Constructive Learning of Context-Free Languages with a Subpansive Tree ⋆ ⋆⋆

Noriko Sugimoto[1], Takashi Toyoshima[2],
Shinichi Shimozono[1], and Kouichi Hirata[1]

[1] Department of Artificial Intelligence,
[2] Department of Human Sciences,
Kyushu Institute of Technology,
Kawazu 680-4, Iizuka 820-8502, Japan
{sugimoto,sin,hirata}@ai.kyutech.ac.jp
toyoshima@lai.kyutech.ac.jp

Abstract. A subpansive tree is a rooted tree that gives a partial order of nonterminal symbols of a context-free grammar. We formalize subpansive trees as background knowledge of CFGs, and investigate query learning of CFGs with the help of subpansive trees. We show a restricted class of CFGs, which we call hierarchical CFGs, is efficiently learnable, while it is unlikely to be polynomial-time predictable.

1 Introduction

Language acquisition is one of the central interests to both theoretical computer science and linguistics. In computational learning theory, Angluin [2] showed that the regular languages are efficiently learnable, proposing a polynomial-time algorithm on finite automata with *terminal* membership queries and *terminal* equivalence queries. Since then, the learnability of context-free classes has come to be the next research topic, and a number of studies have been reported under different acquisition scenarios: Angluin [1] allowed *nonterminal* membership queries for learning a k-bounded context-free grammar (CFG); Sakakibara [25, 26] and Sakamoto [27] proposed learning algorithms from *structured* examples with *structural* queries; Ishizaka [17] dropped nonterminal membership queries for a simple deterministic CFG, but with equivalence queries extended to ask about a grammar from the outside of the target class.

For natural language acquisition by humans, on the other hand, children are *not* always corrected when they produce a "wrong" utterance, or even told when they produce or hear a "wrong" utterance; let alone what is wrong about

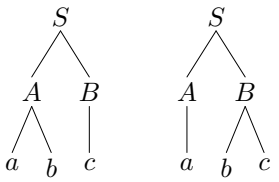
⋆ The research reported here is partially supported by Grants-in-Aid for Encouragement of Young Scientists, Japan Society for the Promotion of Science, awarded to the last three authors (nos. 12710285, 12780286, 11780284, respectively). The standard disclaimer applies.

it. That is, there is no consistently reliable teacher, and structural information is scarcely given. Thus, it is usually held that humans are endowed with some kind of "innate" grammatical knowledge that enables them to acquire a target grammar without *negative* examples [6, 23]. This "innate" grammatical knowledge of humans must be abstract and adaptive enough to develop into concrete particular *grammars* [9, 11], of more than 4,000 languages, by modest estimate, known in the world.

In this paper, we introduce the concept of a *subpansive tree* that formalizes a background knowledge to characterize CFGs. A subpansive tree is a rooted tree that represents a partial order on nonterminal symbols of a target CFG. This partial order on nonterminals models a part of the "innate" grammatical knowledge of humans. In natural languages, sentences are not directly made up from words but internally structured; words constitute phrases, and phrases constitute sentences [10, 18]. In these structures, we can observe *categorical inheritance* to a phrase from its *constituent head* (a part of speech), so that a noun phrase is projected from a noun, for example. This *endocentricity* [4] is the origin of the context-free base [7], and the opposite *exocentricity* and discontinuous constituency/dependency are the principal sources that natural languages are considered not to be properly included in the context-free class [8].

The subpansive tree encodes the categorical inheritance found in phrase structures of natural languages, and helps to derive productions of a target CFG ('expansion rules' to generate sentences) by guessing a nonterminal symbol into which a substring can be intrajected as 'subpansion.' A relation from an ancestor nonterminal A to its descendant nonterminal B implies that from an abstract nonterminal A, specific strings containing B will be obtained in the succeeding productions.

This use of a subpansive tree captures an intuition about the importance of structural information for grammatical inference that is different from the one employed in [25–27]. Rather than to use specific structural information from examples and queries, the subpansive tree is given as an abstract background knowledge about structures, from which a particular grammar — instructions to build specific structures — develops. Human children are not directly exposed to structural information of sentences, but learn to divide them into phrases and down to words; they lean to structure a string of words into phrases, and up to a sentence, i.e., terminal string into nonterminal symbols, and up to the root. Yet, a string of a context-free language (CFL) can be structurally ambiguous. For instance, a string of terminal symbols abc can have either of the following CFG structures:



Here, the set of symbols are exactly the same, but the structures, and hence the CFGs (productions) that generate them, are distinct. This point is already made in [25], but this very ambiguity is what must be resolved by humans through learning a particular grammar, not to be pre-solved as examples or hints; that is, what it means to learn a grammar is to learn how to structure strings of terminal symbols. Since the number of parts of speech is quite limited and the categorical inheritance readily determines what phrase can expand into what kind of strings of words in natural languages, we take, unlike Ishizaka's model [17], the set of nonterminal symbols is given, and as a preliminary study, we adopt Angluin's [1] *nonterminal* membership queries and *terminal* equivalence queries, to see the merits and demerits of the subpansive tree in learning CFGs, though nonterminal membership queries are known to be rather powerful.

In the following, we first define a subpansive tree of a CFG, and discuss a few fundamental issues on relations between CFGs and subpansive trees that require a mild restriction on productions. Next, we study the learnability of CFGs to which a subpansive tree requires that for every production, all nonterminals in the righthand-side are children of the lefthand-side. We call this class of CFGs the *hierarchical* CFGs. In this setting, we design a learning algorithm using terminal equivalence and nonterminal membership queries. This algorithm builds a hypothesis constructively, by generalizing the most specific productions. Furthermore, we show that for any size of DNF formulas, there exists a subpansive tree such that if CFGs with a subpansive tree are predictable in polynomial time, then so is DNF formulas. This implies that the class of hierarchical CFGs is unlikely to be polynomial-time predictable.

2 Preliminaries

An *alphabet* is a non-empty finite set of symbols. For an alphabet X, the set of all finite strings formed from symbols in X is denoted by X^*. The empty string is denoted by ε, and X^+ denotes the set $X^* - \{\varepsilon\}$ of non-empty strings. A *language over* X is a subset of X^*.

Let Σ and N be alphabets that are mutually disjoint $\Sigma \cap N = \emptyset$. A *production* $A \to \alpha$ *on* N *and* Σ is an association form a nonterminal $A \in N$ to a string $\alpha \in (N \cup \Sigma)^*$. A *context-free grammar* (*CFG*, for short) is a 4-tuple (Σ, N, P, S), where $S \in N$ is the distinguished *start symbol* and P is a finite set of productions on N and Σ. Symbols in N are said to be *nonterminals*, while symbols in Σ are called *terminals*. Let α and β be strings in $(\Sigma \cup N)^*$. We say that β *is derived from* α *in one step with* G, and denote $\alpha \Rightarrow_G \beta$, if there exists a production $X \to \chi$ in P such that, for some $\alpha_1, \alpha_2 \in (\Sigma \cup N)^*$, $\alpha = \alpha_1 X \alpha_2$ and $\beta = \alpha_1 \chi \alpha_2$. That is, β is obtained from α by replacing one occurrence of A by α. We extend the relation $\Rightarrow_G$ to the reflexive and transitive closure $\Rightarrow_G^*$.

Let $G = (\Sigma, N, P, S)$ be a CFG, and A a nonterminal in N. The *language* $L_G(A)$ *of* A is the set $\{w \in \Sigma^* \mid A \Rightarrow_G^* w\}$. The *language* $L(G)$ *of* G just refers to $L_G(S)$. A language L is called a *context-free language* (*CFL*, for short) if there exists a CFG G that identifies $L = L(G)$.

A CFG $G = (\Sigma, N, P, S)$ is said to be *reduced* if every $A \in N$ satisfies the following conditions:

1. there exists a string $w \in \Sigma^*$ such that $A \Rightarrow_G^* w$,
2. there exist $\alpha, \beta \in (\Sigma \cup N)^*$ such that $S \Rightarrow_G^* \alpha A \beta$, and
3. $L_G(A) \neq \{\varepsilon\}$.

Throughout this paper, every CFG is assumed to be reduced.

Next, we prepare the models of learning. Let $\mathcal{G}$ be a family of CFGs and $\mathcal{L}$ the family of CFLs corresponding to $\mathcal{G}$, that is, $\mathcal{L} = \{L(G) \mid G \in \mathcal{G}\}$. Then, in the context of identifying a grammar $G \in \mathcal{G}$ through members and nonmembers of $L = L(G)$, the family $\mathcal{G}$ (or $\mathcal{L}$) is said to be the *target concept*, and the grammar G (language $L(G)$) is said to be the *target grammar* (*target language*, respectively). As far as we concern in this paper, sets of nonterminals of the same size are identified. Also, learning algorithms will deal with only a family $\mathcal{G}_{\Sigma,N}$ of CFGs specified by some fixed Σ and N.

Let $G = (\Sigma, N, P, S)$ be a target CFG in a family $\mathcal{G}_{\Sigma,N}$, and L the corresponding language $L = L(G)$. A *positive example of* G (or, equivalently, L) is a string $w \in L$, and a *negative example of* G (or L) is a string $w \notin L$. To obtain these examples or to ensure the target grammar, learning algorithms are allowed to issue queries of the following types to the teacher: For a pair $(w, A) \in \Sigma^* \times N$, a *nonterminal membership query* $NM_G(w, A)$ asks whether w is in $L_G(A)$. For a grammar $G' \in \mathcal{G}$, an *equivalence query* $EQ_G(G')$ asks whether $L(G') = L(G)$: If the empty string is replied, then the answer is 'yes;' Otherwise, if a non-empty string w is replied, then the answer is 'no,' and w is either a *positive counterexample* in $L(G) - L(G')$ or a *negative counterexample* in $L(G') - L(G)$.

A family $\mathcal{G}$ of CFGs is said to be *polynomial-time learnable via equivalence and nonterminal membership queries* if there exists a learning algorithm that use both equivalence and nonterminal membership queries and exactly identifies each $G \in \mathcal{G}$ in polynomial-time with the maximum length of counterexamples. In the same manner, a family of CFGs is *polynomial-time learnable via equivalence queries alone* if it can be identified only with equivalence queries in polynomial-time.

3 A Context-Free Grammar and a Subpansive Tree

In this section, we introduce the concept of *subpansive tree*. In the learning process, the subpansive tree of a CFG works as background knowledge.

Let L and L' be languages over Σ. Then, L' is said to be a *component of* L if there exist two strings $u, v \in \Sigma^*$ such that $w \in L'$ implies $uwv \in L$.

Definition 1. Let $G = (\Sigma, N, P, S)$ be a CFG, and let $T = (N, E)$ be a rooted tree. Then, T is a *subpansive tree of* G if it satisfies the following conditions:

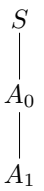
1. S is a root of T, and
2. if $A \in N$ is an ancestor of $B \in N$ on T, then $L_G(B)$ is a component of $L_G(A)$.

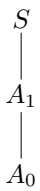
Example 1. Consider the following CFG G:

$$G = (\{S, A_0, A_1\}, \{0, 1, a\}, \{S \to 0A_0 \mid 1A_1, A_0 \to aa \mid aA_1, A_1 \to a \mid aA_0\}, S).$$

Let T_i $(i = 1, 2, 3)$ be the trees as follows:

$T_1 =$ 

$T_2 =$ 

$T_3 =$ 

For each nonterminal S, A_0 and A_1, it holds that

$$L_G(S) = \{0a^{2n} \mid n \geq 1\} \cup \{1a^{2n-1} \mid n \geq 1\},$$
$$L_G(A_0) = \{a^{2n} \mid n \geq 1\}, \quad L_G(A_1) = \{a^{2n-1} \mid n \geq 1\}.$$

Then, the following is clear:

$$\{0w \mid w \in L_G(A_0)\} \subseteq L_G(S), \ \{1w \mid w \in L_G(A_1)\} \subseteq L_G(S),$$
$$\{aw \mid w \in L_G(A_0)\} \subseteq L_G(A_1), \ \{aw \mid w \in L_G(A_1)\} \subseteq L_G(A_0).$$

Hence, every T_i is a subpansive tree of G.

As the relationship between a CFG and a tree, the following proposition holds:

Proposition 1. Let G be a CFG and T a tree. Then, it is undecidable to determine whether T is a subpansive tree of G.

Proof. Deciding whether $L(G') \subseteq L(G'')$ for given two CFGs G' and G'' is undecidable (*cf.* [16]). It can be trivially reduced to asking whether a language $L_G(B)$ is a component of $L_G(A)$ for some nonterminals A and B on T. □

However, in the following case, we can easily find out that a tree T is a subpansive tree of a CFG G.

Proposition 2. Let $G = (\Sigma, N, P, S)$ be a CFG and $T = (N, E)$ a tree. If for every edge $(A, B) \in E$ of T a production from A to a string that contains B exists in G, then T is a subpansive tree of G.

Proof. Straightforward from the definition of a subpansive tree. □

By Proposition 2, we can design the algorithm *SubpansiveTree* to construct one of the subpansive trees of a given CFG as Figure 1. For the CFG G given in Example 1, the algorithm *SubpansiveTree* constructs just the tree T_1.

SubpansiveTree
Input : A CFG $G = (\Sigma, N, P, S)$.
Output : A subpansive tree (N, E) of G.
$E := \emptyset$; $V := \{S\}$;
while $V \neq N$ **do begin**
  for each $A \in V$ **do**
    if there exists a $A \to \alpha_1 B \alpha_2 \in P$ such that $B \notin V$ **then**
      add (A, B) to E and add B to V;
end /* **while** */
output (N, E);

Fig. 1. An algorithm to construct a subpansive tree from a CFG

4 A Hierarchical CFG

In order to discuss the problem how to learn a CFG in terms of a subpansive tree, we introduce an appropriate class of CFGs, which we coin a name, *hierarchical CFGs* or *HCFGs*, from logic programming [21].

Definition 2. A CFG $G = (\Sigma, N, P, S)$ is said to be *hierarchical* or an *HCFG* if there exists a mapping (called a *level mapping* [21]) $f : N \to \{1, \ldots, |N|\}$ such that, for each production $A \to \alpha \in P$ and every nonterminal B occurring in α, $A \neq B$ implies $f(A) > f(B)$. The class of all HCFGs is denoted by $\mathcal{HCFG}$.

Proposition 3. For a CFG $G = (\Sigma, N, P, S)$, the problem of determining whether G is hierarchical is decidable.

Proof. By Definition 2, there exists a mapping $f : N \to \{1, \ldots, |N|\}$. Since the number of all mappings from N to $\{1, \ldots, |N|\}$ is finite, the problem is decidable. □

We associate an HCFG G with a subpansive tree by the level mapping of G. Let T be a tree (N, E) and f be a mapping from N to $\{1, \ldots, |N|\}$. Then, T is *leveled* by f if $f(A) > f(B)$ holds for each $(A, B) \in E$. Then, it is obvious the following proposition.

Proposition 4. Let G be an HCFG (N, Σ, P, S) and T be a tree (N, E). Then, there exists a level mapping f of G such that T is leveled by f iff T is a subpansive tree of G.

An HCFG has the following useful properties for learning languages constructively.

Proposition 5. Let $G = (\Sigma, N, P, S)$ be a HCFG, T be a subpansive tree for G, and k be the depth of T. Then, for any $w \in L(G)$, there exists a parse tree of w whose depth is at most $k + |w| + 1$.

Proof. Let $T_G(w)$, or simply $T(w)$, be a parse tree of w in G. For an internal node A of $T(w)$, the *yield* of A is defined as a concatenation of the leaves in left-to-right order on the sub-tree of $T(w)$ whose root is A.

For every parent-child relationship (A, B) in $T(w)$, if A and B are labeled by same nonterminal and the length of A's yield is same as one of B's yield, then the tree obtained by replacing the sub-tree of $T(w)$ rooted by A with one rooted by B is also a parse tree of w. Therefore, without loss of generality, we can assume that, if A and B have a parent-child relationship in $T(w)$, then the relationship is also preserved in T or the length of B's yield is less than one of A's yield. Under this assumption, the statement can be easily proved by the induction on the length of w. □

By Proposition 5, if the maximum number of nonterminals occurring in the right-hand side of each production is bounded by some constant, then each parse tree of $w \in L(G)$ is computed in polynomial time.

Consider the language family $\{L(G) \mid G \in \mathcal{HCFG}\}$. The following proposition claims that the language family contains all regular languages.

Proposition 6. For each regular expression E, there exists a HCFG G such that $L(G)$ is equivalent to the language L_E represented by E.

Proof. We prove the statement by the induction on the length of E.

If $E = \phi$, then let P be an empty set. If $E = a$ for some $a \in \Sigma \cup \{\epsilon\}$, then let P be the set $\{S \to a\}$. Furthermore, let G be a CFG $(\Sigma, \{S\}, P, S)$.

Let E_1, E_2 be regular expressions, and let $G_i = (\Sigma, N_i, P_i, S_i)$ be a CFG representing L_{E_i}. Without loss of generality, we can assume that $N_1 \cap N_2 = \emptyset$ and $S \notin N_1 \cup N_2$. Then, we construct P_0 as follows:

1. If $E = (E_1 + E_2)$, then let P_0 be $\{S \to S_1 \mid S_2\}$;
2. If $E = (E_1 E_2)$, then let P_0 be $\{S \to S_1 S_2\}$;
3. If $E = (E_1^*)$, then let P_0 be $\{S \to \varepsilon \mid S S_1\}$.

Furthermore, let G be a CFG (Σ, N, P, S), where $N = \{S\} \cup N_1 \cup N_2$ and $P = P_0 \cup P_1 \cup P_2$. It is easy to show that the set $L(G)$ is equal to L_E, and that if both G_1 and G_2 are in $\mathcal{HCFG}$, so is G. □

The language family $\{L(G) \mid G \in \mathcal{HCFG}\}$ *properly* contains all regular languages: Consider the language $L = \{a^n b^n \mid n \geq 1\}$. Then, L is not regular but there exists an HCFG $G = (\{a, b\}, \{S\}, \{S \to aSb \mid ab\}, S)$ such that $L = L(G)$.

On the other hand, the language family $\{L(G) \mid G \in \mathcal{HCFG}\}$ is contained by the language family $\{L(G) \mid G \text{ is a CFG }\}$. However, the properness remains open.

5 Generalization with a Subpansive Tree

In this section, we introduce the *generalization under a subpansive tree*, which will be essential for the learning algorithm of CFGs discussed in Section 6. We introduce three kinds of generalizations; generalization on strings over $(\Sigma \cup N)^*$, on productions, and on the set of productions.

Definition 3. Let $G = (\Sigma, N, P, S)$ be a CFG and $T = (N, E)$ be a subpansive tree of G. Let α, β be non-empty strings in $(N \cup \Sigma)^+$. Then, α is *more general than* β *under* T, denoted by $\alpha \succeq_T \beta$, if there exist $\beta_1, \beta_3 \in (N \cup \Sigma)^*$, $\beta_2 \in (N \cup \Sigma)^+$ and $A \in N$ satisfy the following:

1. $\alpha = \beta_1 A \beta_3$,
2. $\beta = \beta_1 \beta_2 \beta_3$, and
3. every nonterminal in β_2 is either A or a descendant $B \in N$.

The relation $\succeq_T$ is extended to the reflexive and transitive closure $\succeq_T^*$.

Definition 4. Let $G = (\Sigma, N, P, S)$ be a CFG, and let $p = A \to \alpha$ and $q = A \to \beta$ be productions on Σ and N from a nonterminal $A \in N$. Then, p is *more general than* q *under* T, denoted by $p \succeq_T q$, if $\alpha \succeq_T \beta$.

In the above definition, p and q are not necessarily elements of P for a CFG $G = (\Sigma, N, P, S)$.

Definition 5. Let Π be the set of all the productions on N and Σ, and let π_1 and π_2 be finite subsets of Π. Then, π_1 is *more general than* π_2 *under* T, denoted by $\pi_1 \succeq_T \pi_2$, if for every $p_2 \in \pi_2$ there exists $p_1 \in \pi_1$ that is more general than p_2 under T.

In order to associate the generalization in Definition 5 with a subpansive tree, we introduce the following two mappings Γ_T and Γ_T^n over 2^Π, which is motivated by the T_P operator famous in logic programming (*cf.* [21]).

Definition 6. Let G be a CFG and T be a subpansive tree of G. Also let Π be the set of all the productions on G. Then, the mapping $\Gamma_T : 2^\Pi \to 2^\Pi$ is defined as follows:

$$\Gamma_T(\pi) = \{q \in \Pi \mid q \succeq_T p \text{ for some } p \in \pi\}$$

for $\pi \subseteq \Pi$. Furthermore, for a finite set $\pi \subseteq \Pi$ and non-negative integer n, $\Gamma_T^n(\pi)$ is defined as follows:

$$\begin{aligned} \Gamma_T^0(\pi) &= \pi \\ \Gamma_T^n(\pi) &= \Gamma_T\left(\Gamma_T^{n-1}(\pi)\right) \quad \text{for } n \geq 1. \end{aligned}$$

Now, we investigate some properties of Γ_T. In the remainder of this section, the notion $G = (\Sigma, N, P, S)$ always denotes a reduced HCFG, T is a subpansive tree of G, Π is the set of all the productions on G, and π is a finite subset of Π.

Lemma 1. For each $p \in \Pi$, the size of $\Gamma_T(\{p\})$ is bounded by a polynomial with $|N|$ and $|p|$.

Proof. Let p be a production $A \to \alpha$. By the definition of Γ_T, if $B \to \beta \in \Gamma_T(\{p\})$, then $B = A$ and $\beta \succeq_T \alpha$. Since β is obtained by replacing a substring of α with one nonterminal, there exist at most $|\alpha| \cdot (|\alpha| + 1) \cdot |N|$ candidates for β and is a polynomial with $|N|$ and $|p|$. □

Lemma 2. Let m be the maximum length of $p \in \pi$. Then, the size of $\Gamma_T^n(\pi)$ $(n \geq 1)$ is bounded by a polynomial with $|N|$, m and $|\pi|$.

Proof. It follows from Lemma 1. □

Lemma 3. Let m be the maximum length of $p \in \pi$. Then, there exists a polynomial *poly* such that $\Gamma_T^n(\pi) = \Gamma_T^{n+1}(\pi)$ for each $n \geq poly(m, |N|, |\pi|)$.

Proof. Let p be a production $A \to \alpha \in \Pi$. Then, the number of all productions q satisfying $q \succeq_T p$ is bounded by $|\alpha| \cdot (|\alpha| + 1) \cdot |N|$. Furthermore, since $p \succeq_T q$ and $q \succeq_T r$ implies $p \succeq_T r$, for any non-negative integer n, the size of the set $\Gamma_T^n(\{p\})$ is bounded by $|\alpha| \cdot (|\alpha| + 1) \cdot |N|$. Then, the size of $\Gamma_T^n(\pi)$ is bounded by $m \cdot (m + 1) \cdot |N| \cdot |\pi|$ for each $n \geq 0$.

Furthermore, by the definition of Γ_T^n, Γ_T^n is monotonic with respect to n, that is,

1. $\Gamma_T^i(\pi) \supseteq \Gamma_T^j(\pi)$ for $i \geq j$, and
2. if $\Gamma_T^i(\pi) = \Gamma_T^{i+1}(\pi)$ for some i, then $\Gamma_T^j(\pi) = \Gamma_T^i(\pi)$ for each j $(j \geq i)$.

Hence, we can set $poly(m, |N|, |\pi|)$ to $m \cdot (m + 1) \cdot |N| \cdot |\pi|$. □

Lemma 4. Let p be a production $A \to \alpha \in P$. Then, there exist $w \in L_G(A)$ and non-negative integer n such that $p \in \Gamma_T^n(\{A \to w\})$.

Proof. Since G is reduced, there exists a string $w \in \Sigma^+$ such that $A \Rightarrow_G \alpha \Rightarrow_G^* w$, $\alpha = u_0 A_1 u_1 A_2 \cdots A_m u_m$, $w = u_0 v_1 u_1 v_2 \cdots v_m u_m$, and for each i $(i = 1, 2, \ldots, m)$ $v_i \in L_G(A_i) - \{\varepsilon\}$. Since the string α is more general than w, it holds that $(A \to \alpha) \succeq_T (A \to w)$. Hence, it holds that $p \in \Gamma_T^n(\{A \to w\})$. □

Since the properties of the mapping Γ_T^n given in the above lemmas hold in the class of all CFGs, it is useful to discuss learnability of CFGs in the framework of *identification in the limit* [14]. However, it is not suitable to the polynomial time learning, because the size of the set $\Gamma_T^n(P)$ and the length of each production in it are monotonically increasing on n and the length of the given positive example.

On the other hand, for an HCFG G, the *minimal* generalizations of G can be easily computed. Let G be an HCFG (Σ, N, P, S) and T be a subpansive tree (N, E) of G. Then, for each $A \in N$, T_A denotes a subtree (N_A, E_A) of T satisfying the following conditions:

1. $E_A = \emptyset$ if A is a leaf of T; $E_A = \{(A, B) \in E \mid B \in N\}$ otherwise.
2. N_A is the set consisting of A and all children of A.

Then, the next lemma holds.

Lemma 5. Let $G = (\Sigma, N, P, S)$ be an HCFG and T be a subpansive tree of G. Then, for each $p = A \to \alpha \in P$, there exists $w \in L_G(A)$ and a non-negative integer $n \leq |w|$ such that $p \in \Gamma_{T_A}^n(\{A \to w\})$.

The above lemma tells that it is enough to consider the generalization on the subtree T_A of the subpansive tree for an HCFG. In the following section, we propose the procedure *Generalize* which finds the minimal generalizations by using *nonterminal membership queries*.

6 Learnability of HCFGs with a Subpansive Tree

In this section, we discuss learnability of HCFGs in our setting: the subpansive tree of a target HCFG is given as a background knowledge, and two types of queries, nonterminal membership and equivalence queries are available. Then, for a tree T, let $\mathcal{HCFG}[T]$ denote the family $\{G \in \mathcal{HCFG} \mid T \text{ is a subpansive tree of } G\}$.

Theorem 1. *Let T be a tree. Then, $\mathcal{HCFG}[T]$ is polynomial-time learnable via equivalence and nonterminal membership queries.*

We give the learning algorithm *Learn_HCFG* in the following proof, which is a *constructive* algorithm, because the algorithm makes the most specific hypothesis, and repeats generalizing it until the target CFG is obtained.

Proof. Consider the algorithm *Learn_HCFG* as Figure 2, which mainly consists of two procedures, *Generalize* and *Diagnose*. Note that in the first construction of P nonterminal membership queries $NM_{G_*}(\varepsilon, A)$ for $A \in N$ are invoked.

Learn_HCFG
Input : A subpansive tree $T = (N, E)$ of a target CFG G_*
Output : A CFG $G = (\Sigma, N, P, S)$ such that $L(G) = L(G_*)$
$P := \{A \to \varepsilon \mid \varepsilon \in L_{G_*}(A)\}$; $G := (\Sigma, N, P, S)$;
while $EQ_{G_*}(G)$ replies w **do begin**
  if $w \in L(G) - L(G_*)$ **then** /* w: negative counterexample */
    $P := Diagnose(w, P)$;
  else /* w: positive counterexample */
    $G := Generalize(w, G, T)$;
end /* **while** */
output G;

Fig. 2. The procedure *Learn_HCFG*

The procedure *Diagnose* was proposed by Angluin [1] originally for the class of k-bounded CFG in which the right-hand side of each production has at most k nonterminals for a fixed integer k. In our algorithm, the procedure *Diagnose* finds a false production, for example, $A \to \alpha$ in a hypothesis P, and replaces it with the productions that associate A to strings derived from α in one step. In these productions, the number of occurrences of nonterminals in right-hand side is bounded by some constant depending on the target G_*. By Proposition 5, *Diagnose* finds a false production in polynomial time, and the number of new productions is also bounded by a polynomial.

On the other hand, in order to design the procedure *Generalize* in Figure 3, we prepare two notions, $crr_G(W, A)$ and $red_P(Q)$.

Let $G = (\Sigma, N, P, S)$ be a CFG, W a finite subset of Σ^+ and A a nonterminal in N. Also let P_0 be a set $\{A \to w \mid w \in W\}$. Then, the *correct generalizations* of W with respect to A, denoted by $crr_G(W, A)$, is the set R satisfying the following conditions:

1. $R \succeq_T^* P_0$,
2. $\Gamma_{T_A}(R) = R$, and
3. for each $A \to u_0 B_1 u_1 \cdots u_{m-1} B_m u_m$ in R, there exists $u_0 v_1 u_1 \cdots u_{m-1} v_m u_m$ in W such that $v_i \in L_G(B_i)$ for each $1 \leq i \leq m$.

Let P and Q be sets of productions. Then, the *reduced set* $red_P(Q)$ *of* Q *with* P is a subset R of Q which satisfies the following: (1) for any production $A \to \alpha$ in Q, $A \Rightarrow_{R \cup P}^* \alpha$, and (2) for any $R' \subset R$, there is $A \to \alpha$ in Q such that $A \not\Rightarrow_{R' \cup P}^* \alpha$. Here, $\Rightarrow_P$ for a set of productions P implicitly means $\Rightarrow_G$ for a CFG whose set of productions is P.

Generalize
Input : $w \in \Sigma^+$; a CFG $G_0 = (\Sigma, N, P_0, S)$; a tree T; a target CFG G_*;
/* T is a subpansive tree of G_* */
Output : A CFG $G = (\Sigma, N, P, S)$ such that $L(G) \supseteq L(G_0) \cup \{w\}$.
$P := P_0$;
for each $A \in N$ **do begin**
    $W_A := \emptyset$;
    for each substring u of w **do**
        if $NM_{G_*}(u, A)$ replies *yes* **then** $W_A := W_A \cup \{u\}$;
end /* **for each** */
$mark := \{A \in N \mid A$ is a leaf of $T\}$;
for each $A \in mark$ **do** $P := P \cup red_P(crr_G(W_A, A))$;
while $mark \neq N$ **do begin**
    if all children of $A \in (N - mark)$ are in $mark$ **then**
        $P := P \cup red_P(crr_G(W_A, A))$;
    $mark := mark \cup \{A\}$;
end /* **while** */
output G;

Fig. 3. The procedure *Generalize*

In the procedure *Generalize* in Figure 3 we deal with *crr* and *red* directly. Note that we can check the condition 3 in *crr* by using nonterminal membership queries.

In the **while** loop the enumeration of nonterminals is executed in a bottom-up manner, and the number of iterations of the loop is bounded by the depth of the given subpansive tree T. Furthermore, by Lemma 1, 2, and 3, the set $red_P(crr_G(W_A, A))$ can be computed in polynomial time with $|P|$ and $|N|$. Therefore, the amount time of running *Generalize* is bounded by a polynomial

with $|G_0|$, $|w|$ and $|T|$. Since the size of P is bounded by a polynomial at each step of iteration in the learning algorithm *Learn_HCFG*, it terminates in polynomial time. □

On the other hand, let DNF_n be a family of DNF formulas over n Boolean variables. Then, we can show the following theorem.

Theorem 2. *For each $n \geq 0$, there exists a subpansive tree such that, if $\mathcal{HCFG}$ is polynomial-time predictable, then so is DNF_n.*

Proof. Let $d = t_1 \vee \cdots \vee t_m$ be a DNF formula over n Boolean variables $\{x_1, \ldots, x_n\}$. Then, let T be a tree:

$$T = \left(\{S, X_1, \ldots, X_n\}, \bigcup_{1 \leq i \leq n} \{(S, X_i)\} \right).$$

We show the statement by the *prediction-preserving reduction* [24].

The *word transformation* f is an identity function, that is, $f(e) = e$. The *representation transformation* g is defined as follows. First, for each term t_i in d, $h(t_i)$ is a production $S \to w_1 \cdots w_n$, where

$$w_j = \begin{cases} 1 & \text{if } t_i \text{ contains } x_j, \\ 0 & \text{if } t_i \text{ contains } \overline{x_j}, \\ X_i & \text{otherwise.} \end{cases}$$

Furthermore, G_0 is the following set of productions $\bigcup_{j=1}^{n} \{X_j \to 0|1\}$. Then, for each $d \in \mathrm{DNF}_n$, a representation transformation g is defined as:

$$g(d) = \bigcup_{i=1}^{m} h(t_i) \cup G_0.$$

Note here that T is a subpansive tree of $g(d)$, and the size of $g(d)$ is at most $|d|(n+1) + 4n$.

For $e \in \{0,1\}^n$, it holds that

$$\begin{aligned} & e \text{ satisfies } d \ (e \in d) \\ \Longleftrightarrow\ & \text{there exists an } i \ (1 \leq i \leq m) \text{ such that } e \text{ satisfies } t_i \\ \Longleftrightarrow\ & \text{there exists an } i \ (1 \leq i \leq m) \text{ such that } e \in L(\{g(t_i)\} \cup G_0) \\ \Longleftrightarrow\ & f(e) \in L(g(d)). \end{aligned}$$

Hence, there exists a subpansive tree T that predicting DNF_n reduces to predicting $\mathcal{HCFG}$. By using the property of prediction-preserving reduction given by Pitt and Warmuth [24], we can conclude the statement. □

Hence, we conjecture that $\mathcal{HCFG}$ is not polynomial-time predictable. According to [3, 20, 24], we may also conjecture that $\mathcal{HCFG}$ not polynomial-time learnable via equivalence queries alone.

7 Concluding Remarks

We have proposed a new scenario for learning a CFL with a subpansive tree. The subpansive tree characterizes a language by sublanguages, modeling a background knowledge in natural language acquisition.

Through the level mapping technique, we have defined the subclass of CFGs that we named hierarchical CFGs, and associated with a subpansive tree. We have shown that all regular languages are properly contained in the languages generated by the class of hierarchical CFGs, and that the class of hierarchical CFGs is efficiently learnable by nonterminal membership queries and equivalence queries. We also have shown that the class of hierarchical CFGs seems hard to predict in polynomial time.

Consider, finally, a subpansive tree that lacks some nodes, an *incomplete* subpansive tree. An incomplete subpansive tree is constructed from a proper subset of nonterminals of a given complete subpansive tree. This is comparable to a natural language in which only some parts of speech are fixed. Arguably, any natural language has contentive categories, such as noun, verb, adjective, and adposition. These reflect epistemological cognitive categories that roughly depict entities, actions, properties, and relations. In an influential hypothesis of linguistic theory [11, 13], a factor of cross-linguistic variation is attributed to other minor categories, such as determiner and complementizer, which may be absent in some languages. Thus, grammatical inference with incomplete subpansive trees comes closer to natural language acquisition, and if it can succeed, we may also find a useful application in computer science, such as extracting from a program metafunctions and/or subfunctions that are recurrently used but unknown.

Yet, to derive a complete subpansive tree from incomplete ones, we have to consider all the complete subpansive trees that are possible, the number of which is proportional to exponential of nonterminals of the incomplete subpansive tree. This seems rather formidable, but one of our future research topics is to discuss the learnability of languages with incomplete subpansive trees and its implications. Another is to restrict the membership queries to terminals, and drop the equivalence queries all together. After all, humans, even adults, are unconscious about the precise structure of a given sentence, and they cannot completely describe their grammar (target) or know how much a child has developed his or hers (hypothesis); let alone evaluating the differences and giving counterexamples, though the child may be told what can be said and what cannot. In the meantime, it also pays to investigate whether or not the class of hierarchical CFGs is a proper subclass of CFGs, a task we are not yet able to cover in this preliminary study.

References

1. D. Angluin: *Learning k-bounded context-free grammars*, Technical Report YALEU/DCS/RR-557, Yale University (1987).

2. D. Angluin: *Learning regular sets from queries and counterexamples*, Information and Computation **75**, 87–106 (1987).
3. D. Angluin: *Query and concept learning*, Machine Learning **2**, 319–342 (1988).
4. L. Bloomfield: *Language*, Holt, Rinehart & Winston, (1933).
5. P. Berman and R. Roods: *Learning one-counter languages in polynomial time*, Proc. 28th IEEE Symposium on Foundation of Computer Science, 61–67 (1987).
6. S. Crain and D. Lillo-Martin: *Introduction to linguistic theory and natural language acquisition*, Blackwell (1999).
7. N. Chomsky: *Syntactic structures*, Mouton (1957).
8. N. Chomsky: *Formal properties of grammars*, *Handbook of mathematical psychology* (eds.) R. D. Luce, R. R. Bush and E. Galanter, J. Wiley & Sons, 323-418 (1963).
9. N. Chomsky: *Aspects of the theory of syntax*, MIT Press (1965).
10. N. Chomsky: *Remarks on nominalization*, *Readings in English transformational grammar* (eds.) R. A. Jacobs and P. S. Rosenbaum, Ginn & Co,.184-221 (1970).
11. N. Chomsky: *Lectures on government and binding*, Foris Publications (1981).
12. C. Domingo and V. Lavín: *The query complexity of learning some subclasses of context-free grammars*, Proc. 2nd European Conference on Machine Learning, 404–414 (1995).
13. N. Fukui: *Theory of projection in syntax*, CSLI Publications (1995).
14. E. Gold: *Language identification in the limit*, Information and Control **10**, 447–474 (1967).
15. C. D. L. Higuera: *Characteristic sets for polynomial grammatical inference*, Machine Learning **27**, 125–138 (1997).
16. J. E. Hopcroft and J. D. Ullman: *Introduction to automata theory, languages and computation*, Addison-Wesley Publishing (1979).
17. H. Ishizaka: *Polynomial time learnability of simple deterministic languages*, Machine Learning **5**, 151–164 (1990).
18. R. Jackendoff: $\overline{X}$ *syntax: A study of phrase structure*, MIT Press (1977).
19. X. Ling: *Learning and invention of Horn clause theories – a constructive method –*, Methodologies for Intelligent Systems **4**, 323–331 (1989).
20. N. Littlestone: *Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm*, Machine Learning **2**, 285–318, 1988.
21. J. W. Lloyd: *Foundations of logic programming (second, extended edition)*, Springer-Verlag (1987).
22. E. Mäkinen: *On the structural grammatical inference problem for some classes of context-free grammars*, Information Processing Letters **42**, 1–5 (1992).
23. S. Pinker: *Language learnability and language development*, Harvard University Press (1984).
24. L. Pitt and M. K. Warmuth: *Prediction preserving reduction*, Journal of Computer System and Science **41**, 430–467, 1990.
25. Y. Sakakibara: *Learning context-free grammars from structural data in polynomial time*, Theoretical Computer Science **76**, 223–242 (1990).
26. Y. Sakakibara: *Efficient learning of context-free grammars from positive structural examples*, Information and Computation **97**, 23–60 (1992).
27. H. Sakamoto: *Language learning from membership queries and characteristic examples*, Proc. of 6th International Workshop on Algorithmic Learning Theory, LNAI **997**, 55–65 (1995).
28. N. Sugimoto, K. Hirata and H. Ishizaka: *Constructive learning of translations based on dictionaries*, Proc. of 7th International Workshop on Algorithmic Learning Theory, LNAI **1160**, 177–184 (1996).