Identification of Substitutable Context-Free Languages over Infinite Alphabets from Positive Data

Yutaro Numaya YUTARO.NUMAYA.Q3@DC.TOHOKU.AC.JP
Diptarama Hendrian DIPTARAMA@TOHOKU.AC.JP
Ryo Yoshinaka RYOSHINAKA@TOHOKU.AC.JP
Ayumi Shinohara AYUMIS@TOHOKU.AC.JP
Tohoku University, Sendai, Japan

Editors: François Coste, Faissal Ouardi and Guillaume Rabusseau

Abstract

This paper is concerned with the identification in the limit from positive data of substitutable context-free languages CFLs) over infinite alphabets. Clark and Eyraud (2007) showed that substitutable CFLs over finite alphabets are learnable in this learning paradigm. We show that substitutable CFLs generated by grammars whose production rules may have *predicates* that represent sets of potentially infinitely many terminal symbols in a compact manner are learnable if the terminal symbol sets represented by those predicates are learnable, under a certain condition. This can be seen as a result parallel to Argyros and D'Antoni's work (2018) that amplifies the query learnability of predicate classes to that of symbolic automata classes. Our result is the first that shows such amplification is possible for identifying some CFLs in the limit from positive data.

Keywords: Identification in the limit; Substitutable; Context-free languages; Infinite alphabets;

1. Introduction

Recently, several well-established grammatical inference approaches for learning formal languages over finite alphabets have been extended for targeting languages over infinite alphabets. One of the most popular extensions of finite-state automata for working over infinite alphabets is *symbolic finite state automata (*SFA*)*. They have *predicates* as edge labels instead of input symbols, which represent potentially infinite sets of input symbols in a compact manner. While Fisman et al. (2022) have studied the efficient learnability of SFAs under the *identification in the limit* paradigm, most preceding studies on the learning of SFAs are concerned with *query learning*: more specifically, Angluin's minimally adequate teacher model (1987). Those works include the ones by Mens and Maler (2015); Argyros et al. (2016); Drews and D'Antoni (2017); Maler and Mens (2017); Argyros and D'Antoni (2018). Among those, Argyros and D'Antoni's algorithm is quite generic. It learns deterministic SFAs over any predicates when an efficient learning algorithm for the Boolean closure of the predicates is available. For example, as there exist learning algorithms for binary decision diagrams (BDDs) (Nakamura, 2005) and zero-suppressed binary decision diagrams (ZDDs) (Mizumoto et al., 2017), SFAs whose transitions carry BDDs/ZDDs can be learned by Argyros and D'Antoni's algorithm. Their algorithm creates predicate learner instances for transition edges and lets them learn the respective labels. Those predicate learners make queries and the main algorithm answers them appropriately. The communication between

the main algorithm and those predicate learners is the sophistication of their technique. Their technique has been extended to nondeterministic automata (Chubachi et al., 2019) and weighted automata (Suzuki et al., 2021) as well.

This paper attempts to investigate how learnability of predicates can be amplified for learning formal languages over predicates, in a setting different from Argyros and D'Antoni's work. More specifically, we are concerned with the *identification in the limit of substitutable context-free languages (CFLs) from positive data*. Our algorithm extends the one proposed by Clark and Eyraud (2007) for identifying substitutable CFLs from positive data. We employ a predicate learner instance to obtain a predicate where Clark and Eyraud's algorithm puts a terminal rule in a hypothesis context-free grammar (CFG), by feeding the predicate learner with appropriate characters extracted from positive example strings. Different extensions of CFGs and CFLs to those over infinite alphabets have been discussed for theoretical interests (e.g., Autebert et al., 1980; Otto, 1985; Cheng and Kaminski, 1998) and have appeared in application demands, such as model checking (e.g., Maurer, 1990; Majumdar and Xu, 2007; Godefroid et al., 2008; Guo and Subramaniam, 2014). The extension we will use is the simplest one among those: we extend CFGs by replacing terminal symbols in CFGs with predicates. Still, as far as we know, our result is the first that demonstrates Argyros and D'Antoni's idea can be applied to learning CFLs in the learning paradigm of the identification in the limit from positive data.

2. Preliminaries

2.1. Identification in the limit from positive data

We first review the identification in the limit paradigm (Gold, 1967). This paper discusses learning from positive data only. For a countably infinite set D, called the *domain*, a *concept* C is any subset of D and elements of C are *(positive) examples* of C. Let $\mathcal{R}$ be any recursive set of finite descriptions and f be a function mapping descriptions to concepts. An infinite sequence of elements of D is called a *(positive) presentation of a concept* C if and only if all and only examples of C appear in the sequence. A *learner* A on $\mathcal{R}$ is an infinite procedure that takes elements of D one by one and computes a description in $\mathcal{R}$ each time. We say that A converges to $R \in \mathcal{R}$ on a positive presentation $\sigma = \langle c_1, c_2, \ldots \rangle$ if there is $n \in \mathbb{N}$ such that A outputs only R after taking $c_1, \ldots, c_n$. The learner A *identifies a concept* C *in the limit using* $\mathcal{R}$ if for any presentation σ of C, there is $R \in \mathcal{R}$ such that $f(R) = C$ and A converges to R on σ. Moreover, A *identifies a class* $\mathcal{C}$ *of concepts in the limit using* $\mathcal{R}$ if A identifies every concept from $\mathcal{C}$ in the limit using $\mathcal{R}$. We sometimes say that we identify $\mathcal{R}$ when $\mathcal{C} = \{ f(R) \mid R \in \mathcal{R} \}$.

Example 1 *Let the domain be the set* $\mathbb{N}$ *of natural numbers and the concept class be* $\mathcal{C} = \{ C_{n,m} \mid 0 \le m < n \}$ *where* $C_{n,m} = \{ kn + m \in \mathbb{N} \mid k \in \mathbb{N} \}$. *We let* $\psi_{n,m}$ *represent the concept* $C_{n,m} = [\![\psi_{n,m}]\!]$ *for* $n, m \in \mathbb{N}$ *with the denotation funcion* $[\![\cdot]\!]$ *and define* $\Psi_0 = \{ \psi_{n,m} \mid 0 \le m < n \}$. *Then, a procedure that hypothesizes* $\psi_{n,m}$ *for the greatest common divisor* n *of differences of given examples and the remainder* m *of the smallest example divided by* n *identifies* $\mathcal{C}$ *in the limit.*

If a concept class $\mathcal{C}$ contains all finite concepts and at least one infinite concept over D, then $\mathcal{C}$ is called *superfinite*. Gold (1967) showed the following negative result.

Theorem 1 (Gold 1967) *Any superfinite class is not identifiable in the limit from positive data.*

2.2. Languages and grammars

Let Σ be an *alphabet*, nonempty finite or countably infinite set of characters. Σ^* denotes the set of strings over Σ. Note that Σ^* is countably infinite regardless of whether Σ is finite or countably infinite. By $|u|$ we denote the *length* of $u \in \Sigma^*$. We denote the empty string by ε, for which $|\varepsilon| = 0$. We write $\Sigma^+ = \Sigma^* - \{\varepsilon\}$. Any element of $\Sigma^* \times \Sigma^*$ is called a *context*. The length of a context $\langle u_1, u_2\rangle$ is defined to be $|\langle u_1, u_2\rangle| = |u_1| + |u_2|$. For a string $v \in \Sigma^*$ and a context $\langle u_1, u_2\rangle \in \Sigma^* \times \Sigma^*$, the *composition* of them is $\langle u_1, u_2\rangle \odot v = u_1 v u_2 \in \Sigma^*$. We assume an arbitrary but fixed well-order over Σ, so that every subset of Σ has a least element. The well-order is referred to as the *lexicographic order*. We naturally extend the lexicographic order $<$ over Σ to the *length-lexicographic order* over Σ^*. We write $u_1 < u_2$ for $u_1, u_2 \in \Sigma^*$ if (1) $|u_1| < |u_2|$, or (2) $|u_1| = |u_2|$ and there are $v_0, v_1, v_2 \in \Sigma^*$ and $a, b \in \Sigma$ such that $u_1 = v_0 a v_1$, $u_2 = v_0 b v_2$, and $a < b$. It is also extended for contexts so that $\langle u_1, u_2\rangle < \langle v_1, v_2\rangle$ if $u_1 \# u_2 < v_1 \# v_2$ for a fresh symbol $\# \notin \Sigma$ which is assumed to be smaller than any elements of Σ. The cardinality of a set X is denoted by $|X|$. Moreover, if X is a set of contexts or strings, then its size is defined to be $\|X\| = \sum_{x \in X} |x|$.

We extract substrings from a language $L \subseteq \Sigma^*$ as

$$\mathrm{Sub}(L) = \{\, x \in \Sigma^+ \mid uxv \in L \text{ for some } \langle u, v\rangle \in \Sigma^* \times \Sigma^* \,\}.$$

Note that $\mathrm{Sub}(L)$ consists of nonempty strings only.

We write $x \sim_L y$ for $x, y \in \Sigma^+$ and $L \subseteq \Sigma^+$ if there are $\langle u, v\rangle \in \Sigma^* \times \Sigma^*$ such that $uxv \in L$ and $uyv \in L$. We write $x \equiv_L y$ when $uxv \in L \iff uyv \in L$ for all $\langle u, v\rangle \in \Sigma^* \times \Sigma^*$.

Definition 2 (Clark and Eyraud, 2007) *A language L is* substitutable *if $x \sim_L y$ implies $x \equiv_L y$ for any $x, y \in \mathrm{Sub}(L)$.*

The substitutability can be rephrased as

$$uxv, uyv, u'xv' \in L \implies u'yv' \in L \text{ for all } u, v, u', v' \in \Sigma^* \text{ and } x, y \in \Sigma^+.$$

Let us denote the transitive closure of $\sim_L$ by $\approx_L$ and the equivalence class of x modulo $\approx_L$ by $[x]_L$. If L is substitutable, $\sim_L$, $\approx_L$, and $\equiv_L$ all coincide over $\mathrm{Sub}(L)$. In this case, $[x]_L$ is called a *congruence* class.

A *context-free grammar (*CFG*)* is denoted by a quadruple $G = \langle \Sigma, V, P, S\rangle$, where Σ is the finite set of *terminal symbols*, V, disjoint from Σ, is the finite set of *nonterminal symbols*, P is the finite set of *production rules* and $S \in V$ is the *start symbol*. A production rule in P has the form $A \to \beta$ for some $A \in V$ and $\beta \in (\Sigma \cup V)^+$. If $A \to \beta \in P$, we write $\alpha A \gamma \Rightarrow_G \alpha\beta\gamma$ for any $\alpha, \gamma \in (\Sigma \cup V)^*$. The reflexive and transitive closure of $\Rightarrow_G$ is $\Rightarrow_G^*$. The subscript G of $\Rightarrow_G$ is omitted if it is understood from the context. The *context-free language* (CFL) $\mathrm{L}(G)$ generated by G is the set $\mathrm{L}(G, S)$, where $\mathrm{L}(G, \alpha) = \{\, w \in \Sigma^* \mid \alpha \overset{*}{\Rightarrow} w \,\}$ for $\alpha \in (\Sigma \cup V)^*$. The description size of G is defined as $\|G\| = \sum_{A \to \beta \in P}(|A\beta|)$. A nonterminal symbol $A \in V$ is *useless* in G if there are no $x, y, z \in \Sigma^*$ such that $S \overset{*}{\Rightarrow} xAz \overset{*}{\Rightarrow} xyz$. Note that we do not

allow empty right hand side to production rules, and thus any CFLs dealt with in this paper are ε-free.

We extend the definition of CFGs by allowing production rules to have concise representations of sets of terminal symbols. Hereafter, we allow Σ to be a countably infinite or finite set. Let $\mathcal{C}$ be a class of (possibly infinite) subsets of Σ that is equipped with finite representations for those subsets. That is, we have a set Ψ of *predicates* and its semantic function $[\![\cdot]\!]$ such that every set $C \in \mathcal{C}$ has some predicate $\varphi \in \Psi$ with $[\![\varphi]\!] = C$. This paper assumes it is decidable whether $a \in [\![\varphi]\!]$ for any $a \in \Sigma$ and $\varphi \in \Psi$. A Ψ-CFG is a quadruple $G = \langle \Psi, V, P, S \rangle$, which is defined in the same way as a CFG except that $P \subseteq V \times (V \cup \Psi)^+$. Those predicates φ shall be rewritten to any of the elements of $[\![\varphi]\!]$, i.e., we have $\varphi \Rightarrow_G a$ iff $a \in [\![\varphi]\!]$. This allows us to handle languages over infinite alphabets. The language generated by a Ψ-CFG is called a Ψ-CFL. We disallow useless nonterminals. Hence, we assume that Ψ has no predicate that represents the empty set. A Ψ-CFG is said to be in the *branching normal form* if $P \subseteq V \times (VV \cup \Psi)$. Obviously every Ψ-CFL admits a grammar in the branching normal form.

Example 2 *Let Ψ_0 be the predicate class discussed in Example 1 and consider the Ψ_0-CFG whose production rules are*

$$S \to \psi_{4,3} S \psi_{8,1} \mid \psi_{2,0}\,.$$

This generates a substitutable language

$$L = \{\, a_1 \dots a_n b c_1 \dots c_n \mid a_1, \dots, a_n \in C_{4,3},\ b \in C_{2,0},\ c_1, \dots, c_n \in C_{8,1},\ n \geq 0 \,\},$$

which includes, say, 4, 3 8 9 *and* 43 27 0 41 9.

A (conventional) CFG over a finite alphabet Σ can be seen as a special case, where each terminal symbol $a \in \Sigma$ has a unique predicate $a \in \Psi = \Sigma$ that represents $[\![a]\!] = \{a\}$. So we call a CFG a Σ-CFG.

The following lemma is easy.

Lemma 3 *Suppose a Ψ-CFG G generates a substitutable language L. Then, for every nonterminal N, $\mathrm{L}(G, N) \subseteq [x]_L$ for any $x \in \mathrm{L}(G, N)$.*

Proof There is a derivation $S \Rightarrow_G^* uNv$ and thus every member of $\mathrm{L}(G, N)$ shares the context $\langle u, v \rangle \in \Sigma^* \times \Sigma^*$. By the substitutability assumption, the conclusion holds. ■

Therefore, if a Ψ-CFG G that generates a substitutable language has two nonterminals whose languages are subsets of the same congruence class, those nonterminals can be merged without changing the language of the grammar.

2.3. Our learning target

Argyros and D'Antoni (2018) have established a technique to obtain query learnable classes of symbolic automata from learnable classes of predicates. This paper tackles the problem of obtaining a parallel result on substitutable Ψ-CFLs. Clark and Eyraud (2007) have shown the following theorem, for finite alphabets Σ.

Theorem 4 (Clark and Eyraud, 2007) *Substitutable* Σ-CFL*s are identifiable in the limit from positive data.*

We first claim the following negative result.

Proposition 5 *There is a predicate class* Ψ *which is identifiable in the limit from positive data, but the class of substitutable* Ψ-CFL*s is not.*

Proof Consider $\Sigma = \mathbb{N}$ and $\Psi = \mathbb{N} \cup \{*\}$ where $[\![n]\!] = \{n\}$ and $[\![*]\!] = \mathbb{N}$. Clearly Ψ is identifiable in the limit from positive data. If every given positive example is the same number n, one conjectures n as its hypothesis. Otherwise, one should conjecture $*$. On the other hand, for any finite subset $X \subsetneq \mathbb{N}$ of natural numbers, we can have a Ψ-CFG G whose language is $\mathrm{L}(G) = X$. The rules are just $S \to n$ for all and only $n \in X$. Note that every subset of $\mathbb{N}$ is substitutable. Another Ψ-CFG can generate $\mathbb{N}$ using $S \to *$. That is, all nonempty finite subsets of $\mathbb{N}$ and $\mathbb{N}$ itself are substitutable Ψ-CFLs. The class of substitutable Ψ-CFLs is a superclass of a superfinite class (over $\mathbb{N}$ rather than $\mathbb{N}^*$) and thus is not identifiable in the limit from positive data by Theorem 1. ■

We note that the technique by Argyros and D'Antoni (2018) applies only to symbolic automata over *Boolean algebras*, where the predicate class must be closed under union, intersection, and complement. So far we have not assumed any closure property on the predicate class Ψ in concern. Indeed, one can show that if Ψ is closed under union and is identifiable in the limit from positive data, then the class of substitutable Ψ-CFLs is also identifiable in the limit from positive data, using our main result (Theorem 12) and Proposition 8.

Corollary 6 *Suppose that* Ψ *is closed under union and is identifiable in the limit from positive data. Then, substitutable* Ψ-CFL*s are identifiable in the limit from positive data.*

This might appear as a result analogous to Argyros and D'Antoni. However, requiring Ψ to be closed under union looks a little too strong, particularly for the learning paradigm of the identification in the limit from positive data. The predicate class Ψ_0 in Examples 1 and 2 is not closed under union, and one can show that its union closure is not identifiable in the limit from positive data (See Appendix A). Many more predicate classes which are interesting from the learnability point of view are not closed under union. Instead, we target the following broader class Ψ-$\mathcal{SCFL}$ of substitutable Ψ-CFLs.

Definition 7 *Let* Ψ *be a set of predicates which is identifiable in the limit from positive data. By* Ψ-$\mathcal{SCFL}$*, we denote the class of substitutable* Ψ-CFL*s* L *such that every character* $a \in \Sigma \cap \mathrm{Sub}(L)$ *admits some predicate* $\varphi \in \Psi$ *satisfying* $[a]_L \cap \Sigma = [\![\varphi]\!]$.

Proposition 8 *If* Ψ *is identifiable in the limit and closed under union, then every substitutable* Ψ-CFL *belongs to* Ψ-$\mathcal{SCFL}$.

Proof Let $G = \langle \Psi, V, P, S \rangle$ be a Ψ-CFG generating a substitutable language L. For $a \in \mathrm{Sub}(L) \cap \Sigma$, let

$$\Psi_a = \{\, \varphi \mid N \to \varphi \in P \text{ for some } N \text{ and } [\![\varphi]\!] \subseteq [a]_L \,\}.$$

Since Ψ is closed under union, there is φ_a such that $[\![\varphi_a]\!] = \bigcup_{\varphi \in \Psi_a} [\![\varphi]\!] = [a]_L \cap \Sigma$. ■

Lemma 9 *For every $L \in \Psi\text{-}\mathcal{SCFL}$, there is a grammar G such that every predicate φ in G satisfies $[\![\varphi]\!] = [a]_L \cap \Sigma$ for all $a \in [\![\varphi]\!]$.*

Proof Every predicate φ in a Ψ-CFG generating L can be replaced by φ' such that $[\![\varphi']\!] = [a]_L \cap \Sigma$ if $[\![\varphi]\!] \subseteq [a]_L$ for some $a \in \Sigma$. The definition of $\Psi\text{-}\mathcal{SCFL}$ ensures that such φ' can be found in Ψ. ■

3. Learner

This section presents a learner for $\Psi\text{-}\mathcal{SCFL}$ using Ψ-CFGs. Let Λ be a predicate learner for Ψ. That is, against an infinite sequence $\langle a_1, a_2, \dots \rangle$ of characters of Σ, it hypothesizes an infinite sequence of predicates in Ψ which finally converges to a predicate representing $\{a_1, a_2, \dots\}$. We embed instances of predicate learner Λ into the language learner for substitutable CFLs by Clark and Eyraud (2007). Let us briefly review Clark and Eyraud's hypothesis construction in a simplified form (Yoshinaka, 2008). When learning a substitutable Σ-CFL, given a positive example set W, the algorithm constructs nonterminals from all the nonempty substrings of W. We denote the nonterminal symbol indexed by a substring u of a positive example by $\langle\!\langle u \rangle\!\rangle$ (Here, $\langle\!\langle \cdot \rangle\!\rangle$ is not an operator, but it denotes just a nonterminal symbol). We would like $\langle\!\langle u \rangle\!\rangle$ to generate $[u]_{L_*}$ for the learning target L_*. To this end, we first create trivial branching rules $\langle\!\langle uv \rangle\!\rangle \to \langle\!\langle u \rangle\!\rangle \langle\!\langle v \rangle\!\rangle$ and terminal rules $\langle\!\langle a \rangle\!\rangle \to a$ for $a \in \Sigma$, by which we obtain $\langle\!\langle u \rangle\!\rangle \Rightarrow^+ u$ for all nonterminals $\langle\!\langle u \rangle\!\rangle$. In addition, in accordance with the substitutability of the learning target, we add unary rules $\langle\!\langle u \rangle\!\rangle \to \langle\!\langle v \rangle\!\rangle$ iff $u \approx_W v$. The start symbol is $\langle\!\langle w \rangle\!\rangle$ for an arbitrary $w \in W$. Then, after sufficiently many positive examples are collected, the grammar will eventually generate the target language. If we update the hypothesis grammar always, the hypotheses will never converge unless the target language is finite. We reconstruct the grammar only when we find an example which the current hypothesis cannot generate. If the newly given example is generated by the current hypothesis, we do not change the hypothesis.

When Σ is infinite, we cannot have rules of the form $\langle\!\langle a \rangle\!\rangle \to a$ for all terminal symbols a that appear in some positive examples. Instead, we let the Ψ-learner compute an appropriate predicate that might represent $[a]_{L_*} \cap \Sigma$. To this end, we create a copy instance Λ_a of the Ψ-learner Λ for each observed terminal symbol $a \in \Sigma$, and give it the terminal symbols that share some context with a. We use the predicate φ output by Λ_a in a terminal rule as $\langle\!\langle a \rangle\!\rangle \to \varphi$. Under the assumption that Λ identifies Ψ in the limit from positive data, Λ_a will converge to a predicate $\varphi_a \in \Psi$ such that $[\![\varphi_a]\!] = [a]_{L_*} \cap \Sigma$. Indeed, the definition of $\Psi\text{-}\mathcal{SCFL}$ guarantees that such a predicate can be found in Ψ. Other part of the grammar construction (branching and unary rules) will remain the same as the case where Σ is finite.

However, to guarantee that the hypotheses will converge to a correct grammar, a little care is required on the rule update procedure. Whereas the hypothesis grammar is kept unchanged when no counterexample is given in Clark and Eyraud's algorithm, our algorithm keeps giving examples to the predicate learner instances and may update terminal rules using the output predicate. This is to ensure that all created predicate learner instances will converge to a correct predicate.

The pseudocode of our learner is shown in Algorithm 1. The variable $\Sigma_\Lambda \subseteq \Sigma$ maintains predicate learner instances we have created so far. Note that we never "reset" those

instances once they have been created. They have their own memory, and in each iteration of the main **for** loop, Λ_a is fed with a and other characters known to belong to $[a]_{L_*}$. It outputs as many predicates as the input characters, among which we take only the last one and discard the others. We say the output φ by Λ_a is *correct* if $[\![\varphi]\!] = [a]_{L_*} \cap \Sigma$. Similarly, a rule of the form $\langle\!\langle a \rangle\!\rangle \to \varphi$ of our hypothesis grammar is *correct* if $[\![\varphi]\!] = [a]_{L_*} \cap \Sigma$.

Algorithm 1: Learning Ψ-$\mathcal{SCFL}$

Input: positive presentation $\langle w_1, w_2, \ldots \rangle$
Output: sequence of hypothesis Ψ-CFGs $\langle G_1, G_2, \ldots \rangle$
$\Sigma_\Lambda \coloneqq \varnothing$;
for $t \coloneqq 1, 2, \ldots$ **do**
  $W \coloneqq \{w_1, \ldots, w_t\}$;
  if $t = 1$ *or* $W \notin \mathrm{L}(G_{t-1})$ **then**
    $V \coloneqq \{ \langle\!\langle u \rangle\!\rangle \mid u \in \mathrm{Sub}(W) \}$;
    $P_1 \coloneqq \{ \langle\!\langle u \rangle\!\rangle \to \langle\!\langle v \rangle\!\rangle \mid \langle\!\langle u \rangle\!\rangle, \langle\!\langle v \rangle\!\rangle \in V \text{ and } u \approx_W v \}$;
    $P_2 \coloneqq \{ \langle\!\langle uv \rangle\!\rangle \to \langle\!\langle u \rangle\!\rangle\langle\!\langle v \rangle\!\rangle \mid \langle\!\langle u \rangle\!\rangle, \langle\!\langle v \rangle\!\rangle, \langle\!\langle uv \rangle\!\rangle \in V \}$;
    for $a \in \Sigma \cap \mathrm{Sub}(W) - \Sigma_\Lambda$ **do**
      Create a Ψ-learner instance Λ_a;
      Add a to Σ_Λ;
    end
  end
  $P_0 \coloneqq \varnothing$;
  for $a \in \Sigma_\Lambda$ **do**
    Get a predicate φ from Λ_a by feeding it with elements of $[a]_W \cap \Sigma$ in an arbitrary order;
    Add $\langle\!\langle a \rangle\!\rangle \to \varphi$ to P_0;
  end
  Output $G_t = \langle \Psi, V, P_0 \cup P_1 \cup P_2, \langle\!\langle w_1 \rangle\!\rangle \rangle$;
end

Lemma 10 *Suppose that the algorithm creates a predicate learner instance Λ_a for $a \in \Sigma \cap \mathrm{Sub}(L_*)$. Then its hypotheses will eventually converge to φ_a such that $[\![\varphi_a]\!] = [a]_{L_*} \cap \Sigma$.*

Proof Since the algorithm has created Λ_a, there are $u, v \in \Sigma^*$ such that $uav \in W \subseteq L_*$. For every $b \in [a]_{L_*} \cap \Sigma$, at some point $ubv \in L_*$ appears in W as a positive example. Then, b is in $[a]_W \cap \Sigma$. After that, b will be given to Λ_a infinitely many times. Thus, Λ_a is fed with a positive presentation of $[a]_{L_*} \cap \Sigma$ and its hypotheses eventually converge to a predicate representing $[a]_{L_*} \cap \Sigma$. ■

Lemma 11 *If every rule of the form $\langle\!\langle a \rangle\!\rangle \to \varphi \in P_0$ is correct in our hypothesis grammar G, then $\mathrm{L}(G) \subseteq L_*$.*

Proof Define $f(\langle\!\langle u \rangle\!\rangle) = [u]_{L_*}$ for $\langle\!\langle u \rangle\!\rangle \in V$, $f(\varphi) = [\![\varphi]\!]$ for $\varphi \in \Psi$, and $f(a) = \{a\}$ for $a \in \Sigma$. Then, one can show that $\alpha \Rightarrow_G^* \beta$ with $\alpha, \beta \in (\Sigma \cup V \cup \Psi)^+$ implies $\hat{f}(\beta) \subseteq \hat{f}(\alpha)$, where $\hat{f}$ is the homomorphic extension of f. ■

Now we give a sort of a "characteristic set". The definition is just the same as the one defined for substitutable Σ-CFLs (Clark and Eyraud, 2007). Let $G_* = \langle \Psi, V_*, P_*, S_* \rangle$ be a Ψ-CFG in the branching normal form generating L_*. By Lemma 9, we assume that every predicate in G_* represents a congruence class $[a]_{L_*} \cap \Sigma$ for some $a \in \Sigma$. For each nonterminal symbol $N \in V_*$, define

$$\omega_N = \min \mathrm{L}(G_*, N)\,,$$
$$\chi_N = \min\{\, \langle x, z\rangle \in \Sigma^* \times \Sigma^* \mid S \Rightarrow^*_{G_*} xNz \,\}\,,$$

where min is with respect to the length-lexicographic order. Note that ω_N and χ_N are well-defined since the lexicographic order is well-ordered and thus so is the length-lexicographic order. If $\mathrm{L}(G_*, N) \cap \Sigma \neq \varnothing$, then $\omega_N \in \Sigma$. The *core set* W_* of G_* is defined by

$$W_* = \{\, \chi_N \odot \omega_N \mid N \in V_* \,\}$$
$$\cup \{\, \chi_N \odot \omega_{N_1}\omega_{N_2} \mid N \to N_1N_2 \in P_* \,\}\,.$$

We remark that while it is guaranteed that the Σ-CFG constructed using any superset of the core set is correct when learning substitutable Σ-CFLs, it is not necessarily the case for our learner. There may be some delay to let the predicate learner instances converge.

Theorem 12 *The class Ψ-$\mathcal{SCFL}$ is identifiable in the limit from positive data.*

Proof We show that our learner eventually converges to a grammar G such that $\mathrm{L}(G) = L_*$. Let t_0 be the least number such that $W_* \subseteq \{w_1, \dots w_{t_0}\}$.

(Case 1) For all $t \geq t_0$, it holds $\{w_1, \dots, w_t\} \subseteq \mathrm{L}(G_{t-1})$. In this case, the nonterminal set V and the predicate learner set Σ_Λ will never be updated after t_0. The difference between G_{t-1} and G_t can be only in the terminal rules of the form $\langle\!\langle a \rangle\!\rangle \to \varphi$. By Lemma 10, the hypotheses of each predicate learner Λ_a will converge to a correct predicate. Let t_1 be the point when all the predicate learners converge. We have $\mathrm{L}(G_{t_1}) \subseteq L_*$ by Lemma 11 and after that our algorithm will never update the hypothesis. The assumption that $\{w_1, \dots, w_t\} \subseteq \mathrm{L}(G_{t_1})$ for all $t \geq t_1$ means $\mathrm{L}(G_{t_1}) = L_*$.

(Case 2) For some $t \geq t_0$, it holds $\{w_1, \dots, w_t\} \not\subseteq \mathrm{L}(G_{t-1})$. At this moment, the learner's hypothesis is updated so that G_t has nonterminals $\langle\!\langle \omega_N \rangle\!\rangle$ for all $N \in V_*$. The start symbol S_* of G_* can be "simulated" by $\langle\!\langle w_1 \rangle\!\rangle \to \langle\!\langle \omega_{S_*} \rangle\!\rangle$ in G_t. Each rule of the form $N \to N_1N_2 \in P_*$ is simulated by the combination of the rules

$$\langle\!\langle \omega_N \rangle\!\rangle \to \langle\!\langle \omega_{N_1}\omega_{N_2} \rangle\!\rangle \in P_1 \qquad \text{by } \chi_N \odot \omega_N,\ \chi_N \odot \omega_{N_1}\omega_{N_2} \in W_*,$$
$$\langle\!\langle \omega_{N_1}\omega_{N_2} \rangle\!\rangle \to \langle\!\langle \omega_{N_1} \rangle\!\rangle \langle\!\langle \omega_{N_2} \rangle\!\rangle \in P_2\,.$$

For each rule of the form $N \to \varphi \in P_*$, we have a predicate learner Λ_{ω_N}. By Lemma 10, at some point $t_1 \geq t_0$ each predicate learner Λ_{ω_N} converges to a predicate ψ such that $[\![\psi]\!] = [\omega_N]_{L_*} \cap \Sigma$ and we have $\langle\!\langle \omega_N \rangle\!\rangle \to \psi \in P$. Now, the grammar G_t with $t \geq t_1$ can simulate every derivation of G_* and thus $L_* \subseteq \mathrm{L}(G_t)$ holds. After that, the same arguments of (Case 1) apply and finally our hypotheses converge to a correct grammar. ■

4. Learning efficiency

Our learning algorithm is not optimized on its efficiency. It creates many nonterminals which play obviously the same role, as well as Ψ-learner instances. Those should be merged for efficient algorithm implementation.

Nonetheless, evaluating the learning efficiency of a learner under the identification in the limit paradigm is controversial. Even with de la Higuera's criterion (1997), identification in the limit with polynomial time and data, which is one of the most popular criteria, still it is not obvious how we should apply the definition when learning CFLs. Moreover, anyway the efficiency of our algorithm depends on the predicate learners. Hence, it is not easy to give a clear theorem on the efficiency. Instead, this section gives a flavor that our algorithm would be reasonably efficient only.

Suppose that the Ψ-learner is efficient in de la Higuera's sense. That is, Λ returns a hypothesis in polynomial time in the total size of the given data, and each learning target $\varphi \in \Psi$ admits a finite set $K_\varphi \subseteq [\![\varphi]\!]$ of polynomial size in the description size of φ such that whenever a superset of K_φ is given to Λ, then it immediately converges to a correct predicate. Then, obviously our algorithm outputs a hypothesis grammar in polynomial time in the total size of the given data each time a new positive example is given. Moreover, the size of data required for convergence is not too big. Let us call

$$W'_* = W_* \cup \{\, \omega_N \odot K_\varphi \mid N \to \varphi \in P_* \,\}$$

a *semi-characteristic set*. The set cardinality $|W'_*|$ is linear in $\kappa\|G_*\|$ where κ is the maximum size of K_φ for predicates φ used in G_*. Note that the set W'_* is not necessarily the characteristic set in de la Higuera's sense. The algorithm may not converge immediately after the positive example set includes W'_*. Yet, once (Case 2) of the proof of Theorem 12 happens after observing W'_*, the convergence is immediate. On the other hand, in (Case 1), when the positive examples include W'_*, all the created predicate learner instances converge and so does our learner.

5. Discussions

This paper attempts to give a result parallel to Argyros and D'Antoni's work

- for learning (non-regular) CFLs
- in the paradigm of identification in the limit from positive data.

We succeeded in this challenge for substitutable languages. It was possible because the constructed nonterminals have clear semantics and we are able to find the right positive example characters for predicate learners along the semantics. The learning approaches generically called "distributional learning" use nonterminals with such semantics (Clark and Yoshinaka, 2016). Thus, it would be interesting and promising to target subclasses of CFLs which are "distributionally learnable" in various learning paradigms. Namely, identifying k,l-substitutable CFLs in the limit from positive data (Yoshinaka, 2008), learning context-deterministic and congruential CFLs from membership and equivalence queries (Shirakawa and Yokomori, 1993; Clark, 2010a), learning CFLs with finite kernel/context properties using

positive data and membership queries (Clark, 2010b; Yoshinaka, 2011a), and other more. Furthermore, there are distributional learnable languages beyond CFLs (Yoshinaka, 2011b; Yoshinaka and Kanazawa, 2011). Those should be also targeted as future work.

One might be unsatisfied with our definition of CFLs over infinite alphabets. The palindrome language with a center marker is a simple non-regular example of a substitutable Σ-CFL:

$$\{\, a_1 \dots a_n \# a_n \dots a_1 \mid a_1, \dots, a_n \in \Sigma - \{\#\},\ n \geq 0 \,\},$$

and many proposals extending CFLs to be over infinite alphabets include the palindrome language over infinite alphabets. However, our definition does not accept this language as a Ψ-CFL unless the alphabet is finite. It is interesting future work to target CFLs over infinite alphabets under more elaborated definitions than the one we have used in this paper.

Acknowledgments

The authors are grateful to the anonymous reviewers for their helpful comments. This work is supported in part by JSPS KAKENHI Grant Numbers JP18K11150 (RY), JP20H05703 (RY), and JP21K11745 (AS).

References

Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

George Argyros and Loris D'Antoni. The learnability of symbolic automata. In *30th International Conference on Computer Aided Verification*, pages 427–445, 2018.

George Argyros, Ioannis Stais, Aggelos Kiayias, and Angelos D. Keromytis. Back in black: Towards formal, black box analysis of sanitizers and filters. In *IEEE Symposium on Security and Privacy*, pages 91–109, 2016.

Jean-Michel Autebert, Joffroy Beauquier, and Luc Boasson. Langages sur des alphabets infinis. *Discrete Applied Mathematics*, 2(1):1–20, 1980.

Lenore Blum and Manuel Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28(2):125–155, 1975.

Edward Y. C. Cheng and Michael Kaminski. Context-free languages over infinite alphabets. *Acta Informatica*, 35:245–267, 1998.

Kaizaburo Chubachi, Diptarama Hendrian, Ryo Yoshinaka, and Ayumi Shinohara. Query learning algorithm for residual symbolic finite automata. In *10th International Symposium on Games, Automata, Logics, and Formal Verification*, pages 140–153, 2019.

Alexander Clark. Distributional learning of some context-free languages with a minimally adequate teacher. In *Grammatical Inference: Theoretical Results and Applications*, pages 24–37, 2010a.

Alexander Clark. Learning context free grammars with the syntactic concept lattice. In *Grammatical Inference: Theoretical Results and Applications*, pages 38–51, 2010b.

Alexander Clark and Rémi Eyraud. Polynomial identification in the limit of substitutable context-free languages. *Journal of Machine Learning Research*, 8:1725–1745, 2007.

Alexander Clark and Ryo Yoshinaka. Distributional learning of context-free and multiple context-free grammars. In *Topics in Grammatical Inference*, pages 143–172. Springer, 2016.

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

Samuel Drews and Loris D'Antoni. Learning symbolic automata. In *23rd International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 173–189, 2017.

Dana Fisman, Hadar Frenkel, and Sandra Zilles. Inferring symbolic automata. In *30th EACSL Annual Conference on Computer Science Logic*, pages 21:1–21:19, 2022.

Patrice Godefroid, Adam Kiezun, and Michael Y. Levin. Grammar-based whitebox fuzzing. In *Proceedings of the ACM SIGPLAN 2008 Conference on Programming Language Design and Implementation*, pages 206–215, 2008.

E Mark Gold. Language identification in the limit. *Information and Control*, 10(5):447–474, 1967.

Hai-Feng Guo and Mahadevan Subramaniam. Model-based test generation using extended symbolic grammars. *International Journal on Software Tools for Technology Transfer*, 16 (4):437–455, 2014.

Rupak Majumdar and Ru-Gang Xu. Directed test generation using symbolic grammars. In *22nd IEEE/ACM International Conference on Automated Software Engineering*, pages 134–143, 2007.

Oded Maler and Irini-Eleftheria Mens. A generic algorithm for learning symbolic automata from membership queries. In *Models, Algorithms, Logics and Tools*, pages 146–169. Springer, 2017.

Peter M. Maurer. Generating test data with enhanced context-free grammars. *IEEE Software*, 7(4):50–55, 1990.

Irini-Eleftheria Mens and Oded Maler. Learning regular languages over large ordered alphabets. *Logical Methods in Computer Science*, 11(3:13):1–22, 2015.

Hayato Mizumoto, Shota Todoroki, Diptarama, Ryo Yoshinaka, and Ayumi Shinohara. An efficient query learning algorithm for zero-suppressed binary decision diagrams. In *28th International Conference on Algorithmic Learning Theory*, volume 76, pages 360–371, 2017.

Atsuyoshi Nakamura. An efficient query learning algorithm for ordered binary decision diagrams. *Information and Computation*, 201(2):178–198, 2005.

Friedrich Otto. Classes of regular and context-free languages over countably infinite alphabets. *Discrete Applied Mathematics*, 12(1):41–56, 1985.

Hiromi Shirakawa and Takashi Yokomori. Polynomial-time MAT learning of c-deterministic context-free grammars. *Transaction of Information Processing Society of Japan*, 34(3): 380–390, 1993.

Kaito Suzuki, Diptarama Hendrian, Ryo Yoshinaka, and Ayumi Shinohara. Query learning algorithm for symbolic weighted finite automata. In *15th International Conference on Grammatical Inference*, pages 202–216, 2021.

Ryo Yoshinaka. Identification in the limit of k,l-substitutable context-free languages. In *Grammatical Inference: Algorithms and Applications*, pages 266–279, 2008.

Ryo Yoshinaka. Towards dual approaches for learning context-free grammars based on syntactic concept lattices. In *Developments in Language Theory*, pages 429–440, 2011a.

Ryo Yoshinaka. Efficient learning of multiple context-free languages with multidimensional substitutability from positive data. *Theoretical Computer Science*, 412(19):1821–1831, 2011b.

Ryo Yoshinaka and Makoto Kanazawa. Distributional learning of abstract categorial grammars. In *Logical Aspects of Computational Linguistics*, pages 251–266, 2011.

Appendix A.

We show that the union closure of the concept class of Example 1 is not identifiable in the limit from positive data, using the following theorem.

Theorem 13 (Blum and Blum, 1975) *If a learner $\mathcal{A}$ identifies a concept class $\mathcal{C}$ in the limit from positive data, for every positive presentation σ of a concept $C \in \mathcal{C}$, there is a prefix σ_t of σ, called a* locking sequence*, such that $\mathcal{A}(\sigma_t)$ represents C and $\mathcal{A}(\sigma_t \cdot \tau) = \mathcal{A}(\sigma_t)$ for any sequence τ of positive examples from C.*

Suppose a learning algorithm $\mathcal{A}$ identifies $C_{1,0}$. Obviously $\sigma = \langle 0, 1, 2, \ldots \rangle$ is a positive presentation of $C_{1,0}$. Let a prefix $\sigma_n = \langle 0, 1, 2, \ldots, n-1 \rangle$ of σ be a locking sequence. The sequence σ_n is also a prefix of a positive presentation of $\bigcup_{k<n} C_{n+1,k} \subsetneq C_{1,0}$. Thus, $\mathcal{A}$ cannot learn $\bigcup_{k<n} C_{n+1,k}$.