社団法人　情報処理学会　研究報告
IPSJ SIG Technical Report

2006－AL－108（2）
2006／9／27

制限のあるサンプル分布における文脈自由文法の学習可能性

但馬 康宏，小谷 善行

東京農工大学 工学部 情報工学科

あらまし　文脈自由言語は，所属性質問を用いても多項式時間厳密学習が難しいことが知られている．さらに，その部分言語族に対しても，代表的なものについては，同様の否定的な結果が知られている．本研究では，特殊な例集合と質問により多項式時間で厳密学習可能な文脈自由言語の部分族と学習アルゴリズムを示す．この言語族は，過去に同様の設定において学習可能性が示された言語族を真に含む．さらに，その学習アルゴリズムに与えられる特殊な例集合をランダムサンプルから構成可能とする条件を示し，ランダムサンプルと所属性質問による確率的近似学習可能性を示す．

A context-free learning on restricted sample distributions

Yasuhiro TAJIMA and Yoshiyuki KOTANI

Department of Computer Science,
Tokyo University of Agriculture and Technology

Abstract　Polynomial time learning of a context-free language is hard even if the learner can use membership queries. Moreover, the same hardness has been shown for some major subclasses of context-free languages. In this paper, we show a subclass of context-free languages which is polynomial time learnable via membership queries, equivalence queries and a special finite examples. In addition, we show some conversion algorithm from random examples to the set of special examples. From this result, our subclass of context-free languages is polynomial PAC learnable via membership queries and random examples.

1　Introduction

Learning via queries is one of the most important model for grammatical inference. From early studies of this area, the model of learning in the limit has been studied. This model rules the goal of the learning, but definitions of time complexity for this learning model are referred in the later studies[8]. Nevertheless, negative results[3][5] have been shown in this learning model.

In the query learning model, Angluin has shown the polynomial time query learnability of regular sets[2]. Following this result, some extensions of the polynomial time query learnability has been shown[7][9][10]. Unfortunately, there are negative results in polynomial time query learning. With an assumption of one-way functions, the class of context-free languages is not polynomial time learnable even though the learner uses membership and equivalence queries[4].

In this paper, we show a polynomial time learning algorithm for a subclass of context-free grammars with membership queries, equivalence queries and a set of characteristic examples. The subclass of context-free grammars is our newly defined and a grammar in this subclass is called a partitionable and unique path context-free grammar. In the class of partitionable context-free grammars, the learner can decide immediately that, for any pair of rules, they can not be contained in the same grammar. In a unique path context-free grammar G, for every nonterminal A, there exist $u \in \Sigma^*$ and $v \in \Sigma^*$ such that $S \overset{*}{\underset{G}{\Rightarrow}} uAv$ and $A \overset{*}{\underset{G}{\Rightarrow}} w$ iff $uwv \in L(G)$ for any $w \in \Sigma^*$. With this property, the learner can find whether A can derive w or not by a membership query for uwv. We show that the class of simple deterministic grammars is contained in a class of partitionable and unique path context-free grammars. In [6], it has been shown that the class of simple deterministic grammar is not identifiable in the limit from polynomial time and data. Thus, our newly defined subclass is not identifiable in the limit from polynomial time and data.

2 Preliminaries and definitions

A context-free grammar (CFG for short) is denoted by $G = (N, \Sigma, P, S)$, where N is a finite set of nonterminals, Σ is a finite set of terminals, P is a finite set of production rules (in short, rules), and $S \in N$ is the start symbol. Let ε be the word whose length is 0. We assume that every grammar in this paper is ε-free. For a set B, we denote by $|B|$ the cardinality of B. For a string β, we denote by $|\beta|$ the length of β. We assume that every rule is of the form $A \to a\beta$ where $A \in N$, $a \in \Sigma$, $\beta \in N^*$ such that $|\beta| \le 2$ (Greibach normal form). Let $B^n = \{w \in B^* \mid |w| = n\}$ and $B^{\le n} = \{w \in B^* \mid |w| \le n\}$ for a finite set B.

The derivation from $\alpha_1 A \alpha_2$ to $\alpha_1 \beta \alpha_2$ is denoted by $\alpha_1 A \alpha_2 \underset{G}{\Rightarrow} \alpha_1 \beta \alpha_2$ where $\alpha_1, \alpha_2 \in (N \cup \Sigma)^*$ and $A \to \beta$ in P. The reflexive and transitive closure of $\underset{G}{\Rightarrow}$ is denoted by $\overset{*}{\underset{G}{\Rightarrow}}$ or $\overset{*}{\Rightarrow}$. If $\gamma \overset{*}{\underset{G}{\Rightarrow}} w$ and $w \in \Sigma^*$ then w is a word generated from $\gamma \in (N \cup \Sigma)^*$ by G. The language generated from γ by G is denoted by $L_G(\gamma) = \{w \in \Sigma^* \mid \gamma \overset{*}{\underset{G}{\Rightarrow}} w\}$. A word generated from S is called a word generated by G and the language generated by G is denoted by $L(G) = L_G(S)$. A language generated by a CFG G is called a context-free language (CFL for short).

A nonterminal $A \in N$ is said to be reachable if $S \overset{*}{\underset{G}{\Rightarrow}} wA\beta$ for some $w \in \Sigma^*$, $\beta \in N^*$, and A is said to be live if $L_G(A) \neq \emptyset$.

Let $u, v \in \Sigma^*$ and L be a language. We define $u \backslash L = \{w \in \Sigma^* \mid uw \in L\}$, $L/v = \{w \in \Sigma^* \mid wv \in L\}$, and $u \backslash L/v = \{w \in \Sigma^* \mid uwv \in L\}$.

Let $\simeq$ be an equivalence relation on a set P and $D(a)$ be the equivalence class which contains $a \in P$. We denote by $P/\simeq$ the family of equivalence classes. That is $P/\simeq = \{D(a) \mid a \in P\}$.

If a CFG $G = (N, \Sigma, P, S)$ satisfies the followings, then G is called a nondeterministic regular grammar.

- Let $A, B \in N$ and $a \in \Sigma$. There are only rules of the form $A \to a$ or $A \to aB$ in P.

The language generated by a regular grammar is called a regular language. In addition, if $(A \to aB) \in P$ and $(A \to aC) \in P \Rightarrow B = C$ holds, then G is called a (deterministic) regular grammar.

If a CFG $G = (N, \Sigma, P, S)$ satisfies the followings, the G is called a simple deterministic grammar (SDG for short).

- Every rule in P is of the form $A \to a\beta$ where $A \in N$, $a \in \Sigma$ and $\beta \in N^*$ such that $|\beta| \le 2$.
- Let $A \in N$, $a \in \Sigma$ and $\beta, \gamma \in N^*$. If both of $A \to a\beta$ and $A \to a\gamma$ is in P then $\beta = \gamma$.

The language generated by an SDG is called a simple deterministic language (SDL for short).

2.1 Partitionable CFGs

We define a class of CFG called partitionable CFG in the following. We have shown that SDLs are polynomial time learnable with membership queries if a set of representative samples is given[11]. In this paper, we extend the algorithm for the following subclasses of CFLs.

Definition 1 *Let N_n be a set of nonterminals such that $|N_n| = n(\geq 1)$, Let $P_U(n) = \{A \to a\beta \mid A \in N_n, a \in \Sigma, \beta \in N_n^*, |\beta| \leq 2\}$. This is the set of all possible rules in Greibach normal form of a CFG.*

Assume an equivalence relation $\sim_n$ on $P_U(n)$ for every n such that $(A_1 \to \beta) \sim_i (A_2 \to \gamma) \Rightarrow (A_1 \to \beta) \sim_j (A_2 \to \gamma)$ for $i < j$ where either $P_U(i)$ or $P_U(j)$ contains both of $A_1 \to \beta$ and $A_2 \to \gamma$. We denote $\sim_\infty$ by $\sim$.

The subclass of CFGs $\mathcal{G}(\sim)$ is defined as the following. Every $G = (N, \Sigma, P, S) \in \mathcal{G}(\sim)$ satisfies that if $(A \to \beta) \in P$ then any $(B \to \gamma) \notin P$ such that $(A \to \beta) \sim (B \to \gamma)$. In other words, $G \in \mathcal{G}(\sim)$ consists of rules selected at most one rule from every equivalence class.

We call $\mathcal{G}(\sim)$ a $\sim$-partitionable CFG or a partitionable CFG, and the language class generated by $\mathcal{G}(\sim)$ is called $\sim$-partitionable CFLs or partitionable CFLs.

For $G = (N, \Sigma, P, S) \in \mathcal{G}(\sim)$ and $(A \to \beta) \in P$, we define $D_\sim(A \to \beta) = \{(B \to \gamma) \in P \mid (A \to \beta) \sim (B \to \gamma)\}$.

A class of $\mathcal{G}(\sim)$ is obviously a subclass of CFGs. In our learning model, we assume that $\sim$ is *a priori* knowledge of the learner. Thus, the learner can check whether a hypothesis is $\sim$-partitionable or not, immediately.

For example, let $\sim$ be an equivalence relation such that

$$(A \to a\beta) \sim (A \to a\gamma)$$

for $A \in N$, $a \in \Sigma$ and $\beta, \gamma \in N^*$. Then $\sim$-partitionable CFGs is the class of SDGs.

2.2 A unique path and a unique path complete set

We define the following properties for CFGs.

Definition 2 *Let $G = (N, \Sigma, P, S)$ be a CFG and $A \in N$. A is called a unique path nonterminal, if there exist a pair $u, v \in \Sigma^*$ such that*

- *$S \overset{*}{\Rightarrow} uAv$ and*
- *$L_G(A) = u\backslash L_G/v$.*

The derivation $S \overset{*}{\Rightarrow} uAv$ is called a unique path of A.

If every $A \in N$ is a unique path nonterminal, then G is called a unique path CFG and $L(G)$ is called a unique path CFL.

For $w \in \Sigma^*$ and $A \in N$ of a unique path CFG, we can distinguish whether $w \in L_G(A)$ or not by a membership query for $uwv \in \Sigma^*$ where $S \overset{*}{\Rightarrow} uAv$ is a unique path of A.

Definition 3 *Let $G = (N, \Sigma, P, S)$ be a unique path CFG. A finite set $U \subset \Sigma^*$ is a* unique path complete set*(UPCS for short), if there exists $u_A w v_A \in U$ for every $A \in N$ such that*

- *$u_A, w, v_A \in \Sigma^*$, and*
- *$S \overset{*}{\Rightarrow} u_A A v_A$ is a unique path of A.*

We note that either $w \in L(G)$ or $w \notin L(G)$ is admissible for any UPCS U and any $w \in U$.

For example, assume $G = (N, \Sigma, P, S)$ is an SDG such that $N = \{S, A, B, C\}$, $\Sigma = \{a, b\}$, and

$$P = \{S \to aA,\ A \to aAB,\ A \to b,\ B \to b\}.$$

Then, the set $\{aabb\}$ is a UPCS, because $S \underset{G}{\overset{*}{\Rightarrow}} \varepsilon S \varepsilon$ is a unique path of S, $S \underset{G}{\overset{*}{\Rightarrow}} aA$ is a unique path of A, and $S \underset{G}{\overset{*}{\Rightarrow}} aabB$ is a unique path of B.

For a regular grammar $G = (N, \Sigma, P, S)$, a UPCS equals the live-complete set which is used to show that the class of regular sets is polynomial time learnable with membership queries and a live-complete set[1].

2.3 Queries

In this paper, we assume that the learner can use the following queries. Let L_t be the target language and Rp is the representation class by whom the learner guesses a hypothesis.

Membership query takes $w \in \Sigma^*$, and reply with "yes" if $w \in L_t$ or "no" if $w \notin L_t$.

Equivalence query
takes a hypothesis grammar G in Rp. If $L(G) = L_t$ then "yes" is returned. If $L(G) \neq L_t$ then "no" and a counterexample $w \in (L_t - L(G)) \cup (L(G) - L_t)$ are returned.

3 The learning algorithm

We prove the following theorem by showing a learning algorithm.

Theorem 4 *We denote by $\mathcal{T}$ the language generated by G such that*

- *$G \in \mathcal{G}(\sim)$ and G is a unique path CFG.*

Then, $\mathcal{T}$ is polynomial time learnable by the class of $\mathcal{G}(\sim)$ via a UPCS, equivalence queries and membership queries.

Throughout this paper, L_t denotes the target language and we assume $G_t = (N_t, \Sigma, P_t, S_t)$ as the grammar which satisfies $L(G_t) = L_t$. In addition, U denotes the given UPCS.

The learning algorithm for this language class is as follows.

1. Construct nonterminal candidates from the given UPCS and all possible rules.
2. Check consistency of rules by check word set W.
3. Construct *base grammars*.
4. Ask equivalence queries for every hypothesis in base grammars and update W.

We describe details of each step and prove the correctness of this algorithm.

3.1 Candidates of nonterminals and possible rules

From the definition of the UPCS, $C_h = \{(u,v) \in \Sigma \times \Sigma \mid uwv \in U, w \in \Sigma^*\}$ contains the pair of $u_A \in \Sigma^*$ and $v_A \in \Sigma^*$ such that

$$S \overset{*}{\Rightarrow} u_A A v_A$$

and $L_{G_t}(A) = u_A \backslash L_t / v_A$ for every $A \in N_t$. We denote by (u_A, v_A) such corresponding element in C_h to $A \in N_t$. The learner makes an equivalence relation on C_h such that no equivalence class contains both of (u_A, v_A) and (u_B, v_B) for every pair of $A, B \in N_t$. The equivalence classes are candidates of nonterminals in the hypothesis.

To construct such an equivalence relation, we use $W \subset \Sigma^*$ of a check word set. Let $(u_1, v_1) \in C_h$ and $(u_2, v_2) \in C_h$, we define $(u_1, v_1) \overset{W}{=} (u_2, v_2)$ iff $u_1 w u_2 \in L_t \iff u_2 w v_2 \in L_t$ for any $w \in W$. Obviously, we can decide whether $(u,v) \overset{W}{=} (x,y)$ or not by $|W|$ times membership queries for any $(u,v) \in C_h$ and $(x,y) \in C_h$.

The learner initializes $W = \Sigma$ and extends it to make a distinction between every $A \in N_t$ and $B \in N_t$. We refer to the extension method in the following section. We denote $C_h / \overset{W}{=}$ by N_h.

The learner construct the rule set $P_U(|N_h|)$ from N_h, that is

$$P_U(|N_h|) = \{A_1 \to a\beta \mid A_1 \in N_h, a \in \Sigma, \beta \in N_h^{\leq 2}\}.$$

These rules are candidate rules for the hypothesis.

3.2 Consistency check

Let A be an equivalence class in N_h, and $w \in W$. From the definition of N_h, it holds that $uwv \in L_t$ iff $xwy \in L_t$ for any pair of $(u,v), (x,y) \in A$. Thus, we can define $T(A, w) = 1$ if $uwv \in L_t$ where $(u,v) \in A$, otherwise $T(A,w) = 0$. In addition, for $B_1 B_2 \cdots B_m \in N_h^m$ where $B_1, B_2, \cdots, B_m \in N_h$, we define $T(B_1 B_2 \cdots B_m, w) = 1$ if there exists $w_1, w_2, \cdots, w_m \in \Sigma^*$ such that $T(B_1, w_1) = 1$, $T(B_2, w_2) = 1$, $\cdots$, $T(B_m, w_m) = 1$ and $w_1 w_2 \cdots w_m = w$.

The learner decides that a rule $A \to a\beta$ in $P_U(|N_h|)$ is not appropriate for the hypothesis if $T(A, aw) \neq T(\beta, w)$ for some $w \in W$. In the learning algorithm, the learner executes the following steps.

(step 1) Let $P_h = P_U(|N_h|)$.

(step 2) For $A \in N_h$ and $a \in \Sigma$, if $T(A, a) = 0$ then remove $A \to a$ from P_h.

(step 3) For $A \to a\beta$ where $A \in N_h$, $a \in \Sigma$ and $\beta \in N_h^+$, if $T(A, aw) \neq T(\beta, w)$ for some $w \in W$ then remove $A \to a\beta$ from P_h.

Then, the following lemma holds.

Lemma 5 *Let E be the equivalence class in N_h such that $(\varepsilon, \varepsilon) \in E$. Let $G_h = (N_h, \Sigma, P_h, E)$ be a CFG. For any $A \in N_h$ and $w \in W$, it holds that $T(A, w) = 1$ iff $w \in L_{G_h}(A)$.*

Proof: We prove this lemma by the induction of $|w|$ for $w \in W$.

Base step : When $w = a \in \Sigma$, from the description of the consistency check (step 2), it holds that $T(A, a) = 1$ iff $(A \to a) \in P_h$ for any $A \in N_h$ and $a \in \Sigma$.

Induction step : Assume that this lemma holds for $w \in \Sigma^*$ such that $|w| = n$. We are concerned with aw where $a \in \Sigma$. From the definition of the consistency check (step 3), it holds that $T(A, aw) = T(\beta, w)$ where $(A \to a\beta) \in P_h$. On the other hand, it holds that $T(\beta, w) = 1$ iff $w \in L_{G_h}(\beta)$ from the assumption of this induction. Thus, this lemma holds for aw. □

3.3 Base grammars

If we can select the correct rule from $D_\sim(A \to \beta)$ for each $(A \to \beta) \in P_h$, we can construct the correct hypothesis grammar. To complete such selection in polynomial time, we define a set of base grammars whose rules are a subset of P_h.

Definition 6 *Let $P_h / \sim = \{D_1, D_2, \cdots, D_n\}$ and $<_i$ be an arbitrary total order on D_i for each $i = 1, 2, \cdots, n$. For every $j = 1, 2, \cdots, n$, let $(B_j \to \beta_j) \in D_j$ such that $(B_j \to \beta_j) <_j (C \to \gamma)$ for any $(C \to \gamma) \in D_j$. We define $P(A \to \alpha) = \{(B_j \to \beta_j) \in D_j \mid (A \to \alpha) \in D_k, j = 1, 2, \cdots, k-1, k+1, \cdots, n\} \cup \{A \to \alpha\}$. Then, the set of base grammars $\mathcal{B}$ is*

$$\mathcal{B} = \{G = (N_h, \Sigma, P(A \to \alpha), E) \in \mathcal{G}(\sim) \mid (A \to \alpha) \in P_h\}$$

where E is the equivalence class in N_h such that $(\varepsilon, \varepsilon) \in E$.

3.4 Equivalence check and update of the check word set

The learner makes equivalence queries for every $G \in \mathcal{B}$. If an equivalence query is replied with "yes" then the learner outputs the correct hypothesis and terminates. On the other hand, every equivalence queries are replied with "no", then let EX be the set of all counterexamples returned by the equivalence queries.

Then, the learner updates W as

$$W \;:=\; W \cup \{w \in \Sigma^+ \mid u, v \in \Sigma^*, uwv \in EX\}.$$

The learner goes back to the first step with the updated W. In Fig.1, we show the pseudo-code of this algorithm A_1.

3.5 Correctness and complexity

This learning algorithm outputs a correct hypothesis if an equivalence query replies with "yes." Thus, we show that the algorithm terminates in polynomial time.

Lemma 7 *Let P_h be the set of rules which is checked the consistency with W. Suppose that $W' \neq \emptyset$ in our learning algorithm and let $W_2 = W \cup \{w \in \Sigma^+ \mid x, y \in \Sigma^*, xwy \in W'\}$.*

```
INPUT: a UPCS $U$;
OUTPUT: a hypothesis $G \in \mathcal{G}(\sim)$;
begin
    $C_h := \{(x,z) \mid x \in \Sigma^+, y,z \in \Sigma^*, x \cdot y \cdot z \in U\} \cup \{(\varepsilon,\varepsilon)\}$;
    $W := \{a \in \Sigma\}$;
    repeat
        $N_h := C_h / \overset{W}{=}$;
        $P_h := P_U(|N_h|)$;
        check the consistency and delete inappropriate rules from $P_h$;
        let $\mathcal{B}$ be the set of base grammars;
        $W' := \emptyset$;
        for all $G \in \mathcal{B}$ do
            if $L(G) = L_t$ then
                output $G$ and terminate;
            else
                $W' := W' \cup \{w\}$;
            endif
        done
        for all $w \in W'$ do
            $W := W \cup \{y \in \Sigma^+ \mid x,z \in \Sigma^*, x \cdot y \cdot z = w\}$;
        done
    until (forever))
end.
```

Figure 1: The learning algorithm A_1

Let P'_h be the consistency checked rules with W_2.

Then, either of the followings holds.

- *$C_h / \overset{W_2}{=}$ is finer than $C_h / \overset{W}{=}$.*
- *There exists a rule $A \to \beta$ such that $(A \to \beta) \in P_h$ but $(A \to \beta) \notin P'_h$.*

Proof: Assume that $W' \neq \emptyset$ but $C_h / \overset{W_2}{=}$ is not finer than $C_h / \overset{W}{=}$, i.e. $(C_h / \overset{W_2}{=}) = (C_h / \overset{W}{=})$.

If there exists $w' \in W'$ such that $w' \in L(G_0) - L_t$ for some $G_0 \in \mathcal{B}$, then a derivation $E \underset{G_0}{\overset{*}{\Rightarrow}} w'$ exists, here $E \in N_h$ is the start symbol of G_0. On the other hand, $w' \notin L(G'_h)$ where $G'_h = (N_h, \Sigma, P'_h, E)$ since $T(E, w') = 0$, from Lemma 5. Thus, there exist $A \in N_h$, $a \in \Sigma$ and $\beta \in N_h^*$ such that $E \underset{G_0}{\overset{*}{\Rightarrow}} u_0 A v_0 \underset{G_0}{\Rightarrow} u_0 a \beta v_0 \underset{G_0}{\overset{*}{\Rightarrow}} u_0 a w_1 v_0 = w'$ and $T(A, a w_0) \neq T(\beta, w_0)$ where $u_0, v_0 \in \Sigma^*$ and $w_0 \in \Sigma^+$. Obviously, $(A \to a\beta) \notin P'_h$ from the definition of the consistency check. Thus, this lemma holds.

Contrary, we suppose $w' \in W'$ such that $w' \in L_t - L(G_1)$ for some $G_1 \in \mathcal{B}$. It holds that $T(E, w') = 1$ and $w' \in L(G'_h)$ from Lemma 5 where $G'_h = (N_h, \Sigma, P'_h, E)$. Now, we have the assumption that $C_h / \overset{W_2}{=}$ is not finer than $C_h / \overset{W_2}{=}$. Thus, there exist $A \in N_h$, $\beta, \gamma \in N_h^*$ and $a \in \Sigma$ such that

- $E \underset{G'_h}{\overset{*}{\Rightarrow}} u_1 A v_1 \underset{G'_h}{\Rightarrow} u_1 a \beta v_1 \underset{G'_h}{\overset{*}{\Rightarrow}} u_1 a w_1 v_1 = w'$,
- $E \underset{G_1}{\overset{*}{\Rightarrow}} u_1 A v_1 \underset{G_1}{\Rightarrow} u_1 a \gamma v_1$, and
- $T(\gamma, w_1) = 0$ (Fig.2).

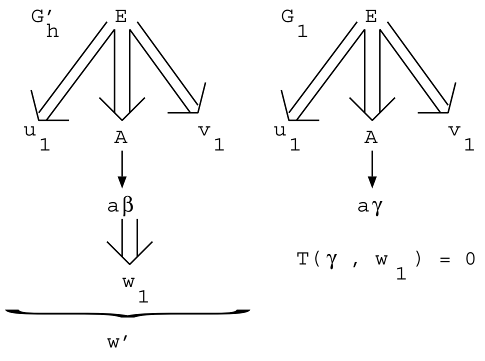

Figure 2: A derivation for a positive example

This concludes that P'_h does not contain $A \rightarrow a\gamma$. Thus, this lemma also holds in this case. $\Box$

Lemma 8 *The time complexity of the learning algorithm is bounded by a polynomial of the size of the given UPCS, and the maximum length of counterexamples.*

Proof: Let l_m be the maximum length of counterexamples. Let l_s and s be the maximum length of words in U and the cardinality of U, respectively. In the learning algorithm, both of $|P_h|$ and $|N_h|$ are monotone decreasing from Lemma 7. Thus, the main loop of the algorithm repeats at most $|C_h||P_U(|C_h|)|$ times. Now, $|C_h||P_U(|C_h|)| \leq |C_h|^4|\Sigma| \leq (l_s^2 s)^4|\Sigma|$. The size of $\mathcal{B}$ and $|W'|$ is bounded by a polynomial of $|P_h|$. Thus, the increase of W is also bounded by a polynomial. $\Box$

4 UPCS Construction from random examples

In probably approximately correct (PAC) learning, the learner can decide whether a hypothesis is equivalent to the target language or not by consistency checking with polynomial number of examples[2]. Thus, if the learner can obtain a UPCS from polynomial number of random examples then the learning algorithm in Fig.1 can be converted to a PAC learning algorithm via membership queries and random examples.

In [11], we have proposed that a set of representative samples can be obtained from m random samples such that

$$m > \frac{1}{d}\log\left(\frac{|P_t|}{\delta}\right)$$

where $|P_t|$ is the size of the target grammar, δ is the given confidence parameter and d is the minimum probability such that a rule is used in the derivation of the sample word.

With the same analysis above, the learner can obtain a UPCS from m' random examples such that

$$m' > \frac{1}{p}\log\left(\frac{|N_t|}{\delta}\right)$$

where $|N_t|$ is the cardinality of the set of nonterminals of the target grammar, and p is the minimum occurring probability of unique paths of any $A \in N_t$.

Now, we obtain the following theorem.

Theorem 9 *We denote by $\mathcal{T}$ the language generated by G such that*

- *$G \in \mathcal{G}(\sim)$ and G is a unique path CFG.*

Then, there exists a learning algorithm which outputs a correct hypothesis with the probability $1-\delta$ using

- *membership queries,*
- *polynomial number of random examples, and*
- *the minimum occurring probability of unique paths of any $A \in N_t$.* $\Box$

5 Conclusions

In this paper, we show that a class of partitionable and unique path CFGs is polynomial time learnable from

- equivalence queries, membership queries, and
- a unique path complete set.

In addition, it has been shown that we can obtain a PAC like learning algorithm by replacing an equivalence query and a UPCS to polynomial number of random examples.

References

[1] Angluin, D.: A note on the number of queries needed to identify regular languages. Info. and Cont., **51** (1981) 76–87

[2] Angluin, D.: Learning regular sets from queries and counterexamples. Info. and Comp., **75** (1987) 87–106

[3] Angluin, D.: Negative results for equivalence queries. Machine Learning, **5** (1990) 121–150

[4] Angluin, D., Kharitonov, M.: When won't membership queries help? J. of Comp. and Syst. Sci., **50** (1995) 336–355

[5] Gold, E. M.: Complexity of automaton identification from given data. Info. and Cont., **37** (1978) 302–320

[6] de la Higuera, C.: Characteristic sets for polynomial grammatical inference. Machine Learning, **27** (1997) 125–138

[7] Ishizaka, H.: Polynomial time learnability of simple deterministic languages. Machine Learning, **5** (1990) 151-164

[8] Pitt, L.: Inductive inference, DFAs, and computational complexity. Proc. of AII-89 Workshop on Anal. and Inductive Inference, LNCS **397** (1989) 18–44

[9] Sakakibara, Y.: Learning context-free grammars from structural data in polynomial time. Theor. Comp. Sci., **76** (1990) 223–242

[10] Takada, Y.: Grammatical inference for even-linear languages based on control sets. Info. Proc. Letters, **28** (1988) 193–199

[11] Tajima, Y., Tomita, E., Wakatsuki, M., Terada, M.: Polynomial time learning of simple deterministic languages via queries and a representative sample. Theor. Comp. Sci., **329** (2004) 203–221