JMLR: Workshop and Conference Proceedings 34:182–192, 2014 Proceedings of the 12th ICGI

An example distribution for probabilistic query learning of simple deterministic languages

Yasuhiro Tajima TAJIMA@CSE.OKA-PU.AC.JP
Genichiro Kikui KIKUI@CSE.OKA-PU.AC.JP
Department of Systems Engineering, Okayama Prefectural University

Editors: Alexander Clark, Makoto Kanazawa and Ryo Yoshinaka

Abstract

In this paper, we show a special example distribution on which the learner can guess a correct simple deterministic grammar in polynomial time from membership queries and random examples. At first, we show a learning algorithm of simple deterministic languages from membership and equivalence queries. This algorithm is not a polynomial time algorithm but, assuming a special example distribution, we can modify it to the polynomial time probabilistic learning algorithm.

Keywords: Learning via query, Sample distribution, Simple deterministic language

1. Introduction

Polynomial time query learning of sub-classes of context-free language is difficult. If the equivalence problem is unsolvable then such the language class can not be learnable in polynomial time by (de la Higuera, 1997). This result has been lead from the studies of teachability(Goldman and Mathias, 1996) and its application to grammatical inference.

Simple deterministic languages are subclass of context-free languages but whose equivalence problem can be solvable in polynomial time(Wakatsuki and Tomita, 1991). Its polynomial consists of the size of the grammars and the "thickness" which is the longest length among the set of shortest words generated by every nonterminal in the grammar. This parameter is important for learning algorithm(Tajima et al., 2004) and teachability(Tajima, 2013).

In this paper, we show a special example distribution for polynomial probabilistic learning of simple deterministic languages. On this distribution, simple deterministic languages are polynomial time learning from membership queries and random examples. This distribution is written by restrictions of the grammar which generates the target language. That is appearance probability of every nonterminal is higher than that of any other words with the length more than two. This algorithm made by modifying the algorithm in (Tajima et al., 2004).

2. Preliminaries

A *context-free grammar* (CFG for short) is a 4-tuple $G = (N, \Sigma, P, S)$ where N is a finite set of *nonterminals*, Σ is a finite set of *terminals*, P is a finite set of *rewriting rules* (rules for short) and $S \in N$ is the *start symbol*. Let ε be the word whose length is 0. If there exists

no rule of the form $A \to \varepsilon$ for any $A(\neq S) \in N$, then G is called ε-free. If $G = (N, \Sigma, P, S)$ is ε-free and any rule in P is of the form $A \to a\beta$ then G is said to be in *Greibach normal form*(Harrison, 1978), where $A \in N, a \in \Sigma$ and $\beta \in N^*$. Moreover, a CFG G in Greibach normal form is called in *2-standard form* if every rule $A \to a\beta$ in P satisfies that $|\beta| \leq 2$. In this paper, $|\beta|$ denotes the length of β if β is a string and $|W|$ denotes the cardinality of W if W is a set.

Let $A \to a\beta$ be in P where $A \in N$, $a \in \Sigma$ and $\beta \in N^*$. Let γ and $\gamma' \in (N \cup \Sigma)^*$. Then $\gamma A\gamma' \underset{G}{\Rightarrow} \gamma a\beta\gamma'$ denotes the *derivation* from $\gamma A\gamma'$ to $\gamma a\beta\gamma'$ in G. We define $\overset{*}{\underset{G}{\Rightarrow}}$ to be the reflexive and transitive closure of $\underset{G}{\Rightarrow}$. When it is not necessary to specify the grammar G, $\alpha \Rightarrow \alpha'$ and $\alpha \overset{*}{\Rightarrow} \beta$ stand for $\alpha \underset{G}{\Rightarrow} \alpha'$ and $\alpha \overset{*}{\underset{G}{\Rightarrow}} \beta$, respectively. A *word* generated from $\gamma \in (N \cup \Sigma)^*$ by G is $w \in \Sigma^*$ such that $\gamma \overset{*}{\underset{G}{\Rightarrow}} w$ and 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 by G for the start symbol S is called a word generated by G and the language generated by G is denoted by $L(G) = L_G(S)$. For every CFG G, there exists a CFG G_2 which is in Greibach normal form and in 2-standard form such that $L(G) = L(G_2)$(Harrison, 1978). 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 nonterminal $D \in N$ is said to be *live* if $L_G(D) \neq \emptyset$.

A CFG G is a *simple deterministic grammar* (SDG for short) iff there exists at most one rule which is of the form $A \to a\beta$ for every pair of $A \in N$ and $a \in \Sigma$, i.e. if $A \to a\beta$ is in P then $A \to a\gamma$ is not in P for any $\gamma \in N^*$ such that $\gamma \neq \beta$(Korenjak and Hopcroft, 1966). We note that there exists exactly one derivation for each $w \in L(G)$ in an SDG G. The language generated by an SDG is called a *simple deterministic language* (SDL for short). In addition, such a set P of rules is called *simple deterministic*. Also there exists an SDG G_2 which is in 2-standard form for every SDG G such that $L(G_2) = L(G)$(Harrison, 1978). Thus, we assume that every SDG is in 2-standard form throughout this paper.

For $w \in \Sigma^+$, $proper\_pre(w) = \{w' \in \Sigma^+ \mid w'w'' = w, w'' \in \Sigma^+\}$ is called a set of *proper prefixes* of w. Without loss of generality, we assume an order $\leq_\Sigma$ on Σ and Σ can be denoted by $\Sigma = (a_1, a_2, \cdots a_{|\Sigma|})$ by $\leq_\Sigma$.

We assume that $G_h = (N_h, \Sigma, P_h, S_h)$ denotes a hypothesis SDG which is guessed by the learner, L_t denotes the target language which is an SDL.

A membership query $MEMBER(w)$ for $w \in \Sigma^*$ on an SDL L_t is defined as follows.

$MEMBER(w)$

Input : $w \in \Sigma^*$.

Output : $\begin{cases} 1 & \ldots \text{if } w \in L_t, \\ 0 & \ldots \text{if } w \notin L_t. \end{cases}$

An equivalence query $EQUIV(G_h)$ for an SDL L_t and a hypothesis SDG G_h is defined as follows.

$EQUIV(G_h)$

Input : an SDG G_h.

Output : $\begin{cases} yes & \ldots \text{if } L(G_h) = L_t, \\ no & \text{and } w \in (L_t - L(G_h)) \cup (L(G_h) - L_t) \ldots \text{if } L_t \neq L(G_h). \end{cases}$

Assume an SDG $G_t = (N_t, \Sigma, P_t, S_t)$ such that $L_t = L(G_t)$. Suppose a probability distribution on Σ^* and $Pr(w)$ which is the probability for $w \in \Sigma^*$. We can define the probability for every rule $A \to a\beta$ in P_t as follows;

$$Pr(A \to a\beta) = \sum_{w \in Z(A \to a\beta)} Pr(w)$$

where

$$Z(A \to a\beta) = \{w \in \Sigma^* \mid S_t \stackrel{*}{\Rightarrow} \alpha_1 A \alpha_2 \Rightarrow \alpha_1 a\beta\alpha_2 \stackrel{*}{\Rightarrow} w\}$$

for some $\alpha_1, \alpha_2 \in (N \cup \Sigma)^*$. That is to say, $Pr(A \to a\beta)$ is the appearing probability of $A \to a\beta$ when a sample word is given. In addition, we can define the probability for every $x \in N_t \cup \Sigma$ as follows;

$$Z(x) = \bigcup_{\forall A \in N_t, \forall a \in \Sigma, \forall \beta \in N_t^*} Z(x \to a\beta) \cup Z(A \to x\beta)$$

$$Pr(x) = \sum_{w \in Z(x)} Pr(w).$$

That is to say, $Pr(x)$ is the appearing probability of x in the derivation of a sample word.

Probably approximately correct (PAC for short) is a criterion introduced by (Valiant, 1984). For the target language L_t, the learner guesses a hypothesis G_h which satisfies that

$$Pr(P(L_t \Delta L(G_h)) \leq \varepsilon) \geq 1 - \delta$$

for given $0 < \varepsilon \leq 1$ and $0 < \delta \leq 1$. Here,

$$P(L_t \Delta L(G_h)) = \sum_{w \in (L_t - L(G_h)) \cup (L(G_h) - L_t)} Pr(w).$$

Equivalence queries in an exact learning algorithm can be replaced by polynomial number of random examples.

Theorem 1 ((Angluin, 1987)) *A learning algorithm which uses equivalence queries can be converted to a probabilistic learning algorithm without equivalence queries. In the converted algorithm, n_i random examples are needed instead of i-th equivalence query in the original algorithm where*

$$n_i = \frac{1}{\varepsilon}\left(\ln\frac{1}{\delta} + (\ln 2)(i+1)\right).$$

3. SDL learning algorithm via membership queries and a representative sample

In (Tajima et al., 2004), we have shown the learning algorithm of SDLs via a representative sample and membership queries.

Definition 2 *Let $G = (N, \Sigma, P, S)$ be an SDG such that every $A \in N$ is reachable and live. Let Q be a finite subset of $L(G)$. Then Q is a* representative sample *(RS for short) of G iff the following holds.*

- *For any $A \to a\beta$ in P, there exists a word $w \in Q$ such that $S \stackrel{*}{\Rightarrow} xA\gamma \Rightarrow xa\beta\gamma \stackrel{*}{\Rightarrow} w$ for some $x \in \Sigma^*$ and $\gamma \in N^*$.*

From this definition, for any SDG $G = (N, \Sigma, P, S)$, there exists an RS Q such that $|Q| \leq |P|$.

Definition 3 *For an SDL L, a finite set $Q \subseteq L$ is an RS iff there exists an SDG $G = (N, \Sigma, P, S)$ such that $L(G) = L$ and Q is an RS of G.*

The definition of an RS for an SDL is independent of representation. Though it is possible that Q is an RS for an SDG G_1 and is not that of an SDG G_2 where $L(G_1) = L(G_2) = L_t$, Q is an RS for L_t. It implies that the teacher in this learning setting does not have to suppose a target SDG during the learning process, though we suppose a certain target SDG to construct an RS prior to our learning process. We can find an RS of an SDG $G = (N, \Sigma, P, S)$ in time of a polynomial of $|N|$, $|\Sigma|$ and the thickness k of G.

The following Ishizaka's lemma holds.

Lemma 4 ((Ishizaka, 1990) Lemma 10) *For any nonterminal $A(\neq S_t) \in N_t$ which is reachable and live, and for any $w \in L(G_t)$ such that*

- *$w_p w_m w_s = w$ where $w_p, w_m \in \Sigma^+$, $w_s \in \Sigma^*$, and*
- *$S_t \stackrel{*}{\Rightarrow} w_p \cdot A \cdot w_s \stackrel{*}{\Rightarrow} w$,*

it holds that $u \in L_{G_t}(A)$ for $u \in \Sigma^+$ iff

- *$MEMBER(w_p \cdot u \cdot w'_s) = 1$ and*
- *for any $u' \in proper\_pre(u)$, $MEMBER(w_p \cdot u' \cdot w'_s) = 0$ where w'_s is the shortest suffix of w_s such that $w_p w_m w'_s \in L(G_t)$.*

Assume a derivation $S \underset{G}{\stackrel{*}{\Rightarrow}} pAs \underset{G}{\stackrel{*}{\Rightarrow}} pms$ for $A \in N$, $p, s \in \Sigma^*$ and $m \in \Sigma^+$. Let $r = (p, m, s)$ and $w \in \Sigma^+$. Now, we can define the the function T as

$$T(r, w) = \begin{cases} 1 & (if\ MEMBER(p \cdot w \cdot short(r)) = 1, and \\ & \quad for\ all\ w' \in proper\_pre(w), \\ & \quad MEMBER(p \cdot w' \cdot short(r)) = 0), \\ 0 & (otherwise), \end{cases}$$

here $short(r)$ is the shortest suffix of s such that $MEMBER(p \cdot m \cdot short(r)) = 1$.

Let $Q \subseteq L_t$. The set of nonterminal candidates R is the following set:

$$\begin{aligned} R \quad = \quad & \{(w_p, w_m, w_s) \in \Sigma^+ \times \Sigma^+ \times \Sigma^* \mid w_p w_m w_s \in Q\} \\ & \cup \{(\varepsilon, w, \varepsilon) \mid w \in Q\} \end{aligned}$$

Let $W = \{y \in \Sigma^+ \mid xyz \in Q, x, z \in \Sigma^*\}$. The equivalence relation $\overset{\pi}{=}$ is defined for $r, r' \in R$ as

$$r \overset{\pi}{=} r' \iff T(r, w) = T(r', w)$$

for any $w \in W$. Let $B(r, \pi) = \{r' \in R \mid r' \overset{\pi}{=} r\}$, here π is the partition over R by $\overset{\pi}{=}$.

Consider the following CFG G_{all} with the equivalence relation $\overset{\pi}{=}$.

$$\begin{aligned}
G_{\text{all}} &= (R/\pi, \Sigma, P_{\text{all}}/\pi, S_{\text{all}}) \\
R/\pi &= \{B(r, \pi) \mid r \in R\}, \\
P_{\text{all}}/\pi &= \{B((w_p, a, w_s), \pi) \to a \mid a \in \Sigma, w_p, w_s \in \Sigma^*, (w_p, a, w_s) \in R\} \\
&\quad \cup \{B((w_p, aw_m, w_s), \pi) \to a \cdot B((w_p a, w_m, w_s), \pi) \mid a \in \Sigma, \\
&\qquad w_p, w_s \in \Sigma^*, w_m \in \Sigma^+, (w_p, aw_m, w_s), (w_p a, w_m, w_s) \in R\} \\
&\quad \cup \{B((w_p, aw_{m1}w_{m2}, w_s), \pi) \to \\
&\qquad a \cdot B((w_p a, w_{m1}, w_{m2}w_s), \pi) \cdot B((w_p aw_{m1}, w_{m2}, w_s), \pi) \mid \\
&\qquad a \in \Sigma, w_p, w_s \in \Sigma^*, w_{m1}, w_{m2} \in \Sigma^+, \\
&\qquad (w_p, aw_{m1}w_{m2}, w_s), (w_p a, w_{m1}, w_{m2}w_s), (w_p aw_{m1}, w_{m2}, w_s) \in R\}, \\
S_{\text{all}} &= B((\varepsilon, w, \varepsilon), \pi),
\end{aligned}$$

where $w \in Q$. In our past algorithm(Tajima et al., 2004), deleting rules from P_{all}/π by some conditions.

condition 1 For every rule which is of the form $A \to aB$ in P_{all}/π:

If there exists $w \in W$ such that $aw \in W$ and $T(r_A, aw) \neq T(r_B, w)$ where $r_A \in B(r_A, \pi) = A$ and $r_B \in B(r_B, \pi) = B$, then $A \to aB$ should be removed from P_{all}/π.

This condition means that $A \to aB$ is incorrect when B can generate w but A cannot generate aw or vice versa.

condition 2 For every rule which is of the form $A \to aBC$ in P_{all}/π: If

- there exists $w \in W$ such that $aw \in W$, $T(r_A, aw) = 1$ and

  $$T(r_B, w_1) = 0 \ \textit{or}\ T(r_C, w_2) = 0$$

  for any $w_1, w_2 \in W$ with $w_1 w_2 = w$ here $r_A \in B(r_A, \pi) = A$, $r_B \in B(r_B, \pi) = B$ and $r_C \in B(r_C, \pi) = C$, or

- there exist $w_1, w_2 \in W$ such that $aw_1 w_2 \in W$, $T(r_A, aw_1 w_2) = 0$ and

  $$T(r_B, w_1) = 1 \ \textit{and}\ T(r_C, w_2) = 1$$

then $A \to aBC$ should be removed from P_{all}/π.

This condition means that $A \to aBC$ is incorrect when BC can generate w but A cannot generate aw or vice versa.

Definition 5 *We define the following for every $A \in R/\pi$.*

$$\begin{aligned}
\Sigma_T(A) &= \{a \in \Sigma \mid T(r, a \cdot w) = 1 \textit{ for some } a \cdot w \in W \textit{ here } A = B(r, \pi)\}. \\
\Sigma_P(A) &= \{a \in \Sigma \mid A \to a\beta \textit{ is in } P_{\text{all}}/\pi \textit{ for some } \beta \in (R/\pi)^*\}.
\end{aligned}$$

If $\Sigma_T(A) = \Sigma_P(A)$ then A is called valid. *On the other hand, if $\Sigma_T(A) \neq \Sigma_P(A)$ then A is called* invalid. *If A corresponds to some nonterminal in the SDL which generates the target language, then $\Sigma_P(A) \supseteq \Sigma_T(A)$ because all rules in P_{all}/π are made from positive examples. In addition, any word $a \cdot w$ is called* invalid word *if it causes that A is invalid.*

Suppose that $\gamma = A_1 A_2 \cdots A_n \in (R/\pi)^+$ and $A_i \in R/\pi$ $(i = 1, \ldots, n)$, then γ is valid if $A_i \in R/\pi$ is valid for every $i = 1, \ldots, n$. Conversely, γ is invalid if there exists $A_j \in R/\pi$ which is invalid for some $1 \leq j \leq n$.

Now, we assume that all rules which meet the above conditions have been removed from P_{all}/π. Then,

condition 3 For every rule in P_{all}/π which is of the form $A \to a\beta$ where $\beta \in (R/\pi)^+$:

If β is invalid then $A \to a\beta$ should be removed from P_{all}/π.

This condition means that $A \to a\beta$ is incorrect when β contains $B \in R/\pi$ such that B should generate a word whose first symbol is $b \in \Sigma$ for example, but P_{all}/π contains no rule which is of the form $B \to b\gamma$ for any $\gamma \in (R/\pi)^*$, or vice versa.

Then increasing W by checking SDGs among P_{all}/π, the learner updates G_{all}. Assume an order over all members in P_{all}/π denoted by $\leq_P$, arbitrarily. Let $P(A, a)$ be a rule $A \to a\beta$ in P_{all}/π such that $P(A, a) \leq_P A \to a\gamma$ for any $\gamma \in (R/\pi)^*$. For a rule in P_{all}/π, say $A \to a\beta$, let $G(A \to a\beta) = (R/\pi, \Sigma, P_{A \to a\beta}, S_0)$ be an SDG such that

$$\begin{aligned} P_{A \to a\beta} &= \{P(B, b) \in P_{\mathrm{all}}/\pi \mid B \in R/\pi, b \in \Sigma, B \neq A\} \\ &\quad \cup \{P(B, b) \in P_{\mathrm{all}}/\pi \mid B \in R/\pi, b \in \Sigma, b \neq a\} \\ &\quad \cup \{A \to a\beta\}, \\ S_0 &= B((\varepsilon, w, \varepsilon), \pi), \end{aligned}$$

where $w \in Q$. Then, let $\mathbf{G}$ be the set of SDGs such that

$$\mathbf{G} = \{G(A \to a\beta) \mid A \to a\beta \text{ is in } P_{\mathrm{all}}/\pi\}.$$

If Q is an RS then, comparing SDGs in $\mathbf{G}$, we can obtain the correct hypothesis in polynomial time. We call this algorithm A_{RS}. This is already shown algorithm in (Tajima et al., 2004).

4. Learning via membership queries and counterexamples

If we can obtain an RS by counterexamples then we can construct a query learning algorithm of SDLs via membership and equivalence queries. This is not a polynomial time algorithm. The basic idea of this algorithm is gathering an RS by counterexamples and results of membership queries which concern with Condition 3.

Definition 6 *Let $A \in R/\pi$. $r = (p, m, s) \in A$ is called "shortest" if $|p| \leq |p'|$ for any $r' = (p', m', s') \in A$.*

At first, it calls $EQUIV(\emptyset)$ and obtains a positive counterexample w. Let $Q = \{w\}$, then it simulates the learning algorithm A_{RS} with Q as the input. If A_{RS} makes a hypothesis grammar G_h then $EQUIV(G_h)$ is called. The equivalence query returns "yes" when Q

contains an RS of L_t. If a counterexample is returned then W is increased by it. When repeating A_{RS}, the output of A_{RS} is not different to the previous execution, Q is increased by invalid words. For every $A \in R/\pi$ and $a \in \Sigma$ such that A has an invalid word aw, $p \cdot aw \cdot short(r)$ is added to Q where $r = (p, m, s) \in A$ is the shortest. It causes increasing of R. Repeating these process, Q becomes an RS and A_{RS} outputs a correct hypothesis. We call this algorithm A_2. Algorithm1 is the learning algorithm A_2. This algorithm is made by adding a hypothesis check function to A_{RS}.

Algorithm 1: SDL learning via membership and equivalence queries A_2

Output: a hypothesis SDG G_h
$P_0 := \emptyset, Q := \emptyset$;
$G_h := \emptyset$;
if $EQUIV(G_h) = yes$ **then**
  output G_h and terminate;
else
  let the positive counterexample be w;
  $W := \{y \in \Sigma^+ \mid x, z \in \Sigma^*, x \cdot y \cdot z = w\}$;
  $Q := Q \cup \{w\}$;
end
repeat
  $W := W \cup \{y \in \Sigma^+ \mid x, z \in \Sigma^*, x \cdot y \cdot z \in Q\}$;
  $R := \{(x, y, z) \mid z \in \Sigma^*, x, y \in \Sigma^+, x \cdot y \cdot z \in Q\} \cup \{(\varepsilon, w, \varepsilon) \mid w \in Q\}$;
  $\mathbf{G} := \emptyset$;
  repeat
    find $T(r, w)$ for all $r \in R$ and $w \in W$ and define the equivalence relation $\overset{\pi}{=}$;
    delete incorrect rules from P_{all}/π by **Conditions 1, 2 and 3**;
    find $P(A, a)$ for all $A \in R/\pi$ and $a \in \Sigma$;
    make the set $\mathbf{G}$ of SDLs;
    $W' := \emptyset$;
    for *every pair of* $G_1, G_2 \in \mathbf{G}$ *and every* $A \in R/\pi$ **do**
      find a word $w \in (L_{G_1}(A) - L_{G_2}(A)) \cup (L_{G_2}(A) - L_{G_1}(A))$ such that $|w|$ is bounded by a polynomial;
      $W' := W' \cup \{w\}$;
    end
    for *all* $w \in W'$ **do**
      $W := W \cup \{y \in \Sigma^+ \mid x, z \in \Sigma^*, x \cdot y \cdot z = w\}$;
    end
  until $W' = \emptyset$;
  call [hypothesis check];
  $P_0 := P_{\text{all}}/\pi$;
until *forever*;

Algorithm 2: [hypothesis check]

if $P_{\mathrm{all}}/\pi = P_0$ **then**
  for *every* $A \in R/\pi$ **do**
    if *there exists an invalid word* $aw \in W$ *such that* $a \leq_\Sigma b$ *for any invalid word* bu *of* A **then**
      let $r = (p, m, s) \in A$ be the shortest;
      $Q := Q \cup \{p \cdot aw \cdot short(r)\}$;
      $W := W \cup \{w' | x, y \in \Sigma^*, xw'y = p \cdot aw \cdot short(r)\}$;
    end
  end
else
  select $G_h \in \mathbf{G}$, arbitrarily;
  if $EQUIV(G_h) = yes$ **then**
    output G_h and terminate;
  end
  let the counterexample be w;
  $W := W \cup \{y \in \Sigma^+ \mid x, z \in \Sigma^*, x \cdot y \cdot z = w\}$;
end

Definition 7 *Let* $G = (N, \Sigma, P, S)$ *be an SDG and* $Q \subseteq L(G)$. *We define*

$$U_G(Q) = \{A \to \beta \in P | w \in Q, \alpha, \gamma \in (N \cup \Sigma)^*, S \overset{*}{\underset{G}{\Rightarrow}} \alpha A \gamma \underset{G}{\Rightarrow} \alpha\beta\gamma \overset{*}{\underset{G}{\Rightarrow}} w\}$$

i.e. the set of rules to derive Q.

Now, the following lemma holds.

Lemma 8 *Let* Q_1 *be the* Q *in* A_2 *when "hypothesis check" is called. Let* Q_2 *be also the* Q *when the next time "hypothesis check" is called after* Q_1 *is updated. It holds that* $U_{G_t}(Q_2) \supset U_{G_t}(Q_1)$.

Proof *Suppose that* $U_{G_t}(Q_1) = U_{G_t}(Q_2)$. *Then* Q_2 *can be derived by rules in* $U_{G_t}(Q_1)$. *From the assumption,* $Q_2 \supset Q_1$. *In addition* $P_{\mathrm{all}}/\pi = P_0$ *holds. It implies that the counterexample given when* A_w *holds* Q_1 *makes no change to* P_{all}/π. *Thus,* G_{all} *can not generate the counterexample. It implies that* $U_{G_t}(Q_2) \supset U_{G_t}(Q_1)$. *This is contradiction.* ■

Time complexity of A_2 is as follows.

Lemma 9 *The size of* $|R|$ *in* A_2 *is bounded by* $O(l^4 2^{2|N_t||\Sigma|})$.

Proof *The time complexity of Algorithm* A_2 *except for calling "hypothesis check" is bounded by a polynomial of* $|N_t|, |\Sigma|$ *and the length of the longest counterexample* l *because of the complexity of* A_{RS} *in (Tajima et al., 2004).*

When "hypothesis check" is called, there are two cases : $P_{\mathrm{all}}/\pi = P_0$ and $P_{\mathrm{all}}/\pi \neq P_0$. $P_{\mathrm{all}}/\pi = P_0$ occurs at most $|N_t||\Sigma|$ times because of Lemma 8. In addition, $P_{\mathrm{all}}/\pi \neq P_0$ occurs at most $|R||\Sigma|$ times because every counterexample which satisfies $P_{\mathrm{all}}/\pi \neq P_0$ makes R/π finer or some rules in P_{all}/π are deleted.

Now, we evaluate $|R|$. $|R|$ is $O(|Q|k^2)$ where k is the length of the longest word in Q. At the beginning of A_2, k is bounded by the length of the longest counterexample l. Also the length of the longest word in Q is bounded by l. The length of the longest word in Q is doubled when Q is updated. Thus, $k = O(l2^{|N_t||\Sigma|})$ and $|R| = O(|Q|l^2 2^{2|N_t||\Sigma|})$.

At the beginning of A_2, $|Q|$ is bounded by $O(l^2)$. We can conclude that $|R| = O(l^4 2^{2|N_t||\Sigma|})$.

■

This size is not a polynomial. Also, the time complexity of A_2 is not bounded by a polynomial. The enumeration of all SDGs with the size of n takes $O(|N_t|^{|N_t||\Sigma|})$ time. We can say that A_2 is an improved algorithm comparing to the enumerating algorithm.

5. An example distribution to probabilistic learning

Suppose an example distribution such that $Pr(A) > Pr(B_1)Pr(B_2) + \rho$ satisfies for any $A \in N_t$ and for any $B_1, B_2 \in N_t \cup \Sigma$. Here, ρ is a fixed value such that $0 < \rho$ and $Pr(A) >> \rho$ for any $A \in N_t$. From Angluin's result(Angluin, 1987), i-th equivalence queries can be replaced by checking consistency of

$$n_i = \frac{1}{\varepsilon}\left(\ln\frac{1}{\delta} + (\ln 2)(i+1)\right)$$

examples. When the learner can use membership queries and random examples drawn by the distribution, We can modify the "hypothesis check" procedure to A_p in Algorithm3.

This procedure is replaced calling $EQUIV()$ by random examples. If there needs to expand Q by invalid words then select the most frequent appearing nonterminal A and whose invalid word is added to Q.

Now, we evaluate the sample complexity of A_p. If the most appearing $B \in R/\pi$ does not correspond to any $A \in N_t$ then B must corresponds to $\beta \in (N_t \cup \Sigma)^+$ such that $|\beta| \geq 2$. It implies that the expectation $Pr(\beta) < Pr(A)$ for some $A \in N_t$ from the assumption. In addition, $Pr(A) - Pr(\beta) > \rho$ holds. Let X_A be a random variable whether A is used to derive an example word. Also, let X_β be that of β to the derivation. From Chernoff-Hoeffding bound, it holds that

$$Pr(X_A \geq \mu_A + n\frac{\rho}{2}) \leq \exp(\frac{-2n^2\rho^2\frac{1}{4}}{n})$$

and

$$Pr(X_\beta \leq \mu_\beta - n\frac{\rho}{2}) \leq \exp(\frac{-2n^2\rho^2\frac{1}{4}}{n})$$

for n examples. Here, $\mu_A = n \cdot Pr(A)$, $\mu_\beta = n \cdot Pr(\beta)$.

Algorithm 3: probabilistic hypothesis check : A_p

if $P_{\mathrm{all}}/\pi = P_0$ **then**
  take n_0 examples;
  select the most frequent appearing $A \in R/\pi$ which has an invalid word $aw \in W$;
  let $r = (p, m, s) \in A$ be the shortest;
  $Q := Q \cup \{p \cdot aw \cdot short(r)\}$;
  $W := W \cup \{w' | x, y \in \Sigma^*, xw'y = p \cdot aw \cdot short(r)\}$;
else
  select $G_h \in \mathbf{G}$, arbitrarily;
  let i be the number of counts to reach here;
  take $\frac{1}{\varepsilon}\left(\ln\frac{1}{\delta} + (\ln 2)(i+1)\right)$ examples;
  if *all examples are consistent* **then**
    output G_h and terminate;
  else
    for *every example word* w *which is not consistent with* $L(G_h)$ **do**
      $W := W \cup \{y \in \Sigma^+ \mid x, z \in \Sigma^*, x \cdot y \cdot z = w\}$;
    end
  end
end

If $Pr(X_A \geq \mu_A + n\frac{\rho}{2}) \cdot Pr(X_\beta \leq \mu_\beta - n\frac{\rho}{2}) \leq \delta$ holds then Q is increased by an invalid word of $A \in N_t$. Now,

$$\begin{aligned} Pr(X_A \geq \mu_A + n\frac{\rho}{2}) \cdot Pr(X_\beta \leq \mu_\beta - n\frac{\rho}{2}) &\leq \exp(\frac{-4n^2\rho^2\frac{1}{4}}{n}) \\ &= \exp(-n\rho^2) \end{aligned}$$

Thus,

$$\begin{aligned} \exp(-n\rho^2) &\leq \delta \\ -n\rho^2 &\leq \ln(\delta) \\ n &\geq \frac{\ln(\delta)}{-\rho^2} \\ &= \frac{\ln(\frac{1}{\delta})}{\rho^2} \end{aligned}$$

If $\rho^2 > \varepsilon$ then this value is less than

$$n_i = \frac{1}{\varepsilon}\left(\ln\frac{1}{\delta} + (\ln 2)(i+1)\right)$$

which is examples replaced by i-th equivalence query. Thus, the algorithm A_p needs polynomial size examples to terminate when $\rho^2 > \varepsilon$.

6. Conclusion

We show a special example distribution which can be represented by differences of nonterminal appearance probability. On this distribution, SDLs can be learnable in polynomial time with membership queries and random examples. The probabilistic learning algorithm shown in this paper is collecting a representative sample using our algorithm in (Tajima et al., 2004). For the future works, it is needed that finding more relaxed distributions and conditions to polynomial time learning.

References

D. Angluin. Learning regular languages from queries and counterexamples. *Inf. & Comp.*, 75:87–106, 1987.

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

S. A. Goldman and H. D. Mathias. Teaching a smart learner. *Journal of Comp. and Sys. Sci.*, 52:255–267, 1996.

M. A. Harrison. *Introduction to Formal Language Theory.* Addison-Wesley, 1978.

H. Ishizaka. Polynomial time learnability of simple deterministic languages. *Machine Learning*, 5:151–164, 1990.

A. J. Korenjak and J. E. Hopcroft. Simple deterministic languages. In *Proc. IEEE 7th Annu. Symp. on Switching and Automata Theory*, pages 36–46, 1966.

Y. Tajima. Teachability of a subclass of simple deterministic languages. *IEICE Trans. on Info. & Sys.*, E96-D:2733–2742, 2013.

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

L. G. Valiant. A theory of the learnable. *Comm. of the ACM*, 27:1134–1142, 1984.

M. Wakatsuki and E. Tomita. An improved branching algorithm for checking the equivalence of simple dpda's and its worst-case time complexity (in japanese). *Trans. of IEICE*, J74-D-I:595–603, 1991.