Polynomial-Time Identification of Multiple Context-Free Languages from Positive Data and Membership Queries

Ryo Yoshinaka*

Minato Discrete Structure Manipulation System Project,
Erato, Japan Science and Technology Agency
ryoshinaka@erato.ist.hokudai.ac.jp

Abstract. This paper presents an efficient algorithm that identifies a rich subclass of multiple context-free languages in the limit from positive data and membership queries by observing where each tuple of strings may occur in sentences of the language of the learning target. Our technique is based on Clark et al.'s work (ICGI 2008) on learning of a subclass of context-free languages. Our algorithm learns those context-free languages as well as many non-context-free languages.

1 Introduction

Most approaches for investigating learnable classes of languages assume a fixed form of grammars or automata and then design a learner which computes conjectures in that form. Clark and Eyraud's work [1] of learning *substitutable context-free languages (*CFL*s)* is based on a quite different approach, where the target languages are defined by a purely language theoretic closure property and have no grammatical characterization. Their idea behind is to identify a syntactic category with the set of phrases belonging to that category and moreover with the set of contexts in which those phrases may occur. Then they use contextual information for determining nonterminal symbols and production rules. The literature has already got several results in this line, where the phrase-context relation plays a crucial role. The discussion in this paper should be put in this line of research, and it can particularly be seen as a unification of techniques from Clark et al. [2] and Yoshinaka [3].

Yoshinaka [3] has presented efficient algorithms that identify special kinds of *multiple context-free languages (*MCFL*s)* in the limit from positive data based on the technique by Clark and Eyraud [1] for learning substitutable CFLs. MCFLs are a natural extension of CFLs and they form a representative class of *mildly context-sensitive languages*. Since some natural language phenomena were found not to be context-free, the notion of mildly context-sensitive languages was proposed for better describing natural languages. It is an important and challenging issue

* He is concurrently working in Graduate School of Information Science and Technology, Hokkaido University.

in grammatical inference to find an efficiently learnable class of mildly context-sensitive languages.

While substitutable context-free languages are not very expressive, Clark et al. [2] have shown that a much richer subclass of CFLs is efficiently identifiable in the limit from positive data with the aid of an oracle for membership queries. Actually their learning algorithm runs on *contextual binary feature grammars*, which are another generalization of context-free grammars. Though their algorithm seems to be able to learn some non-context-free languages, they focus on a sufficient condition on CFLs for being learnt and it is not discussed what kind of context-sensitive languages are learnt by their algorithm. In spite of several virtues of contextual binary feature grammars [4], the merit of the formalism is not very clear from a learning theoretical point of view.

This paper proposes algorithms that identify special sorts of MCFLs in the limit from positive data and membership queries which update their conjecture in polynomial time in the size of the given positive data. Our discussion is based on Clark et al.'s [2], but includes an expansion of the class of learnable languages and a refinement of the conditions for learnability. Our algorithms learn complex languages like $\{ a^m b^n c^m d^n \mid 1 \le m \le n \}$ and $\{ w\overline{w} \mid w \in \Sigma^+ \}$.

2 Preliminaries

2.1 Basic Definitions and Notations

The set of non-negative integers is denoted by $\mathbb{N}$ and this paper will consider only numbers in $\mathbb{N}$. The cardinality of a set S is denoted by $|S|$. If w is a string over an alphabet Σ, $|w|$ denotes its length. $\varnothing$ is the empty set and λ is the empty string. Σ^* denotes the set of all strings over Σ. We write $\Sigma^+ = \Sigma^* - \{\lambda\}$, $\Sigma^k = \{ w \in \Sigma^* \mid |w| = k \}$ and $\Sigma^{\le k} = \{ w \in \Sigma^* \mid |w| \le k \}$. Any subset of Σ^* is called a *language (over Σ)*. If L is a finite language, its size is defined as $\|L\| = |L| + \sum_{w \in L} |w|$. An *m-word* is an m-tuple of strings and we denote the set of m-words by $(\Sigma^*)^{\langle m \rangle}$. Similarly we define $(\cdot)^{\langle * \rangle}$, $(\cdot)^{\langle + \rangle}$, $(\cdot)^{\langle \le m \rangle}$. Any m-word is called a *multiword*. Thus $(\Sigma^*)^{\langle * \rangle}$ denotes the set of all multiwords. For $\boldsymbol{w} = \langle w_1, \dots, w_m \rangle \in (\Sigma^*)^{\langle m \rangle}$, $|\boldsymbol{w}|$ denotes its length m and $\|\boldsymbol{w}\|$ denotes its *size* $m + \sum_{1 \le i \le m} |w_i|$. We use the symbol $\Box$, assuming that $\Box \notin \Sigma$, for representing a "hole", which is supposed to be replaced by another string. We write $\Sigma_\Box$ for $\Sigma \cup \{\Box\}$. A string x over $\Sigma_\Box$ is called an *m-context* if x contains m occurrences of $\Box$. m-contexts are also called *multicontexts*. For an m-context $\mathsf{x} = x_0 \Box x_1 \Box \dots \Box x_m$ with $x_0, \dots, x_m \in \Sigma^*$ and an m-word $\boldsymbol{y} = \langle y_1, \dots, y_m \rangle \in (\Sigma_\Box^*)^{\langle m \rangle}$, we define

$$\mathsf{x} \odot \boldsymbol{y} = x_0 y_1 x_1 \dots y_n x_n$$

and say that $\boldsymbol{y}$ is a *sub-multiword* of $\mathsf{x} \odot \boldsymbol{y}$. Note that $\Box$ is the empty context and we have $\Box \odot \langle y \rangle = y$ for any $y \in \Sigma^*$. For $L \subseteq \Sigma^*$ and $p \ge 1$, we define

$$\mathcal{S}^{\le p}(L) = \{ \boldsymbol{y} \in (\Sigma^+)^{\langle \le p \rangle} \mid \mathsf{x} \odot \boldsymbol{y} \in L \text{ for some } \mathsf{x} \in \Sigma_\Box^* \},$$
$$\mathcal{C}^{\le p}(L) = \{ \mathsf{x} \in \Sigma_\Box^* \mid \mathsf{x} \odot \boldsymbol{y} \in L \text{ for some } \boldsymbol{y} \in (\Sigma^+)^{\langle \le p \rangle} \},$$

and for $\boldsymbol{y} \in (\Sigma^*)^{\langle * \rangle}$, we define

$$L/\boldsymbol{y} = \{\, \mathsf{x} \in \Sigma^*_\square \mid \mathsf{x} \odot \boldsymbol{y} \in L \,\}.$$

Computation of $\mathcal{S}^{\leq p}(L)$ and $\mathcal{C}^{\leq p}(L)$ can be done in $O(\|L\|^{2p})$ time if L is finite.

2.2 Linear Regular Functions

Let us suppose a countably infinite set Z of *variables* disjoint from Σ. A function f from $(\Sigma^*)^{\langle m_1 \rangle} \times \cdots \times (\Sigma^*)^{\langle m_n \rangle}$ to $(\Sigma^*)^{\langle m \rangle}$ is said to be *linear regular*, if there is $\langle \alpha_1, \dots, \alpha_m \rangle \in ((\Sigma \cup \{\, z_{ij} \in Z \mid 1 \leq i \leq n,\ 1 \leq j \leq m_i \,\})^*)^{\langle m \rangle}$ such that each variable z_{ij} occurs exactly once in $\langle \alpha_1, \dots, \alpha_m \rangle$ and

$$f(\boldsymbol{w}_1, \dots, \boldsymbol{w}_n) = \langle \alpha_1[\boldsymbol{z} := \boldsymbol{w}], \dots, \alpha_m[\boldsymbol{z} := \boldsymbol{w}] \rangle$$

for any $\boldsymbol{w}_i = \langle w_{i1}, \dots, w_{im_i} \rangle \in (\Sigma^*)^{\langle m_i \rangle}$ with $1 \leq i \leq n$, where $\alpha_k[\boldsymbol{z} := \boldsymbol{w}]$ denotes the string obtained by replacing each variable z_{ij} with the string w_{ij}. We say that f is *λ-free* if no α_k from $\langle \alpha_1, \dots, \alpha_m \rangle$ is λ. If z_{ij} always occurs left of $z_{i(j+1)}$ in $\alpha_1 \dots \alpha_m$ for $1 \leq i \leq n$ and $1 \leq j < m_i$, f is said to be *non-permuting*. The *rank* of f is defined to be $\mathrm{rank}(f) = n$ and the *size* of f is $\mathrm{size}(f) = \|\langle \alpha_1, \dots, \alpha_m \rangle\|$.

Example 1. Among the functions defined below, where $a, b \in \Sigma$, f is not linear regular, while g and h are linear regular. Moreover h is λ-free and non-permuting.

$$\begin{aligned}
f(\langle z_{11}, z_{12} \rangle, \langle z_{21} \rangle) &= \langle z_{11} z_{12}, z_{11} z_{21} z_{11} \rangle, \\
g(\langle z_{11}, z_{12} \rangle, \langle z_{21} \rangle) &= \langle z_{12}, z_{11} b z_{21}, \lambda \rangle, \\
h(\langle z_{11}, z_{12} \rangle, \langle z_{21} \rangle) &= \langle a, z_{11} b z_{21}, z_{12} \rangle.
\end{aligned}$$

Lemma 1. *For any language $L \subseteq \Sigma^*$ and any $\boldsymbol{u}_1, \dots, \boldsymbol{u}_n, \boldsymbol{v}_1, \dots, \boldsymbol{v}_n \in (\Sigma^*)^{\langle * \rangle}$ such that $|\boldsymbol{u}_i| = |\boldsymbol{v}_i|$ and $L/\boldsymbol{u}_i \subseteq L/\boldsymbol{v}_i$ for all i, we have $L/f(\boldsymbol{u}_1, \dots, \boldsymbol{u}_n) \subseteq L/f(\boldsymbol{v}_1, \dots, \boldsymbol{v}_n)$ for any non-permuting linear regular function f.*

Proof. Let $m_i = |\boldsymbol{u}_i| = |\boldsymbol{v}_i|$. Suppose that $\mathsf{x} \in L/f(\boldsymbol{u}_1, \dots, \boldsymbol{u}_n)$, i.e., $\mathsf{x} \odot f(\boldsymbol{u}_1, \dots, \boldsymbol{u}_n) \in L$. The following inference is allowed:

$$\begin{gathered}
\mathsf{x} \odot f(\square^{\langle m_1 \rangle}, \boldsymbol{u}_2, \dots, \boldsymbol{u}_n) \in L/\boldsymbol{u}_1 \subseteq L/\boldsymbol{v}_1 \implies \mathsf{x} \odot f(\boldsymbol{v}_1, \boldsymbol{u}_2, \dots, \boldsymbol{u}_n) \in L \\
\implies \mathsf{x} \odot f(\boldsymbol{v}_1, \square^{\langle m_2 \rangle}, \boldsymbol{u}_3, \dots, \boldsymbol{u}_n) \in L/\boldsymbol{u}_2 \subseteq L/\boldsymbol{v}_2 \implies \\
\mathsf{x} \odot f(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{u}_3, \dots, \boldsymbol{u}_n) \in L \implies \dots \implies \mathsf{x} \odot f(\boldsymbol{v}_1, \dots, \boldsymbol{v}_n) \in L.
\end{gathered}$$

Hence $\mathsf{x} \in L/f(\boldsymbol{v}_1, \dots, \boldsymbol{v}_n)$. □

2.3 Multiple Context-Free Grammars

A *multiple context-free grammar (*MCFG*)* is a tuple $G = \langle \Sigma, V_{\mathrm{dim}}, F, P, I \rangle$, where

– Σ is a finite set of *terminal symbols*,

- $V_{\dim} = \langle V, \dim \rangle$ is the pair of a finite set V of *nonterminal symbols* and a function dim giving a positive integer, called a *dimension*, to each element of V,
- F is a finite set of *linear regular functions*,[1]
- P is a finite set of *rules* of the form $A \to f(B_1, \dots, B_n)$ where $A, B_1, \dots, B_n \in V$ and $f \in F$ maps $(\Sigma^*)^{\langle \dim(B_1) \rangle} \times \dots \times (\Sigma^*)^{\langle \dim(B_n) \rangle}$ to $(\Sigma^*)^{\langle \dim(A) \rangle}$,
- I is a subset of V and all elements of I have dimension 1. Elements of I are called *initial symbols*.

We note that our definition of MCFGs is slightly different from the original [5], where functions from F may delete some arguments (some variable z_{ij} may be absent in $\langle \alpha_1, \dots, \alpha_m \rangle$ in the definition of f in Section 2.2) and grammars have exactly one initial symbol. In fact this modification does not change their generative capacity.

We will simply write V for $V_{\dim}$ if no confusion occurs. If a rule has a function f, then its right hand side must have rank(f) occurrences of nonterminals by definition. If rank$(f) = 0$ and $f() = \boldsymbol{v}$, we may write $A \to \boldsymbol{v}$ instead of $A \to f()$. If rank$(f) = 1$ and f is the identity, we may write $A \to B$ instead of $A \to f(B)$, where $\dim(A) = \dim(B)$. The *size* $\|G\|$ of G is defined as $\|G\| = |P| + \sum_{\rho \in P} \mathrm{size}(\rho)$ where $\mathrm{size}(A \to f(B_1, \dots, B_n)) = \mathrm{size}(f) + n + 1$.

For all $A \in V$ we define $\mathcal{L}(G, A)$ to be the smallest subset of $(\Sigma^*)^{\langle \dim(A) \rangle}$ such that if $A \to f(B_1, \dots, B_n)$ is a rule and $\boldsymbol{w}_i \in \mathcal{L}(G, B_i)$ for $i = 1, \dots, n$, then $f(\boldsymbol{w}_1, \dots, \boldsymbol{w}_n) \in \mathcal{L}(G, A)$. Those series of recursive steps for defining elements of $\mathcal{L}(G, A)$ are called *derivations*. The *language* $\mathcal{L}(G)$ *generated by* G means the set $\{ w \in \Sigma^* \mid \langle w \rangle \in \mathcal{L}(G, S) \text{ with } S \in I \}$, which is called a *multiple context-free language (*MCFL*)*. Two grammars G and G' are *equivalent* if $\mathcal{L}(G) = \mathcal{L}(G')$.

We denote by $\mathbb{G}(p, r)$ the collection of MCFGs G whose nonterminals are assigned a dimension at most p and whose functions have a rank at most r. Then we define $\mathbb{L}(p, r) = \{ \mathcal{L}(G) \mid G \in \mathbb{G}(p, r) \}$. We also write $\mathbb{G}(p, *) = \bigcup_{r \in \mathbb{N}} \mathbb{G}(p, r)$ and $\mathbb{L}(p, *) = \bigcup_{r \in \mathbb{N}} \mathbb{L}(p, r)$. The class of context-free grammars is identified with $\mathbb{G}(1, *)$ and $\mathbb{G}(1, 2)$ corresponds to the class of context-free grammars in Chomsky normal form. Thus $\mathbb{L}(1, 2) = \mathbb{L}(1, *)$.

Example 2. Let G be the MCFG $\langle \Sigma, V, F, P, \{S\} \rangle$ over $\Sigma = \{a, b, c, d\}$ whose rules are

$$\pi_1 : S \to f(A, B) \text{ with } f(\langle z_{11}, z_{12} \rangle, \langle z_{21}, z_{22} \rangle) = \langle z_{11} z_{21} z_{12} z_{22} \rangle,$$
$$\pi_2 : A \to g(A) \text{ with } g(\langle z_1, z_2 \rangle) = \langle a z_1, c z_2 \rangle, \qquad \pi_3 : A \to \langle a, c \rangle,$$
$$\pi_4 : B \to h(B) \text{ with } h(\langle z_1, z_2 \rangle) = \langle z_1 b, z_2 d \rangle, \qquad \pi_5 : B \to \langle b, d \rangle,$$

where $V = \{S, A, B\}$ with $\dim(S) = 1$, $\dim(A) = \dim(B) = 2$, and F consists of f, g, h and the constant functions appearing in the rules π_3 and π_5. For example, $\langle a, c \rangle \in \mathcal{L}(G, A)$ by π_3, $\langle aa, cc \rangle \in \mathcal{L}(G, A)$ by π_2, $\langle b, d \rangle \in \mathcal{L}(G, B)$ by π_5 and $\langle aabccd \rangle \in \mathcal{L}(G, S)$ by π_1. We have $\mathcal{L}(G) = \{ a^m b^n c^m d^n \mid m, n \geq 1 \}$.

[1] We identify a function with its name for convenience.

If all functions of an MCFG G are λ-free, non-permuting, we say that G is *λ-free*, *non-permuting*, respectively. In fact every MCFG in $\mathbb{G}(p,r)$ has an equivalent λ-free and non-permuting one in $\mathbb{G}(p,r)$ modulo λ [5,6]. We assume without loss of generality that all MCFGs are λ-free and non-permuting in this paper.

Seki et al. [5] and Rambow and Satta [7] have investigated the hierarchy of MCFLs.

Proposition 1 (Seki et al. [5], Rambow and Satta [7]).
For $p \geq 1$, $\mathbb{L}(p,*) \subsetneq \mathbb{L}(p+1,*)$.
For $p \geq 2$, $r \geq 1$, $\mathbb{L}(p,r) \subsetneq \mathbb{L}(p,r+1)$ except for $\mathbb{L}(2,2) = \mathbb{L}(2,3)$.
For $p \geq 1$, $r \geq 3$ and $1 \leq k \leq r-2$, $\mathbb{L}(p,r) \subseteq \mathbb{L}((k+1)p, r-k)$.

Theorem 1 (Seki et al. [5], Kaji et al. [8]). *Let p and r be fixed. It is decidable in $O(\|G\|^2|w|^{p(r+1)})$ time whether $w \in \mathcal{L}(G)$ for any* MCFG *$G \in \mathbb{G}(p,r)$ and $w \in \Sigma^*$.*

2.4 Multiple Context-Free Grammars with Finite Kernel Property

Now we introduce subclasses of MCFLs that will be our learning targets.

Definition 1. Let $G = \langle \Sigma, V, F, P, I \rangle \in \mathbb{G}(p,r)$. If each $A \in V$ has an element $\boldsymbol{v}_A \in \mathcal{L}(G,A)$ such that we have $\mathcal{L}(G)/\boldsymbol{v}_A \subseteq \mathcal{L}(G)/\boldsymbol{w}$ for all $\boldsymbol{w} \in \mathcal{L}(G,A)$, then we say that G has the *finite kernel property (*FKP*).*

We let $\mathbb{KG}(p,r)$ denote the class of MCFGs with the FKP in $\mathbb{G}(p,r)$, and $\mathbb{KL}(p,r)$ the corresponding class of languages.

When $p=1$ and $r \geq 2$, this definition is equivalent to Clark et al.'s [2] definition for CFGs in Chomsky normal form. It is not difficult to see that $\mathbb{KL}(1,2) = \bigcup_{r\in\mathbb{N}} \mathbb{KL}(1,r)$ and that $\mathbb{KG}(1,1)$ includes all regular languages [2].

Example 3. Let us consider the grammar $G \in \mathbb{G}(2,1)$ with the following rules:

$$\begin{aligned}
&S \to f(A) \text{ with } f(\langle z_1, z_2\rangle) = \langle z_1 z_2 \rangle,\\
&A \to g(A) \text{ with } g(\langle z_1, z_2\rangle) = \langle z_1 b, z_2 d \rangle,\\
&A \to h(A) \text{ with } h(\langle z_1, z_2\rangle) = \langle a z_1 b, c z_2 d \rangle,\\
&A \to \langle b, d \rangle,
\end{aligned}$$

where S is the only initial symbol of G. The generated language is $\mathcal{L}(G) = \{\, a^m b^n c^m d^n \mid m < n \,\}$. For each $\langle a^m b^n, c^m d^n \rangle \in \mathcal{L}(G,A)$ with $m < n$, we have

$$\begin{aligned}
\mathcal{L}(G)/\langle b^n, d^n \rangle &= \{\, a^i b^{j_1} \Box b^{j_2} c^i d^{j_3} \Box d^{j_4} \mid i \leq n + j_1 + j_2 = n + j_3 + j_4 \,\},\\
\mathcal{L}(G)/\langle a^m b^n, c^m d^n \rangle &= \{\, a^i \Box b^j c^i \Box d^j \mid i \leq n - m + j \,\} \quad \text{for } m \geq 1.
\end{aligned}$$

Thus we have $\mathcal{L}(G)/\langle abb, cdd \rangle \subseteq \mathcal{L}(G)/\langle a^m b^n, c^m d^n \rangle$ for any $m, n \in \mathbb{N}$ with $m < n$. Clearly $\mathcal{L}(G)/\langle w \rangle = \Box$ for any $\langle w \rangle \in \mathcal{L}(G,S)$. Therefore $G \in \mathbb{KG}(2,1)$.

Another example with the FKP is $\{\, wcwc \mid w \in \{a,b\}^* \,\} \in \mathbb{KL}(2,1)$. On the other hand, the language $\{\, a^n b^n \mid n \geq 1 \,\} \cup \{\, a^n b^{2n} \mid n \geq 1 \,\}$ does not have the FKP.

3 Learning Multiple Context-Free Languages with the Finite Kernel Property

3.1 Hypotheses

Hereafter we arbitrarily fix two natural numbers $p \geq 1$ and $r \geq 1$. Our learning algorithm $\mathcal{A}(p,r)$ computes grammars in $\mathbb{G}(p,r)$ using positive data and membership queries. Let $L_* \subseteq \Sigma^*$ be the target language belonging to $\mathbb{KL}(p,r)$. The hypothesized grammar is defined by three parameters $K \subseteq \mathcal{S}^{\leq p}(L_*)$, $X \subseteq \mathcal{C}^{\leq p}(L_*)$ and L_*. K and X are finite sets computed from given positive data. Of course $\mathcal{A}(p,r)$ cannot take L_* as a part of input, but in fact a finite number of membership queries is enough to construct the following MCFG $\mathcal{G}^r(K,X,L_*) = \langle \Sigma, V, F, P, I \rangle$. The set of nonterminal symbols is $V = K$ and we will write $[\![\boldsymbol{v}]\!]$ instead of $\boldsymbol{v}$ for clarifying that it means a nonterminal symbol (indexed with $\boldsymbol{v}$). The dimension $\dim([\![\boldsymbol{v}]\!])$ is $|\boldsymbol{v}|$. The set of initial symbols is

$$I = \{\, [\![\langle w \rangle]\!] \mid \langle w \rangle \in K \text{ and } w \in L_* \,\},$$

where each element of I has dimension 1. The rules of P are divided into the following two types:

- (Type I) $[\![\boldsymbol{v}]\!] \to f([\![\boldsymbol{v}_1]\!], \dots, [\![\boldsymbol{v}_n]\!])$, if $0 \leq n \leq r$, $\boldsymbol{v}, \boldsymbol{v}_1, \dots, \boldsymbol{v}_n \in K$ and $\boldsymbol{v} = f(\boldsymbol{v}_1, \dots, \boldsymbol{v}_n)$ for f λ-free and non-permuting;
- (Type II) $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$, if $L_*/\boldsymbol{u} \cap X \subseteq L_*/\boldsymbol{v} \cap X$ and $\boldsymbol{u}, \boldsymbol{v} \in K$,

where rules of the form $[\![\boldsymbol{v}]\!] \to [\![\boldsymbol{v}]\!]$ are of Type I and Type II at the same time, but they are anyway meaningless. F is the set of λ-free and non-permuting functions that appear in the definition of P.

We want each nonterminal symbol $[\![\boldsymbol{v}]\!]$ to derive $\boldsymbol{v}$. Here the construction of I appears to be trivial: initial symbols derive elements of L_* that appear in K. This property is realized by the rules of Type I. For example, for $p = r = 2$ and $K = \mathcal{S}^{\leq 2}(\{\, ab\})$, one has the following rules $\pi_1, \dots, \pi_5$ of Type I that have $[\![\langle a, b \rangle]\!]$ on its left hand side:

$$\begin{aligned}
&\pi_1 : [\![\langle a,b \rangle]\!] \to \langle a,b \rangle, \\
&\pi_2 : [\![\langle a,b \rangle]\!] \to f_a([\![\langle b \rangle]\!]) \quad \text{with} \quad f_a(\langle z \rangle) = \langle a, z \rangle, \\
&\pi_3 : [\![\langle a,b \rangle]\!] \to f_b([\![\langle a \rangle]\!]) \quad \text{with} \quad f_b(\langle z \rangle) = \langle z, b \rangle, \\
&\pi_4 : [\![\langle a,b \rangle]\!] \to g([\![\langle a \rangle]\!], [\![\langle b \rangle]\!]) \quad \text{with} \quad g(\langle z_1 \rangle, \langle z_2 \rangle) = \langle z_1, z_2 \rangle, \\
&\pi_5 : [\![\langle a,b \rangle]\!] \to [\![\langle a,b \rangle]\!],
\end{aligned}$$

where π_1 indeed derives $\langle a, b \rangle$, while π_5 is meaningless. Instead of deriving $\langle a, b \rangle$ directly by π_1, one can derive it by two steps with π_3 and $\pi_6 : [\![\langle a \rangle]\!] \to \langle a \rangle$ (or π_2 and $\pi_7 : [\![\langle b \rangle]\!] \to \langle b \rangle$), or by three steps by π_4, π_6 and π_7. One may regard application of rules of Type I as a decomposition of the multiword that appears on its left hand side. It is easy to see that there are finitely many rules of Type I, because K is finite and nonterminals on the right hand side of a rule are all

λ-free sub-multiwords of that on the left hand side. If the grammar had only rules of Type I, then it should derive all and only elements of I.

Now we explain the intuition behind rules of Type II. Suppose that $L_*/\boldsymbol{u} \subseteq L_*/\boldsymbol{v}$. This means that L_* is closed under substituting $\boldsymbol{v}$ for $\boldsymbol{u}$, and such substitution is realized by the rule $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ of Type II. However the algorithm cannot check whether $L_*/\boldsymbol{u} \subseteq L_*/\boldsymbol{v}$ in finitely many steps, while it can approximate this relation with $L_*/\boldsymbol{u} \cap X \subseteq L_*/\boldsymbol{v} \cap X$ by membership queries, because X is finite. Clearly $L_*/\boldsymbol{u} \subseteq L_*/\boldsymbol{v}$ implies that $L_*/\boldsymbol{u} \cap X \subseteq L_*/\boldsymbol{v} \cap X$, but the inverse is not true. We say that a rule $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ of Type II is *wrong (with respect to L_*)* if $L_*/\boldsymbol{u} \nsubseteq L_*/\boldsymbol{v}$. It might often happen that $L_*/\boldsymbol{u} \cap X = L_*/\boldsymbol{v} \cap X$ and in this case we have symmetric rules $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ and $[\![\boldsymbol{v}]\!] \to [\![\boldsymbol{u}]\!]$ and then trivially they derive the same set of multiwords. One can merge those nonterminals to compact the grammar.

Computation process of rules of Type II can be handled by a collection of matrices, called *observation tables*. For each dimension $m \le p$, we have an observation table T_m. Let K_m and X_m be the sets of m-words from K and m-contexts from X, respectively. The rows of the table T_m are indexed with just elements of K_m and the columns are indexed with elements of X_m. For each pair $\boldsymbol{u}, \boldsymbol{v} \in K_m$, to compare the sets $L_*/\boldsymbol{u} \cap X$ and $L_*/\boldsymbol{v} \cap X$, one needs to know whether $\mathbf{x} \odot \boldsymbol{v} \in L_*$ or not for all of $\mathbf{x} \in X_m$. The membership of $\mathbf{x} \odot \boldsymbol{v}$ is recorded in the corresponding entry of the observation table with the aid of a membership query. By comparing the entries of the rows corresponding to $\boldsymbol{u}$ and $\boldsymbol{v}$, one can determine whether the grammar should have the rule $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$.

Note that the initial symbols are determined by K and L_* and the rules of Type I are constructed solely by K, while X is used only for determining rules of Type II. As Clark et al.'s [2] learning algorithm does, our algorithm $\mathcal{A}(p, r)$ monotonically expands K and X, where the set I and rules of Type I monotonically increase, while wrong rules of Type II may be deleted.

Example 4. Let $p = 2$, $r = 1$, $L = \{\, a^m b^n c^m d^n \mid 1 \le m \le n \,\} \in \mathbb{L}(2, 1)$, $K = \{\langle aabbccdd\rangle, \langle aabb, ccdd\rangle, \langle aab, ccd\rangle, \langle abb, cdd\rangle, \langle ab, cd\rangle\})$ and $X = \{\Box, a\Box c\Box, \Box b\Box d, a\Box bc\Box d\}$. The computed grammar $\hat{G} = \mathcal{G}^1(K, X, L)$ has the following rules of Type I:

$$\begin{array}{lll}
\pi_0: & [\![\langle aabbccdd\rangle]\!] \to f_0([\![\langle aabb, ccdd\rangle]\!]) & \text{with} \quad f_0(\langle z_1, z_2\rangle) = \langle z_1 z_2\rangle, \\
\pi_1: & [\![\langle aabb, ccdd\rangle]\!] \to f_{abcd}([\![\langle ab, cd\rangle]\!]) & \text{with} \quad f_{abcd}(\langle z_1, z_2\rangle) = \langle a z_1 b, c z_2 d\rangle, \\
\pi_2: & [\![\langle aabb, ccdd\rangle]\!] \to f_{bd}([\![\langle aab, ccd\rangle]\!]) & \text{with} \quad f_{bd}(\langle z_1, z_2\rangle) = \langle z_1 b, z_2 d\rangle, \\
\pi_3: & [\![\langle aabb, ccdd\rangle]\!] \to f_{ac}([\![\langle abb, cdd\rangle]\!]) & \text{with} \quad f_{ac}(\langle z_1, z_2\rangle) = \langle a z_1, c z_2\rangle, \\
\pi_4: & [\![\langle aabbccdd\rangle]\!] \to (f_0 \circ f_{abcd})([\![\langle ab, cd\rangle]\!])), & \\
\pi_5: & [\![\langle aabbccdd\rangle]\!] \to (f_0 \circ f_{bd})([\![\langle aab, ccd\rangle]\!]), & \\
\pi_6: & [\![\langle aabbccdd\rangle]\!] \to (f_0 \circ f_{ac})([\![\langle abb, cdd\rangle]\!]), & \\
\pi_7: & [\![\langle aab, ccd\rangle]\!] \to f_{ac}([\![\langle ab, cd\rangle]\!]), & \pi_8: \quad [\![\langle abb, cdd\rangle]\!] \to f_{bd}([\![\langle ab, cd\rangle]\!]), \\
\pi_9: & [\![\langle aabbccdd\rangle]\!] \to \langle aabbccdd\rangle, & \pi_{10}: \quad [\![\langle aabb, ccdd\rangle]\!] \to \langle aabb, ccdd\rangle, \\
\pi_{11}: & [\![\langle aab, ccd\rangle]\!] \to \langle aab, ccd\rangle, & \pi_{12}: \quad [\![\langle abb, cdd\rangle]\!] \to \langle abb, cdd\rangle, \\
\pi_{13}: & [\![\langle ab, cd\rangle]\!] \to \langle ab, cd\rangle &
\end{array}$$

Here we ignore all meaningless rules of the form $[\![\boldsymbol{v}]\!] \to [\![\boldsymbol{v}]\!]$.

We have the following observation tables T_1 and T_2:

T_1	$\Box$
$\langle aabbccdd \rangle$	1

T_2	$\Box\Box$	$a\Box c\Box$	$\Box b\Box d$	$a\Box bc\Box d$
$\langle aabb, ccdd \rangle$	1	0	1	1
$\langle ab, cd \rangle$	1	0	1	1
$\langle aab, ccd \rangle$	0	0	1	0
$\langle abb, cdd \rangle$	1	1	1	1

which induce the following rules of Type II:

$$\rho_1 : [\![\langle ab, cd \rangle]\!] \to [\![\langle abb, cdd \rangle]\!], \qquad \rho_2 : [\![\langle aab, ccd \rangle]\!] \to [\![\langle ab, cd \rangle]\!],$$
$$\rho_3 : [\![\langle aabb, ccdd \rangle]\!] \to [\![\langle ab, cd \rangle]\!], \qquad \rho_4 : [\![\langle ab, cd \rangle]\!] \to [\![\langle aabb, ccdd \rangle]\!].$$

Note that while ρ_3 and ρ_4 are symmetric, $\hat{G}$ does not have the rules symmetric to ρ_1 nor ρ_2.

Let us see that $a^m b^n c^m d^n \in \mathcal{L}(\hat{G})$ for any m, n if $1 \leq m \leq n$. Trivially $\langle ab, cd \rangle \in \mathcal{L}(\hat{G}, [\![\langle ab, cd \rangle]\!])$ by π_{13}. By repeatedly applying the rules π_1 and ρ_4, one gets $\langle a^m b^m, c^m d^m \rangle \in \mathcal{L}(\hat{G}, [\![\langle ab, cd \rangle]\!])$ for all $m \geq 1$. Then alternating applications of the rules π_8 and ρ_1 give us $\langle a^m b^n, c^m d^n \rangle \in \mathcal{L}(\hat{G}, [\![\langle ab, cd \rangle]\!])$ for all $n \geq m$. The rules ρ_3 and π_0 give $\langle a^m b^n c^m d^n \rangle \in \mathcal{L}(\hat{G}, [\![\langle aabbccdd \rangle]\!])$. Since $[\![\langle aabbccdd \rangle]\!]$ is an initial symbol of $\hat{G}$, we conclude that $a^m b^n c^m d^n \in \mathcal{L}(\hat{G})$. On the other hand, it is not hard to see that $a^m b^n c^m d^n \notin \mathcal{L}(\hat{G})$ if $m > n$.

Before giving the overall structure of the algorithm $\mathcal{A}(p, r)$, we discuss properties of $\mathcal{G}^r(K, X, L)$, where L can be an arbitrary language over Σ.

3.2 Properties of Hypotheses

Lemma 2. *Let $G = \mathcal{G}^r(K, X, L)$ and $G' = \mathcal{G}^r(K', X, L)$. If $K \subseteq K'$, then $\mathcal{L}(G) \subseteq \mathcal{L}(G')$.*

Proof. By definition, every rule of G is also a rule of G' and every initial symbol of G is also an initial symbol of G'. $\Box$

Lemma 3. *Let $G = \mathcal{G}^r(K, X, L)$ and $G' = \mathcal{G}^r(K, X', L)$. If $X \subseteq X'$, then $\mathcal{L}(G') \subseteq \mathcal{L}(G)$.*

Proof. Clearly G and G' share the same set of initial symbols and the same rules of Type I. If G' has a rule $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ of Type II, we have $L/\boldsymbol{u} \cap X' \subseteq L/\boldsymbol{v} \cap X'$. By $X \subseteq X'$, we have $L/\boldsymbol{u} \cap X \subseteq L/\boldsymbol{v} \cap X$ and thus $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ is a rule of G, too. $\Box$

Definition 2. $X \subseteq \Sigma^*_\Box$ is said to be *fiducial on* $K \subseteq (\Sigma^*)^{\langle * \rangle}$ *with respect to* L if $L/\boldsymbol{u} \cap X \subseteq L/\boldsymbol{v} \cap X$ implies $L/\boldsymbol{u} \subseteq L/\boldsymbol{v}$ for any $\boldsymbol{u}, \boldsymbol{v} \in K$.

By definition $G = \mathcal{G}^r(K, X, L)$ has no wrong rules of Type II if and only if X is fiducial on K. In order to get rid of wrong rules from the conjecture, our algorithm wants a fiducial set. If X is fiducial on K with respect to L and $X \subseteq X'$, then X' is also fiducial on K and we have $\mathcal{G}^r(K, X, L) = \mathcal{G}^r(K, X', L)$. The following lemma is easy.

Lemma 4. *Every pair of K and L admits a finite fiducial set X whose cardinality is at most $|K|^2$.*

Proof. For each pair $\boldsymbol{u}, \boldsymbol{v} \in K$ such that $L/\boldsymbol{u} \not\subseteq L/\boldsymbol{v}$, there is $\mathsf{x} \in L/\boldsymbol{u} - L/\boldsymbol{v}$. By collecting all such x, a finite fiducial set X_K on K is obtained. □

Lemma 5. *Suppose that X is fiducial on K with respect to L and let $G = \mathcal{G}^r(K, X, L)$. If $\boldsymbol{w} \in \mathcal{L}(G, [\![\boldsymbol{v}]\!])$, then $L/\boldsymbol{v} \subseteq L/\boldsymbol{w}$ for any $\boldsymbol{v} \in K$.*

Proof. We show the lemma by induction on the derivation of $\boldsymbol{w}$. Suppose that the fact $\boldsymbol{w} \in \mathcal{L}(G, [\![\boldsymbol{v}]\!])$ is due to a rule of Type I of the form $[\![\boldsymbol{v}]\!] \to f([\![\boldsymbol{v}_1]\!], \dots, [\![\boldsymbol{v}_n]\!])$ and $\boldsymbol{w} = f(\boldsymbol{w}_1, \dots, \boldsymbol{w}_n)$ for $\boldsymbol{w}_i \in \mathcal{L}(G, [\![\boldsymbol{v}_i]\!])$ with $i = 1, \dots, n$. By induction hypothesis, we have $L/\boldsymbol{v}_i \subseteq L/\boldsymbol{w}_i$ for $i = 1, \dots, n$. Applying Lemma 1, we get $L/\boldsymbol{v} \subseteq L/\boldsymbol{w}$.

Suppose that the fact $\boldsymbol{w} \in \mathcal{L}(G, [\![\boldsymbol{v}]\!])$ is due to a rule of Type II of the form $[\![\boldsymbol{v}]\!] \to [\![\boldsymbol{u}]\!]$ and $\boldsymbol{w} \in \mathcal{L}(G, [\![\boldsymbol{u}]\!])$. By the presence of the rule, we have $\boldsymbol{v}, \boldsymbol{u} \in K$ and $L/\boldsymbol{v} \cap X \subseteq L/\boldsymbol{u} \cap X$, which implies $L/\boldsymbol{v} \subseteq L/\boldsymbol{u}$ by the fiduciality of X. Together with the induction hypothesis that $L/\boldsymbol{u} \subseteq L/\boldsymbol{w}$, we get $L/\boldsymbol{v} \subseteq L/\boldsymbol{w}$. □

Corollary 1. *If X is fiducial on K with respect to L, then $\mathcal{L}(G) \subseteq L$ for $G = \mathcal{G}^r(K, X, L)$.*

Proof. If $w \in \mathcal{L}(G)$, then G has an initial symbol $[\![\langle v \rangle]\!]$ such that $\langle w \rangle \in \mathcal{L}(G, [\![\langle v \rangle]\!])$ and $v \in L$. Then $\square \in L/\langle v \rangle \subseteq L/\langle w \rangle$ by Lemma 5, i.e., $w \in L$. □

Corollary 1 establishes that a fiducial set indeed ensures that the grammar does not overgeneralize. We now turn our attention to a condition for preventing undergeneralization.

Definition 3. A finite set $K \subseteq (\Sigma^*)^{\langle \le p \rangle}$ is said to be a *p, r-kernel* of a language $L \subseteq \Sigma^*$, if $L \subseteq \mathcal{L}(\mathcal{G}^r(K, X, L))$ for any finite set $X \in \Sigma^*_\square$ of multicontext.

Lemma 6. *$K \subseteq \Sigma^{\langle \le p \rangle}$ is a p, r-kernel of L if and only if $L = \mathcal{L}(\mathcal{G}^r(K, X_*, L))$ for a fiducial set X_* on K.*

Proof. The "only if" part is trivial by Definition 3 and Corollary 1.

Conversely suppose that $L = \mathcal{L}(\mathcal{G}^r(K, X_*, L))$ for a fiducial set X_* on K. Let $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ be a rule of Type II from $\mathcal{G}^r(K, X_*, L)$. Then, for an arbitrary $X \subseteq \Sigma^*_\square$,

$$L/\boldsymbol{u} \cap X_* \subseteq L/\boldsymbol{v} \cap X_* \implies L/\boldsymbol{u} \subseteq L/\boldsymbol{v} \implies L/\boldsymbol{u} \cap X \subseteq L/\boldsymbol{v} \cap X.$$

Thus every rule of $\mathcal{G}^r(K, X_*, L)$ belongs to $\mathcal{G}^r(K, X, L)$, too. That is, $L = \mathcal{L}(\mathcal{G}^r(K, X_*, L)) \subseteq \mathcal{L}(\mathcal{G}^r(K, X, L))$. □

Proposition 2. *A language L has a finite p, r-kernel if and only if $L \in \mathbb{KL}(p, r)$.*

Proof. First suppose that L has a finite p, r-kernel K and let $\hat{G} = \mathcal{G}^r(K, X, L)$ where X is a fiducial set X on K with respect to L, where $L = \mathcal{L}(\hat{G})$ by Lemma 6.

We show that $\hat{G}$ has the FKP. For each nonterminal symbol $[\![\boldsymbol{v}]\!]$ of $\hat{G}$, Lemma 5 claims that $\boldsymbol{v} \in \mathcal{L}(G, [\![\boldsymbol{v}]\!])$ satisfies Definition 1.

Conversely suppose that we have an MCFG $G = \langle \Sigma, V, F, P, I_0 \rangle \in \mathbb{KG}(p, r)$ such that $\mathcal{L}(G) = L$. Let $\boldsymbol{v}_A \in \mathcal{L}(G, A)$ satisfy Definition 1. We show that

$$K = \{\, \boldsymbol{v}_A \mid A \in V \,\} \cup \{\, f(\boldsymbol{v}_{B_1}, \dots, \boldsymbol{v}_{B_n}) \mid A \to f(B_1, \dots, B_n) \in P \,\}$$

is a p, r-kernel of L. Let $G_X = \mathcal{G}^r(K, X, L)$ for $X \subseteq \Sigma^*_\square$. By induction on derivation we show that if $\boldsymbol{w} \in \mathcal{L}(G, A)$ then $\boldsymbol{w} \in \mathcal{L}(G_X, [\![\boldsymbol{v}_A]\!])$. Then in particular when $\langle w \rangle \in \mathcal{L}(G, A)$ with $A \in I_0$, we have $\langle w \rangle \in \mathcal{L}(G_X, [\![\langle v_A \rangle]\!])$, where $[\![\langle v_A \rangle]\!] \in I$ by $v_A \in \mathcal{L}(G)$. That is, $w \in \mathcal{L}(G_X)$.

Suppose that $\boldsymbol{w} \in \mathcal{L}(G, A)$ due to $A \to f(B_1, \dots, B_n) \in P$, $\boldsymbol{w}_i \in \mathcal{L}(G, B_i)$ and $\boldsymbol{w} = f(\boldsymbol{w}_1, \dots, \boldsymbol{w}_n)$. We have $\boldsymbol{w}_i \in \mathcal{L}(G_X, [\![\boldsymbol{v}_{B_i}]\!])$ by induction hypothesis. By definition $\boldsymbol{u} = f([\![\boldsymbol{v}_{B_1}]\!], \dots, [\![\boldsymbol{v}_{B_n}]\!]) \in K \cap \mathcal{L}(G, A)$. We have $L/\boldsymbol{v}_A \subseteq L/\boldsymbol{u}$ by the assumption. Hence G_X has the rules $[\![\boldsymbol{u}]\!] \to f([\![\boldsymbol{v}_{B_1}]\!], \dots, [\![\boldsymbol{v}_{B_n}]\!])$ of Type I and $[\![\boldsymbol{v}_A]\!] \to [\![\boldsymbol{u}]\!]$ of Type II, and thus $\boldsymbol{w} \in \mathcal{L}(G_X, [\![\boldsymbol{v}_A]\!])$. □

3.3 Learning Algorithm

Our algorithm $\mathcal{A}(p, r)$, shown in Algorithm 1, wants a p, r-kernel of the target language L_* and a fiducial set on the kernel, from which it can compute a grammar generating the target language. Expanding X infinitely causes no problem, because it is used only for removing wrong rules. On the other hand, expansion of K means increment of nonterminal symbols and production rules, which should not happen infinitely many times. Thus $\mathcal{A}(p, r)$ expands K only when it knows that K is not a p, r-kernel of L_*, i.e., when it observes that it is undergenerating.

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

```
Data: A sequence of strings $w_1, w_2, \dots \in L_*$; membership oracle $\mathcal{O}$
Result: A sequence of MCFGs $G_1, G_2, \dots \in \mathbb{G}(p, r)$
let $D := \varnothing$; $K := \varnothing$; $X := \varnothing$; $\hat{G} := \mathcal{G}^0(K, X, L_*)$;
for $n = 1, 2, \dots$ do
   let $D := D \cup \{w_n\}$; $X := \mathcal{C}^{\le p}(D)$;
   if $D \not\subseteq \mathcal{L}(\hat{G})$ then
      let $K := \mathcal{S}^{\le p}(D)$;
   end if
   output $\hat{G} = \mathcal{G}^r(K, X, L_*)$ as $G_n$;
end for
```

Lemma 7. *If the current conjecture $\hat{G}$ is such that $L_* \not\subseteq \mathcal{L}(\hat{G})$, then the learner will discard $\hat{G}$ at some point.*

Proof. At some point, some element from $L_* - \mathcal{L}(\hat{G})$ is given to the learner. □

Lemma 8. *If $\mathcal{L}(\hat{G}) \not\subseteq L_*$, then the learner will discard $\hat{G}$ at some point.*

Proof. The fact $\mathcal{L}(\hat{G}) \nsubseteq L_*$ implies that X is not fiducial on K by Corollary 1. That is, $\hat{G}$ has a wrong rule $[\![\boldsymbol{u}]\!] \to [\![\boldsymbol{v}]\!]$ of Type II, where $L_*/\boldsymbol{u} \nsubseteq L_*/\boldsymbol{v}$. At some point the learner will have $D \subseteq L_*$ such that $\mathbf{x} \odot \boldsymbol{u} \in D$ and $\mathbf{x} \odot \boldsymbol{v} \notin L$ for some $\mathbf{x} \in L_*/\boldsymbol{u} - L_*/\boldsymbol{v}$. The wrong rule must be removed. □

Theorem 2. *The learner $\mathcal{A}(p, r)$ identifies $\mathbb{KL}(p, r)$ in the limit.*

Proof. Let $L_* \in \mathbb{KL}(p, r)$ be the learning target. By Lemmas 7 and 8, the learner never converges to a wrong hypothesis. It is impossible that the set K is changed infinitely many times, because K is monotonically expanded and sometime K will be a p, r-kernel of L_*, in which case the learner never updates K any more by Definition 3. Then sometime X will be fiducial on the p, r-kernel K by Lemmas 8 and 4, where $\hat{G}$ has no wrong rules of Type II. Thereafter no rules will be added to or removed from $\hat{G}$ any more. □

The converse does hold. If $\mathcal{A}(p, r)$ converges to a grammar $\hat{G} = \mathcal{G}^r(K, X, L_*)$ such that $\mathcal{L}(\hat{G}) = L_*$, then K is a p, r-kernel of L_* by definition.

Lemma 9. *The algorithm updates its conjecture in polynomial time in $\|D\|$, where the degree of the polynomial is linear in pr.*

Proof. Let us think about computing $\mathcal{G}^r(K, X, L_*)$. By $K \subseteq \mathcal{S}^{\le p}(D)$ and $X \subseteq \mathcal{C}^{\le p}(D)$, $\|K\|$ and $\|X\|$ are bounded by a polynomial in $\|D\|$ with a degree linear in p.

We first estimate the number of rules of Type I that have a fixed $[\![\boldsymbol{v}]\!]$ on the left hand side, which are of the form $[\![\boldsymbol{v}]\!] \to f([\![\boldsymbol{v}_1]\!], \dots, [\![\boldsymbol{v}_n]\!])$. Roughly speaking, this is the number of ways to decompose $\boldsymbol{v}$ into sub-multiwords $\boldsymbol{v}_1, \dots, \boldsymbol{v}_n$ with the aid of a linear regular function f. Once one determines where the occurrence of each substring from $\boldsymbol{v}_1, \dots, \boldsymbol{v}_n$ starts and ends in $\boldsymbol{v}$, the function f is uniquely determined. We have at most pr substrings from $\boldsymbol{v}_1, \dots, \boldsymbol{v}_n$, hence the number of ways to determine such starting and ending points are at most $\|\boldsymbol{v}\|^{2pr}$. Thus, the number of rules of Type I is at most $O(|K|\ell^{2pr})$ where ℓ is the maximal size of elements of K. Clearly the description size of each rule is at most $O(\ell)$.

One can construct the observation tables by calling the oracle at most $|K||X|$ times. Then one can determine initial symbols and rules of Type II in polynomial time. Clearly the size $\|\hat{G}\|$ of $\hat{G}$ is bounded by a polynomial, whose degree is linear in pr, in $\|D\|$.

For checking whether $D \subseteq \mathcal{L}(\hat{G})$, it takes $O(\|\hat{G}\|^2\|D\|^{p(r+1)})$ steps by Theorem 1.

All in all, the algorithm updates its conjecture in polynomial time in $\|D\|$, where the degree of the polynomial is linear in pr. □

We have assumed that the parameters p and r are known to the learner a priori, but it is not mandatory for identifiability, if we give up the polynomial-time update. By assuming $p = r = \max\{\, |w| \mid w \in D \,\}$, the algorithm can compute its conjecture.

3.4 Slight Enhancement

Our algorithm $\mathcal{A}(p, r)$ learns all regular languages and some CFLs and MCFLs. As Clark et al. [2] have discussed, the language $L_0 = a^+ \cup \{ a^n b^n \mid n > 0 \}$ is not learnt by $\mathcal{A}(1, r)$. This is because, to derive a^n for infinitely many n, the conjectured grammar must have a rule of Type II of the form $[\![a^m]\!] \to [\![a^{m+k}]\!]$ for some positive integers m and k. However, as $\square b^m \in L_0/\langle a^m \rangle - L_0/\langle a^{m+k} \rangle$, such a rule should be discarded at some point. The algorithm does not converge to a correct grammar. It is not hard to modify the algorithm $\mathcal{A}(1, r)$ to learn L_0 by using the fact that $\$L_0\$ = \$a^+\$ \cup \{ \$a^n b^n\$ \mid n > 0 \}$ is identifiable by $\mathcal{A}(1, r)$, where \$ is a new terminal symbol not in Σ.

Our alternative algorithm $\mathcal{B}(p, r)$ is a slight modification of $\mathcal{A}(p, r)$. It behaves as if it wants to learn $\$L_*\$$ instead of the target L_* by assuming the special marker \$ at the head and the tail of each positive example. It is obvious that checking whether or not $w \in \$L_*\$$ is possible by using a membership query for L_*. The new algorithm $\mathcal{B}(p, r)$ identifies languages such as L_0 (with $p = 1$, $r = 1$) and $\{ w\overline{w} \mid w \in \{a, b\}^+ \}$ (with $p = 2$, $r = 1$), which $\mathcal{A}(p, r)$ does not learn. Still languages like $\{ a^n b^n \mid n > 0 \} \cup \{ a^n b^{2n} \mid n > 0 \}$ or $\{ ww \mid w \in \{a, b\}^+ \}$ are not learnt by $\mathcal{B}(p, r)$.

Example 5. Let $L = \{ w\#w \mid w \in \{a, b\}^* \} \in \mathbb{L}(2, 1)$, $D = \{a\#a, b\#b\} \subseteq L$, $K_\$ = \mathcal{S}^{\leq 2}(\$D\$)$ and $X_\$ = \mathcal{C}^{\leq 2}(\$D\$)$. The following table is a fragment of the observation table T_2:

T_2	$\$\square\#\square\$$	$\square\#\square\$$	$\$\square\$$	$\square a\#a\square$	$\square b\#b\square$	$\square\square\$$
$\langle a, a \rangle$	1	0	0	0	0	0
$\langle b, b \rangle$	1	0	0	0	0	0
$\langle \$a, a \rangle$	0	1	0	0	0	0
$\langle \$b, b \rangle$	0	1	0	0	0	0
$\langle a, \#a \rangle$	0	0	1	0	0	0
$\langle b, \#b \rangle$	0	0	1	0	0	0
$\langle \$, \$ \rangle$	0	0	0	1	1	0
$\langle \$a, a\$ \rangle$	0	0	0	1	0	0
$\langle \$b, b\$ \rangle$	0	0	0	0	1	0
$\langle \$, \# \rangle$	0	0	0	0	0	1
$\langle \$a, \#a \rangle$	0	0	0	0	0	1
$\langle \$b, \#b \rangle$	0	0	0	0	0	1

The conjectured grammar $\hat{G} = \mathcal{G}^1_\$(K_\$, X_\$, L)$ by $\mathcal{B}(2, 1)$ generates all the elements of L by using the rules of Type II among $[\![\langle \$, \# \rangle]\!]$, $[\![\langle \$a, \#a \rangle]\!]$ and $[\![\langle \$b, \#b \rangle]\!]$, which are not wrong, and the rules of Type I: $[\![\langle \$a, \#a \rangle]\!] \to f_a([\![\langle \$, \# \rangle]\!])$ with $f_a(\langle z_1, z_2 \rangle) = \langle z_1 a, z_2 a \rangle$ and $[\![\langle \$b, \#b \rangle]\!] \to f_b([\![\langle \$, \# \rangle]\!])$ with $f_b(\langle z_1, z_2 \rangle) = \langle z_1 b, z_2 b \rangle$. However some of the rules of Type II inferred from this table are wrong, e.g., the rule $[\![\langle a, a \rangle]\!] \to [\![\langle b, b \rangle]\!]$ and its symmetry. Suppose that the next positive example is $aba\#aba$. Because it is generated by the current conjecture $\hat{G}$, the algorithm does not expand K, while it finds wrong rules by expanding the table:

T_2	$\$\square a\#a\square\$$	$\square a\#a\square\$$	$\$a\square\square a\$$	$\square ba\#ab\square$
$\langle a, a\rangle$	1	0	0	0
$\langle b, b\rangle$	0	0	0	0
$\langle \$a, a\rangle$	0	1	0	0
$\langle \$b, b\rangle$	0	0	0	0
$\langle a, \#a\rangle$	0	0	1	0
$\langle b, \#b\rangle$	0	0	0	0
$\langle \$, \$\rangle$	0	0	0	0
$\langle \$a, a\$\rangle$	0	0	0	1
$\langle \$b, b\$\rangle$	0	0	0	0
$\langle \$, \#\rangle$	0	0	0	0
$\langle \$a, \#a\rangle$	0	0	0	0
$\langle \$b, \#b\rangle$	0	0	0	0

Then a half of the wrong rules are now removed. If furthermore $\mathcal{B}(2,1)$ gets $bab\#bab$ for example, the rest half disappears. Then the rules of Type II that will survive are those among $[\![\langle \$, \#\rangle]\!]$, $[\![\langle \$a, \#a\rangle]\!]$ and $[\![\langle \$b, \#b\rangle]\!]$ (and those among $[\![\langle \#, \$\rangle]\!]$, $[\![\langle a\#, a\$\rangle]\!]$ and $[\![\langle b\#, b\$\rangle]\!]$), except the trivial ones, i.e., $A \to A$ for all nonterminals A. Here the inserted marker \$, as well as the center marker #, plays a crucial role.

4 Discussion

Observation tables are a classical technique used in different schemes for computing deterministic finite state automata (e.g., [9,10]),

Rows and columns of an observation table are indexed with (potential) prefixes and suffixes of elements of the target language, respectively, and each entry shows whether the corresponding combination of the prefix and the suffix gives an element of the language. Each row corresponds to a state and if one finds two rows of the same entries, they are identified. If the index of a row is obtained by adding an input symbol on the tail of the index of another row, the corresponding transition rule is inferred.

This technique has been extended in several algorithms that learns structures more complex than deterministic finite automata such as context-free grammars [11], regular tree languages [12,13] and even a non-context-free formalism, multidimensional tree languages [14]. While those extensions take trees as input, a recent study shows that a rich subclass of context-free languages can be learnt from string data by using an observation table [15]. They use substructures as indices of rows and their context as indices of columns like our algorithm.

An important difference of our algorithm from those preceding ones is that our algorithm finds an asymmetric relation between two rows that have comparable entries, while the classical techniques merge two rows with identical entries. Two rows with identical entries will be identified as a result of finding the inclusion relations of the both directions. This generalization enables us to learn complex languages like $\{\, a^m b^n c^m d^n \mid 1 \le m \le n \,\}$.

Table 1. Comparison of observation tables for different language classes

target	regular languages [9,10]	congruential CFLs [15]	MCFLs with FKP
rows	states	nonterminals	nonterminals
row indices	prefixes	substrings	multiwords
column indices	suffixes	contexts	multicontexts
key relation between rows	equivalence	equivalence	inclusion

Our algorithm computes a grammar from any observation tables, while the standard technique for learning deterministic finite state automata requires the observation table to be *closed* and *consistent*. We may think our observation tables are always closed, because rules of Type I are just decompositions and the row indices are sub-multiword-closed. We may think of two kinds of consistency, *g-consistency* and *f-consistency*. Observation tables are said to be g-consistent if the grammar computed from the tables is compatible with all the entries of the tables. Observation tables are said to be f-consistent if one can assume that all the rules of Type II are not wrong without contradicting Lemma 1. More formally, we say that observation tables are f-consistent, if for any $\boldsymbol{u}, \boldsymbol{u}_1, \dots, \boldsymbol{u}_n, \boldsymbol{v}, \boldsymbol{v}_1, \dots, \boldsymbol{v}_n \in K$ such that $\boldsymbol{u} = f(\boldsymbol{u}_1, \dots, \boldsymbol{u}_n)$ and $\boldsymbol{v} = f(\boldsymbol{v}_1, \dots, \boldsymbol{v}_n)$ for some linear regular function f and $L/\boldsymbol{u}_i \cap X \subseteq L/\boldsymbol{v}_i \cap X$, we have $L/\boldsymbol{u} \cap X \subseteq L/\boldsymbol{v} \cap X$. For example, the first observation table T_2 of Example 5 is not f-consistent. The two rows $\langle a, a\rangle$ and $\langle b, b\rangle$ have the same entries, but the incomparability of the rows $\langle \$a, a\$\rangle$ and $\langle \$b, b\$\rangle$ entails that $L/\langle a, a\rangle$ and $L/\langle b, b\rangle$ are also incomparable by Lemma 1. One can modify our algorithm so that it extends the observation tables until they become f-consistent and g-consistent. The new algorithm is more restrained from asking equivalence queries. However currently we have no detailed mathematical analysis on this modification, which is future work.

Acknowledgement

The author is grateful to Alexander Clark, Anna Kasprzik and Takeshi Shibata for valuable comments and suggestions on a draft of this paper. This work was supported in part by Grant-in-Aid for Young Scientists (B-20700124) from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

1. Clark, A., Eyraud, R.: Polynomial identification in the limit of substitutable context-free languages. Journal of Machine Learning Research 8, 1725–1745 (2007)
2. Clark, A., Eyraud, R., Habrard, A.: A polynomial algorithm for the inference of context free languages. In: [16], pp. 29–42

3. Yoshinaka, R.: Learning mildly context-sensitive languages with multidimensional substitutability from positive data. In: Gavaldà, R., Lugosi, G., Zeugmann, T., Zilles, S. (eds.) ALT 2009. LNCS, vol. 5809, pp. 278–292. Springer, Heidelberg (2009)
4. Clark, A., Eyraud, R., Habrard, A.: A note on contextual binary feature grammars. In: EACL 2009 workshop on Computational Linguistic Aspects of Grammatical Inference, pp. 33–40 (2009)
5. Seki, H., Matsumura, T., Fujii, M., Kasami, T.: On multiple context-free grammars. Theoretical Computer Science 88(2), 191–229 (1991)
6. Kracht, M.: The Mathematics of Language. Studies in Generative Grammar, vol. 63, pp. 408–409. Walter de Gruyter, Berlin (2003)
7. Rambow, O., Satta, G.: Independent parallelism in finite copying parallel rewriting systems. Theor. Comput. Sci. 223(1-2), 87–120 (1999)
8. Kaji, Y., Nakanishi, R., Seki, H., Kasami, T.: The universal recognition problems for parallel multiple context-free grammars and for their subclasses. IEICE Transaction on Information and Systems E75-D(7), 499–508 (1992)
9. Gold, E.M.: System identification via state characterization. Automatica 8(5), 621–636 (1972)
10. Angluin, D.: Learning regular sets from queries and counterexamples. Information and Computation 75(2), 87–106 (1987)
11. Sakakibara, Y.: Learning context-free grammars from structural data in polynomial time. Theoretical Computer Science 76(2-3), 223–242 (1990)
12. Drewes, F., Högberg, J.: Learning a regular tree language from a teacher. In: Ésik, Z., Fülöp, Z. (eds.) DLT 2003. LNCS, vol. 2710, pp. 279–291. Springer, Heidelberg (2003)
13. Besombes, J., Marion, J.Y.: Learning tree languages from positive examples and membership queries. Theoretical Computer Science 382(3), 183–197 (2007)
14. Kasprzik, A.: A learning algorithm for multi-dimensional trees, or: Learning beyond context-freeness. In: [16], pp. 111–124
15. Clark, A.: Distributional learning of some context-free languages with a minimally adequate teacher. In: proceedings of the 10th International Colloquium on Grammatical Inference (2010) (to appear)
16. Clark, A., Coste, F., Miclet, L. (eds.): ICGI 2008. LNCS (LNAI), vol. 5278. Springer, Heidelberg (2008)