(D)$ for all $i = 1, \ldots, k$. The incorrect rule must be removed by $\boldsymbol{v}_i \in C_i^{(K)}$ for $K = \mathrm{Sub}_{\leq p}(D)$. ■

Theorem 10 *The learner $\mathcal{A}(p,q,r,s)$ identifies $\mathbb{G}(p,q,r,s)$ in the limit.*

Proof Let $L_* \in \mathbb{L}(p,q,r,s)$ be the learning target. By Lemmas 8 and 9, the learner never converges to a wrong hypothesis. It is impossible that the set F is changed infinitely many times because F is monotonically expanded and sometime F will include a characterising set C_A for every nonterminal $A \in N_*$ of a target grammar G_* generating L_* and F_0 will include a word derived using every rule, in which case the learner never updates F any more by Lemma 7. Then sometime K will be fiducial on F by Lemmas 9 and 5, where $\hat{G}$ has no incorrect rules. Thereafter no rules will be added to or removed from $\hat{G}$ any more. ■

6. Hardness of Primal Approach

Among two different types of approaches in distributional learning, this paper takes the so-called *dual* approach for learning PMCFGs. One might expect that the other, called *primal* approach, would work as well. A primal counterpart to the s-FCP could be defined as follows:

> Let us say that a grammar G has the s-FKP if every nonterminal A admits a finite string set K_A of cardinality at most s such that $\pi[K_A] \subseteq \mathcal{L}(G)$ iff $\pi[\mathcal{L}(G, A)] \subseteq \mathcal{L}(G)$ for any context π.

However, the simplest non-linear grammar with the rule set $\{\, S(x_1x_1) :\!- S(x_1),\ S(a) :\!- \,\}$ does not have the 1-FKP, which contrasts with the fact that every grammar with a single nonterminal has the 1-FCP. This grammar still has the 2-FKP, but the authors did not yet find a non-semilinear language generated by a grammar with the 1-FKP.

An even more serious problem is in the hardness of avoiding overgeneralisation while still only using polynomial-time computation (cf. Lemma 6). To get a primal learner for PMCFGs with the s-FKP, it seems a natural idea to combine the techniques proposed in this paper and by Yoshinaka (2011a). Nonterminal symbols should be indexed by string sets K of cardinality at most s. A rule of the form

$$[\![K_0]\!](\boldsymbol{\pi}) :\!- [\![K_1]\!](\boldsymbol{x}_1), \ldots, [\![K_k]\!](\boldsymbol{x}_k)$$

would be said to be *correct* if $\pi_0[\boldsymbol{\pi}[K_1, \ldots, K_k]] \subseteq L_*$ for every π_0 such that $\pi_0[K_0] \subseteq L_*$ where L_* is our learning target. However, to avoid exponential growth of computation time, we have to consider only r-copying contexts as π_0 for some fixed number r. Suppose, for example, $r = 2$ and $L_* = \{a^4, b^4, ad, bd, cd\}$. Since $\pi[\{a, b\}] \subseteq L_*$ iff $\pi[\{c\}] \subseteq L_*$ for every 2-copying context π, though $x_1^4[\{a, b\}] \subseteq L_*$ and $x_1^4[\{c\}] \nsubseteq L_*$, our learner will construct a rule $[\![\{a, b\}]\!](x_1) :\!- [\![\{c\}]\!](x_1)$. Together with other rules $[\![\{a^4, b^4\}]\!](x_1x_1) :\!- [\![\{aa, bb\}]\!](x_1)$, $[\![\{aa, bb\}]\!](x_1x_1) :\!- [\![\{a, b\}]\!](x_1)$ and $[\![\{c\}]\!](c) :\!-$, we can derive $c^4 \notin L_*$. Thus, we can even find a *finite* language that this primal approach fails to learn. In the dual approach we have taken to learn PMCFGs, strings are used to exclude incorrect rules, and gathering all substrings from positive data can be done in polynomial time. On the other hand, in the primal approach, extracting all contexts from positive data is computationally intractable and thus we have to consider only r-copying contexts, though actually strings may be recursively copied any number of times during a derivation process. This limits our ability to control overgeneralisation in the primal approach.

7. Conclusion

In this paper, we have extended distributional learning to the inference of non-semilinear languages. This result also includes as a corollary a significant extension of the learnable classes of MCFGSs where the nonterminals are based on contexts: a dual model in the sense of Yoshinaka (2011a).

The combination of these two extensions gives, for the first time, an efficiently learnable class of languages that plausibly includes all natural languages, even under the worst case that all of the questionable examples in Section 3 are valid; more precisely, a class where there are no arguments that suggest that there is a natural language which is *not* in the class. This leaves open two interesting issues: finding an appropriate learnable feature calculus to represent the large set of nonterminals required, and the more fundamental question of whether these grammars are also strongly adequate: adequate not just in terms of the sets of strings that they generate but in terms of the sets of structural descriptions.

References

Rajesh Bhatt and Aravind K. Joshi. Semilinearity is a syntactic invariant: A reply to Michaelis and Kracht 1997. *Linguistic inquiry*, 35(4):683–692, 2004.

Alexander Clark. Learning context free grammars with the syntactic concept lattice. In Sempere and García (2010), pages 38–51.

Alexander Clark and Rémi Eyraud. Polynomial identification in the limit of substitutable context-free languages. *Journal of Machine Learning Research*, 8:1725–1745, 2007.

Sharon Inkelas and Cheryl Zoll. *Reduplication: doubling in morphology*. Cambridge University Press, 2005.

Aravind K. Joshi, K. Vijay-Shanker, and David J. Weir. The convergence of mildly context-sensitive grammar formalisms. In Peter Sells, Stuart Shieber, and Thomas Wasow, editors, *Foundational Issues in Natural Language Processing*, pages 31–81. MIT Press, 1991.

Gregory M. Kobele. *Generating Copies: An investigation into structural identity in language and grammar*. PhD thesis, University of California Los Angeles, 2006.

Jens Michaelis and Marcus Kracht. Semilinearity as a syntactic invariant. In *Logical aspects of computational linguistics*, pages 329–345. Springer, 1997.

Daniel Radzinski. Chinese number-names, tree adjoining languages, and mild context-sensitivity. *Computational Linguistics*, 17(3):277–299, 1991.

Hiroyuki Seki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229, 1991.

José M. Sempere and Pedro García, editors. *Grammatical Inference: Theoretical Results and Applications, 10th International Colloquium, ICGI 2010*, 2010. Springer.

Stuart M. Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343, 1985.

Raymond M. Smullyan. *Theory of Formal Systems*. Princeton University Press, 1961.

K. Vijay-Shanker, David J. Weir and Aravind K. Joshi. Characterizing Structural Descriptions produced by Various Grammatical Formalisms. In *25th Annual Meeting of the Association for Computational Linguistics*, pages 104–111. ACL, 1987.

Ryo Yoshinaka. Towards dual approaches for learning context-free grammars based on syntactic concept lattices. In *Developments in Language Theory*, pages 429–440. Springer, 2011a.

Ryo Yoshinaka. Polynomial-time identification of multiple context-free languages from positive data and membership queries. In Sempere and García (2010), pages 230–244.

Ryo Yoshinaka. Efficient learning of multiple context-free languages with multidimensional substitutability from positive data. *Theoretical Computer Science*, 412(19):1821–1831, 2011b.