ON THE LEARNABILITY OF CLASSES OF MINIMAL GRAMMARS WITH RESPECT TO SOME LINEAR PREORDERING RELATIONS

Abstract. In this paper we prove the learnability of the classes of minimal grammars with respect to some linear preorderings on the set of categorial grammars. We give some examples of the linear preorder relations on the set of categorial grammars satisfying necessary conditions.

Keywords: categorial grammar, unification, learning function, learning algorithm.

1. Introduction

Buszkowski and Penn presented in (Buszkowski & Penn, 1990) a unification discovery procedure (learning algorithm) for finding categorial grammars compatible with a given sample of expressions of a language. Kanazawa in (Kanazawa, 1998) studied Gold-style learning functions based on these algorithms.

In this paper we present a learning function for classes of minimal grammars with respect to some linear preorderings on the set of categorial grammars. These results are related to Kanazawa’s open problem of whether the class of minimal grammars is learnable.

In sections 2, 3 and 4 we present basic definitions and fundamental theorems of learnability theory for categorial grammars. In section 5 we define some linear preordering relations on a set of categorial grammars. We prove the learnability of classes of minimal grammars with respect to these preordering relations.
2. Learnability Theory

We present basic notions of learnability theory, after (Gold, 1967), (Kanazawa, 1996), (Kanazawa, 1998), (Osheron, de Jongh, Martin & Weinstein, 1997).

A triple \((\Omega, S, L)\) such that \(\Omega\) is a class of grammars, \(S\) is a class of sentences (a certain recursive subset of \(\Sigma^*\) for some fixed finite alphabet \(\Sigma\)) and \(L\) is a function that maps elements of \(\Omega\) to subsets of \(S\) is called a grammar system. We assume that the universal membership problem \(s \in L(G)\), for \(s \in S, G \in \Omega\), is decidable.

Let \((\Omega, S, L)\) be a grammar system. A learning function is a partial function \(\varphi\) that maps non-empty finite sequences of sentences to grammars \((\varphi : \bigcup_{k \geq 1} S^k \rightarrow \Omega)\). Usually, we are interested in learning functions that are effectively computable. The term learning algorithm means an algorithm that computes a learning function.

Let \(\langle s_i \rangle_{i \in \mathbb{N}} = \langle s_0, s_1, \ldots \rangle\) be an infinite sequence of sentences from \(S\). Given \(\langle s_i \rangle_{i \in \mathbb{N}}\), a learning function \(\varphi\) determines a grammar \(G_i = \varphi(\langle s_0, \ldots, s_i \rangle)\) for each \(i \in \mathbb{N}\) such that \(\varphi\) is defined on \(\langle s_0, \ldots, s_i \rangle\). We say that \(\varphi\) converges to \(G\) on \(\langle s_i \rangle_{i \in \mathbb{N}}\) if \(G_i = \varphi(\langle s_0, \ldots, s_i \rangle)\) is defined and equal to \(G\) for all but finitely many \(i \in \mathbb{N}\); equivalently, if there exists an \(n \in \mathbb{N}\) such that for all \(i \geq n\), \(G_i\) is defined and equal to \(G\).

Let \(G\) be a class of grammars in \(\Omega\). \(G\) determines the corresponding class of languages, \(L(G) = \{L(G)|G \in G\}\).

Let a grammar system \((\Omega, S, L)\) be given, and let \(G \subseteq \Omega\). A learning function \(\varphi\) is said to learn \(G\) if the following condition holds: for every language \(L\) in \(L(G)\), for every infinite sequence \(\langle s_i \rangle_{i \in \mathbb{N}}\) that enumerates the elements of \(L\) \((\{s_i| i \in N\} = L)\), there exists some \(G\) in \(G\) such that \(L(G) = L\) and \(\varphi\) converges to \(G\) on \(\langle s_i \rangle_{i \in \mathbb{N}}\).

A class \(G\) of grammars is said to be learnable if there is a computable learning function that learns \(G\).

A learning function \(\varphi\) is said to be conservative if, for any finite sequence \(\langle s_0, \ldots, s_i \rangle\) of sentences and for any sentence \(s_{i+1}\), whenever \(\varphi(\langle s_0, \ldots, s_i \rangle)\) is defined and \(s_{i+1} \in L(\varphi(\langle s_0, \ldots, s_i \rangle)), \varphi(\langle s_0, \ldots, s_i, s_{i+1} \rangle)\) is also defined and \(\varphi(\langle s_0, \ldots, s_i, s_{i+1} \rangle) = \varphi(\langle s_0, \ldots, s_i \rangle)\).

A class \(L\) of languages is said to have infinite elasticity if there exists an infinite sequence \(\langle s_n \rangle_{n \in \mathbb{N}}\) of sentences and an infinite sequence \(\langle L_n \rangle_{n \in \mathbb{N}}\) of languages in \(L\) such that, for all \(n \in \mathbb{N}, s_n \notin L_n\) and \(\{s_0, \ldots, s_n\} \subseteq L_{n+1}\).

A class \(L\) of languages is said to have finite elasticity if it does not have infinite elasticity.
Lemma 2.1. (Kapur & Shyam, 1991)

Let \( L \) be a language class with finite elasticity. Then for each language \( L \in \mathcal{L} \), there is a finite set \( D_L \subseteq L \) such that \( L \) is the smallest element of \( \{ L' \in \mathcal{L} | D_L \subseteq L' \} \).

3. Unification

Let \( L \) be a first-order language. By \( \text{TER}(L) \) we denote the set of all terms of \( L \). \( \text{VAR} \) denotes the set of individual variables, and \( \text{CON}(L) \) denotes the set of all individual constants of \( L \). A substitution \( \alpha \) is a mapping \( \alpha : \text{VAR} \rightarrow \text{TER}(L) \). Each substitution \( \alpha \) is uniquely extended to a homomorphism \( \alpha : \text{TER}(L) \rightarrow \text{TER}(L) \) by setting: (i) \( \alpha(a) = a \), for \( a \in \text{CON}(L) \), (ii) \( \alpha(f(t_1, \ldots, t_n)) = f(\alpha(t_1), \ldots, \alpha(t_n)) \). \( \alpha \circ \beta \) denotes the composition of \( \alpha \) and \( \beta \); \( (\alpha \circ \beta)(t) = \alpha(\beta(t)) \), for any term \( t \). Terms \( s, t \) are called variants, if there are substitutions \( \alpha, \beta \) such that \( \alpha(s) = t \) and \( \beta(t) = s \) (then, \( s \) is equal to \( t \) up to renaming variables).

The set of all substitutions \( \alpha : \text{VAR} \rightarrow \text{TER}(L) \) we denote by \( \text{SUB} \). A set \( X \subseteq \text{SUB} \) is said to be hereditary, if, for all \( \alpha, \beta \in \text{SUB} \) if \( \alpha \circ \beta \in X \), then \( \beta \in X \).

A set \( T \subseteq \text{TER}(L) \) is said to be unifiable, if there is a substitution \( \alpha \) such that \( \alpha(s) = \alpha(t) \), for any pair \( (s, t) \in T \); \( \alpha \) is called a unifier of \( T \). A substitution \( \alpha \) is called a most general unifier (m.g.u.) of \( T \), if \( \alpha \) is a unifier of \( T \) and, for every unifier \( \beta \) of \( T \), there is a substitution \( \gamma \) such that \( \beta = \gamma \circ \alpha \).

The well-known unification algorithm (Doets, 1994), (Lloyd, 1987) takes an arbitrary finite set of terms as an input and returns an m.g.u. of \( T \), if \( T \) is unifiable; otherwise, the algorithm replies negatively.

For our purposes, it is more expedient to consider unifiers of relations. Let \( R \) be a finite binary relation on \( \text{TER}(L) \). A substitution \( \alpha \) is called a unifier of \( R \), if \( \alpha(s) = \alpha(t) \), for any pair \( (s, t) \in R \).

The relation \( R \) is said to be unifiable, if there exists a unifier of \( R \). The notion of an m.g.u. of \( R \) is defined as above. It is easy to adopt the unification algorithm to binary relations as inputs. Let \( R = \{(s_1, t_1), \ldots, (s_n, t_n)\} \). Take a new n-ary function symbol \( F \) and consider the set \( T \) consisting of two terms: \( F(s_1, \ldots, s_n) \) and \( F(t_1, \ldots, t_n) \). Then, unifiers of \( R \) are precisely unifiers of \( T \) (not using \( F \)), and consequently, the unification algorithm applied to \( T \) provides an m.g.u. of \( R \), if \( R \) is unifiable, and replies negatively, otherwise.

Let \( \mathcal{S} \) be a finite family of finite sets of terms. A substitution \( \alpha \) is called a unifier of \( \mathcal{S} \), if \( \alpha \) is a unifier of every set \( T \in \mathcal{S} \) (but not necessarily of the
join of $\mathcal{S}$). For a family $\mathcal{S}$, we define a relation $R(\mathcal{S})$ as follows: $(s, t) \in R(\mathcal{S})$ iff, for some $T \in \mathcal{S}$, both $s \in T$ and $t \in T$. Then, unifiers of $\mathcal{S}$ are precisely unifiers of $R(\mathcal{S})$, and similarly for other notions, defined in terms of unifiers.

For a substitution $\alpha$, the relation $\text{Ker}(\alpha)$ (the kernel of $\alpha$) is defined as follows: for $s, t \in \text{TER}(L)$, $(s, t) \in \text{Ker}(\alpha)$ iff $\alpha(s) = \alpha(t)$. For a relation $R$ between terms, we set $\text{Ker}_R(\alpha) = R \cap \text{Ker}(\alpha)$. Clearly, $\alpha$ is a unifier of $\text{Ker}_R(\alpha)$ and $\text{Ker}_R(\alpha) \subseteq \text{Ker}_R(\beta \circ \alpha)$, for all $R$, $\alpha$ and $\beta$.

### 4. Classical Categorial Grammars

The notion of a categorial grammar has been introduced by Ajdukiewicz in (Ajdukiewicz, 1935). In a categorial grammar, each symbol in the alphabet is associated with a finite number of types. $\text{Pr}$ denotes the set of primitive types. One member $t$ of $\text{Pr}$ is singled out as the distinguished type. The members of $\text{Pr}$ other than $t$ are called variables. The set of variables is denoted $\text{VAR}$. Types are constructed from primitive types by two type-forming operators, $/$ and $\backslash$. $Tp$ (the set of all types) is defined as the smallest set satisfying the following conditions:

1. $\text{Pr} \subseteq Tp$,
2. if $A \in Tp$ and $B \in Tp$, then $A\backslash B \in Tp$,
3. if $A \in Tp$ and $B \in Tp$, then $A/B \in Tp$.

One assumes the following reduction rules:

- $A, A\backslash B \to B$
- $A/B, B \to A$.

A classical categorial grammar over $\Sigma$ is a finite relation $G$ between $\Sigma$ and $Tp$. For $c \in \Sigma$ and $A \in Tp$, if $(c, A) \in G$, then we say that $G$ assigns $A$ to $c$, and write $G : c \to A$. We say that $G$ assigns type $A$ to string $c_1 \ldots c_n$ if there exist $(c_i, A_i) \in G, i = 1, \ldots, n$, such that $A_1, \ldots, A_n$ reduces to $A$ by a successive application of the above rules.

Types can be identified with terms of a first-order language with function symbols $\backslash$, $/$ and the constant $t$.

Let $\sigma$ be a substitution. $\sigma(G)$ denotes a grammar obtained by applying $\sigma$ to the type assignments of $G$, that is: $\sigma(G) = \{(c, \sigma(A)) : (c, A) \in G\}$.

We define the set $\Sigma^F$ (of functor-argument structures over alphabet $\Sigma$) as follows:

1. $c \in \Sigma^F$ for all $c \in \Sigma$,
2. if $X_1, X_2 \in \Sigma^F$, then also $(X_1, X_2)_i \in \Sigma^F$, $i = 1, 2$.

$X_i$ is called the functor and $X_j, j \neq i$ is called an argument.
Let $v \in \Sigma$. By $I_G(v)$ we denote the set $\{ A \in Tp : G : v \rightarrow A \}$. For $T_1, T_2 \subseteq Tp$, we set: $f_1(T_1, T_2) = \{ B \in Tp : \exists A (B/A \in T_1 \land A \in T_2) \}$, $f_2(T_1, T_2) = \{ B \in Tp : \exists A (A \in T_1 \land A \backslash B \in T_2) \}$. Let $T_G$ be a mapping from $\Sigma^F$ into the subsets of $Tp$ defined as follows: $T_G(v) = I_G(v)$, for $v \in \Sigma$, $T_G((X,Y)_i) = f_i(T_G(X), T_G(Y))$. We define the structure language of $G$ by $FL(G) = \{ X \in \Sigma^F : t \in T_G(X) \}$.

Let $D$ be a finite set of functor-argument structures. A grammar $GF(D)$ (the general form determined by $D$) is constructed in the following way:

(i) we assign the distinguished type $t$ to all functor-argument structures $X \in D$,

(ii) we assign distinct variables to all argument-substructures of all $X \in D$,

(iii) we assign types to all functor-substructures according to the following rules:
- if $Y \rightarrow A$, $(X,Y)_1 \rightarrow B$, then $X \rightarrow B/A$,
- if $X \rightarrow A$, $(X,Y)_2 \rightarrow B$, then $Y \rightarrow A \backslash B$.

(iv) we collect the types assigned to all symbols from the alphabet into the grammar $GF(D)$.

We define: $dom(G) = \{ c \in \Sigma : \exists A \in Tp (G : c \rightarrow A) \}$.

Let $\sqsubseteq$ be a binary relation on grammars such that $G_1 \sqsubseteq G_2$ if and only if there exists a substitution $\sigma$ with the following properties:

(i) for all $c \in dom(G)$, if $G_1 : c \rightarrow A$, $G_1 : c \rightarrow B$ and $A \neq B$, then $\sigma(A) \neq \sigma(B)$,

(ii) $\sigma(G_1) \subseteq G_2$.

A grammar $G$ is said to be in reduced form if there is no $G'$ such that $G' \sqsubseteq G$ and $FL(G) = FL(G')$. $G' \sqsubseteq G$ means: $G' \sqsubseteq G$ and $G \not\sqsubseteq G'$.

**Proposition 4.1** (Kanazawa, 1998)

For every grammar $G$, one can effectively find a grammar $G'$ in reduced form such that $G' \sqsubseteq G$ and $FL(G) = FL(G')$.

By $range(G)$ we denote the set $\{ A : (c, A) \in G \text{ for some } c \in \Sigma \}$.

Let $Tp(G) = \{ A : A \text{ is a subtype of some } B \in range(G) \}$. A type $A \in Tp(G)$ is useless if and only if there is no string $c_1 \ldots c_n$ such that there exist $(c_i, A_i) \in G, i = 1, \ldots, n$, such that $A$ is a subtype of some $A_i$ and $A_1, \ldots, A_n$ reduces to $t$.

**Proposition 4.2** (Kanazawa, 1998)

If a grammar $G$ is in reduced form, then $G$ has no useless type.
5. Classes of minimal grammars with respect to some linear preordering relations

We prove the learnability of the classes of minimal grammars with respect to some linear preorderings on the set of categorial grammars.

Recall that a preorder relation on a set $X$ is a reflexive and transitive relation on $X$. If $\leq$ is a preorder relation on $X$, then we write $a \sim b$ iff $a \leq b$ and $b \leq a$; and $a < b$ if $a \leq b$ and $b \not\leq a$, for $a, b \in X$.

Let $X \subseteq \text{SUB}$ be hereditary. Let $R$ be a finite relation between types. A preorder relation $\leq$ on $X$ is said to be sound for $R$, if it satisfies the following conditions:

(C1) if $\text{Ker}_R(\alpha) \subset \text{Ker}_R(\beta)$, then $\beta < \alpha$,
(C2) if $\text{Ker}_R(\alpha) = \text{Ker}_R(\beta)$, then $\alpha \sim \beta$,

for all $\alpha, \beta \in X$.

A linear preorder on a set $X$ is a preorder relation on $X$ satisfying the following condition: $a \leq b$ or $b \leq a$, for all $a, b \in X$.

Let $D$ be a set of functor-argument structures. Then $\mathcal{A}(D) = \{ \{ A : GF(D) : c \rightarrow A \} : c \in \text{dom}(GF(D)) \}$. We set $R_D = R_{\mathcal{A}(D)}$.

Let $\preceq$ be a linear preorder on a set of categorial grammars satisfying the following conditions:

(G1) $G_1 \sqsubseteq G_2 \Rightarrow G_1 \preceq G_2$,
(G2) the preorder $\preceq$ on a set of substitutions defined by: $\alpha \preceq \beta$ iff $\alpha(GF(D)) \preceq \beta(GF(D))$ is sound for $R_D$, for all finite sets of functor-argument structures $D$,
(G3) the class of structure languages $\{ FL(G') : G' \preceq G \}$ has finite elasticity for any categorial grammar $G$.

We define: $G_1 \simeq G_2$ iff $G_1 \preceq G_2$ and $G_2 \preceq G_1$; $G_1 \prec G_2$ iff $G_1 \preceq G_2$ and $G_2 \not\preceq G_1$.

Let $L \subseteq \Sigma^F$. A grammar $G$ is said to be minimal with respect to $(L, \preceq)$ if $L \subseteq FL(G)$ and there is no grammar $G'$ such that $G' \prec G$ and $L \subseteq FL(G')$.

A grammar $G$ is said to be minimal with respect to $\preceq$ iff $G$ is minimal with respect to $(FL(G), \preceq)$. The class of minimal grammars with respect to $\preceq$ is denoted by $\mathcal{G}_{\text{min}, \preceq}$. By $\mathcal{F}L_{\text{min}, \preceq}$ we denote the set $\{ FL(G) : G \in \mathcal{G}_{\text{min}, \preceq} \}$.

A substitution $\alpha$ is called an optimal unifier of a relation $R$, if it satisfies the following conditions (Buszkowski & Penn, 1990), (Buszkowski & Dziemidowicz, 2000):
On the Learnability of Classes of Minimal Grammars...

(OU.1) for every substitution $\beta$, $\text{Ker}_R(\alpha) = \text{Ker}_R(\beta \circ \alpha)$,

(OU.2) $\alpha$ is an m.g.u. of $\text{Ker}_R(\alpha)$.

Let $R$ be a finite relation between types. We define:

$$OU(R) = \{ \sigma : \sigma \text{ is optimal unifier of } R \}.$$  

We provide an Algorithm OG based on Buszkowski’s algorithm (Buszkowski & Penn, 1990):

Algorithm OG

Input: a finite set $D$ of functor-argument structures
Output: a finite set of grammars $G$ such that $D \subseteq FL(G)$.

1. Construct $GF(D)$.
2. Compute $OU(R_D)$.
3. Let $OG(D) = \{ \sigma(GF(D)) : \sigma \in OU(R_D) \}$.

A grammar $G$ is said to be optimal if the following two conditions hold:

(i) $G$ has no useless types,
(ii) if $G : c \to A$ and $G : c \to B$ and $A \neq B$, then $R = \{(A,B)\}$ is not unifiable.

The class of optimal grammars is denoted by $G_{optimal}$.

Lemma 5.1. (Kanazawa, 1998)

$$\{ G \in G_{optimal} : G \text{ is in reduced form } \} \subset \bigcup \text{range}(OG).$$

Lemma 5.2. (Kanazawa, 1998)

If $G \in OG(D)$ and $T \in FL(G)$, then $G \in OG(D \cup \{ T \})$.

A categorial grammar assigning at most $k$ types to each symbol in the alphabet is called a $k$-valued grammar. The set of $k$-valued grammars is denoted by $G_k$.

Let $A = \{ T_1, \ldots, T_n \}$ be a family of sets of types. If $P_i = \{ P_{i,1}, \ldots, P_{i,l_i} \}$ is a partition of $T_i$ for $1 \leq i \leq n$, then the family $P = \bigcup \{ P_i : 1 \leq i \leq n \}$ is called a partition of $A$. $P$ is called a $k$-partition of $A$ if $l_i \leq k$ for each $i$.

Let $PU_k(A(D)) = \{ \text{m.g.u.}(P) : P \text{ is a } k \text{-partition of } A(D) \}$.

We provide an Algorithm $VG_k$:

Algorithm $VG_k$

Input: a finite set $D$ of functor-argument structures
Output: a finite set $G$ of $k$-valued grammars such that for each $G \in G$, $D \subseteq FL(G)$
1. Construct \(GF(D)\).
2. Compute \(PU_k(A(D))\).
3. Let \(VG_k(D) = \{\sigma(GF(D)) : \sigma \in PU_k(A(D))\}\).

**Lemma 5.3.** (Kanazawa, 1998)

For \(k \geq 2\), \(\{G \in G_k : G \text{ is in reduced form} \} \subset \bigcup range(VG_k)\).

**Lemma 5.4.**

Let \(G\) be a grammar in reduced form. Then there is a substitution \(\sigma\) such that \(G = \sigma(GF(D))\) for some set of functor-argument structures \(D\).

**Proof.** Let \(G\) be a grammar in reduced form. Find \(k\), such that \(G \in G_k\). By lemma 5.3, we have \(G \subset \bigcup range(VG_k)\), so there are a substitution \(\sigma\) and a set of functor-argument structures \(D\) such that \(G = \sigma(GF(D))\). \(\square\)

Let \(D\) be a finite set of functor-argument structures. We define:

\[
MG \preceq(D) = \{G \in OG(D) : \neg \exists G' \in OG(D) (G' \prec G)\}.
\]

**Lemma 5.5.**

The following are equivalent:

(i) \(G \in G_{\min, \preceq}\),

(ii) there is a finite subset \(D \subseteq FL(G)\) such that \(G\) is minimal with respect to \((D, \preceq)\).

**Proof.** (ii) \(\Rightarrow\) (i) is obvious. To prove (i) \(\Rightarrow\) (ii) assume \(G \in G_{\min, \preceq}\). Let \(\langle T_i \rangle_{i \in N} \) be an infinite sequence enumerating \(FL(G)\) \(\{\{T_i : i \in N\} = FL(G)\)\). For any \(G' \prec G\), \(FL(G) \not\subseteq FL(G')\). Let \(i_{G'}\) be the least \(i\) such that \(T_i \not\in FL(G')\). Then \(\{T_0, \ldots, T_{i_{G'}-1}\} \subseteq FL(G')\). Since \(\{FL(H) : H \prec G\}\) has finite elasticity by (G3), this implies that there is \(m \in N\) such that \(i_{G'} \leq m\) for all \(G' \in \{H : H \prec G\}\). Then \(G\) must be minimal with respect to \((D, \preceq)\), where \(D = \{T_0, \ldots, T_m\}\). \(\square\)

**Lemma 5.6.**

If \(G \in OG(D) \cap G_{\min, \preceq}\), then there exists a finite set \(D'\) such that \(G \in MG \preceq(D')\).

**Proof.** Assume \(G \in OG(D)\) and \(G \in G_{\min, \preceq}\). By lemma 5.5 we can find a finite subset \(D_0 \subseteq FL(G)\) such that \(G\) is minimal with respect to \((D_0, \preceq)\). Then \(D \cup D_0 \subseteq FL(G)\) and \(G\) is minimal with respect to \((D \cup D_0, \preceq)\). By lemma 5.2, \(G \in OG(D \cup D_0)\). Then \(G \in MG \preceq(D \cup D_0)\). \(\square\)
Lemma 5.7.

For each $G \in \mathcal{G}_{\text{min,} \preceq}$, there exists $G_0 \in \mathcal{G}_{\text{min,} \preceq}$ such that $G_0$ is in reduced form, $FL(G_0) = FL(G)$ and $G_0 \in MG_{\preceq}(D)$ for some finite set $D \subseteq FL(G)$.

Proof. Assume $G \in \mathcal{G}_{\text{min,} \preceq}$. There exists $G_0$ such that $G_0$ is in reduced form and $FL(G_0) = FL(G)$. We show that $G_0 \in \mathcal{G}_{\text{min,} \preceq}$. Assume not. Then, there is a grammar $G'$ such that $G' \prec G_0$ and $FL(G) \subseteq FL(G')$. We have $G_0 \preceq G$, by $G_0 \subseteq G$ and the condition (G1). We obtain $G' \prec G$, which contradicts $G \in \mathcal{G}_{\text{min,} \preceq}$. We have to show that there exists some finite set $D \subseteq FLG$ such that $G_0 \in MG_{\preceq}(D)$. By lemma 5.6 and lemma 5.1, it is enough to show that $G_0 \in \mathcal{G}_{\text{optimal}}$. Suppose not. Since $G_0$ is in reduced form, it has no useless types, so by the definition of an optimal grammar there are types $A$ and $A'$ assigned to the same symbol by $G_0$ such that $A \neq A'$ and the relation $\{(A, A')\}$ has a unifier $\alpha$. By lemma 5.4, there is a substitution $\sigma$ such that $G_0 = \sigma(GF(D))$ for some set of functor-argument structures $D$. Then, by (G2) and (C1) we have $\alpha(\sigma(GF(D))) < \sigma(GF(D))$, so $\alpha(G_0) < G_0$. $FL(G_0) \subseteq FL(\alpha(G_0))$. It contradicts $G_0 \in \mathcal{G}_{\text{min,} \preceq}$. □

Let $\mu_{FL}$ be a computable function that maps a non-empty finite set $\mathcal{G}$ of grammars to a grammar $G \in \mathcal{G}$ such that $FL(G)$ is a minimal element of $\{FL(G) : G \in \mathcal{G}\}$.

We define the learning function $\varphi_{MG_{\preceq}}$ for $\langle \mathcal{G}_{\text{min,} \preceq}, \Sigma^F, FL \rangle$ in the following way:

(i) $\varphi_{MG_{\preceq}}(\langle T_0 \rangle) = \mu_{FL}(MG_{\preceq}(\{T_0\}))$,
(ii) $\varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_{i+1} \rangle) = \varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_i \rangle)$ if $\varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_i \rangle)$ is defined and $T_{i+1} \in FL(\varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_i \rangle))$,
(iii) $\varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_{i+1} \rangle) = \mu_{FL}(MG_{\preceq}(\{T_0, \ldots, T_{i+1}\}))$ otherwise.

Lemma 5.8.

$FL(\varphi_{MG_{\preceq}}(\langle T_0, \ldots, T_i \rangle))$ is always a minimal element of $\{FL(G) : G \in MG_{\preceq}(\{T_0, \ldots, T_i\})\}$

Proof. It is easy to show by definition of the function $\varphi_{MG_{\preceq}}$ and definition of the function $\mu_{FL}$. □

Theorem 5.9

$\varphi_{MG_{\preceq}}$ learns $\mathcal{G}_{\text{min,} \preceq}$.

Proof. Let $L \in FL_{\text{min,} \preceq}$ and let $\langle T_i \rangle_{i \in \mathbb{N}}$ be an infinite sequence enumerating $L$. Let $G_0 \in \mathcal{G}_{\text{min,} \preceq}$ be a grammar in reduced form such that $FL(G_0) = L$. 

45
We obtain the grammar $GF(D)$:

\[
\begin{align*}
John & \rightarrow x_1, x_5, x_6/x_7 \\
likes & \rightarrow x_7, (x_1 \setminus t)/x_2, (x_3 \setminus t)/x_5 \\
only & \rightarrow x_3/x_4, (x_6 \setminus t)/x_8 \\
Helen & \rightarrow x_2, x_4, x_8
\end{align*}
\]
It is not unifiable. The following substitutions:

1: \((x_1 : x_6/x_7, x_5 : x_6/x_7, x_3 : x_6/x_7, x_2 : x_6/x_7, x_4 : x_6/x_7, x_8 : x_6/x_7)\)

2: \((x_2 : x_6 \setminus t, x_4 : x_6 \setminus t, x_8 : x_6 \setminus t, x_3 : x_6 \setminus t, x_5 : x_6 \setminus t, x_1 : x_6 \setminus t, x_7 : ((x_6 \setminus t)/(x_6 \setminus t)))\)

3: \((x_4 : x_2, x_8 : x_2, x_3 : x_6 \setminus t, x_1 : x_6/x_7, x_5 : x_6/x_7)\)

4: \((x_1 : x_6/x_7, x_5 : x_6/x_7, x_3 : x_6 \setminus t, x_4 : x_8, x_2 : x_8)\) are optimal unifiers of \(R_D\).

We obtain the following grammars:

1: \(\text{John} \rightarrow x_6/x_7\)

- \(\text{likes} \rightarrow x_7, ((x_6/x_7) \setminus t)/(x_6/x_7)\)
- \(\text{only} \rightarrow (x_6/x_7)/(x_6/x_7), (x_6 \setminus t)/(x_6/x_7)\)
- \(\text{Helen} \rightarrow x_6/x_7\)

2: \(\text{John} \rightarrow x_6 \setminus t, x_6/(((x_6 \setminus t) \setminus t)/(x_6 \setminus t))\)

- \(\text{likes} \rightarrow ((x_6 \setminus t) \setminus t)/(x_6 \setminus t)\)
- \(\text{only} \rightarrow (x_6 \setminus t)/(x_6 \setminus t)\)
- \(\text{Helen} \rightarrow x_6 \setminus t\)

3: \(\text{John} \rightarrow x_6/x_7\)

- \(\text{likes} \rightarrow x_7, ((x_6/x_7) \setminus t)/x_2, ((x_6 \setminus t) \setminus t)/(x_6/x_7)\)
- \(\text{only} \rightarrow (x_6 \setminus t)/x_2\)
- \(\text{Helen} \rightarrow x_2\)

4: \(\text{John} \rightarrow x_6/x_7\)

- \(\text{likes} \rightarrow x_7, ((x_6/x_7) \setminus t)/x_8, ((x_6 \setminus t) \setminus t)/(x_6/x_7)\)
- \(\text{only} \rightarrow (x_6 \setminus t)/x_8\)
- \(\text{Helen} \rightarrow x_8\)

The grammar (2) is minimal for \((D, \preceq_1)\) and \((D, \preceq_2)\).

**References**


