2007－AL－115 (9)
2007／11／30

ある種の有限状態変換器に対する多項式時間極限同定

若月 光夫† 富田 悦次†

†電気通信大学 電気通信学部 情報通信工学科 〒182-8585 東京都調布市調布ケ丘 1-5-1
E-mail: †{wakatuki,tomita}@ice.uec.ac.jp

あらまし 本稿では，strict prefix deterministic finite state transducer (SPDFST と略す) と呼ぶ，状態数が有限個の変換器の真部分クラスに対する，正の例からの極限同定問題を扱う．SPDFST および，SPDFST によって表現される言語 (受理される入力記号列とその際に出力される記号列の対の集合) に関する性質を示した後で，SPDFST のクラスに対する正の例からの極限同定アルゴリズムを提案する．また，Yokomori の定義において，SPDFST のクラスが正の例から多項式時間極限同定可能であることを示す．この極限同定可能性は，多項式の個数の特徴サンプルを与えることによって証明される．

Polynomial Time Identification of Finite State Transducers in Some Class*

Mitsuo WAKATSUKI† and Etsuji TOMITA†

†Department of Information and Communication Engineering, Faculty of Electro-Communications, The University of Electro-Communications, Chofugaoka 1-5-1, Chofu, Tokyo 182-8585, Japan
E-mail: †{wakatuki,tomita}@ice.uec.ac.jp

Abstract. This report is concerned with a subclass of finite state transducers, called *strict prefix deterministic finite state transducers* (*SPDFST* for short), and studies a problem of identifying the subclass in the limit from positive data. After providing some properties of languages represented by SPDFST's (that is, sets of pairs of input strings accepted by SPDFST's and their corresponding output strings), we show that the class of SPDFST's is polynomial time identifiable in the limit from positive data in the sense of Yokomori. This identifiability is proved by giving an exact *characteristic sample* of polynomial size for a language represented by an SPDFST.

1 Introduction

In the study of inductive inference of formal languages, Gold [5] defined the notion of *identification in the limit* and showed that the class of languages containing all finite sets and one infinite set, which is called a *superfinite* class, is not identifiable in the limit from *positive data*. This means that even the class of regular languages is not identifiable in the limit from positive data. Angluin [1] has given several conditions for a class of languages to be identifiable in the limit from positive data, and she has presented some examples of identifiable classes. She has also proposed subclasses of regular languages called k-reversible languages for each $k \geq 0$, and has shown that these classes are identifiable in the limit from positive data, requiring a polynomial time for updating conjectures [2].

From the practical point of view, the inductive inference algorithm must have a good *time efficiency* in addition to running with only positive data. One may define the notion of *polynomial time identification in the limit* in various ways. Pitt [10] has proposed a reasonable definition for polynomial time identifiability in the limit. By making a slight modification of his definition, Yokomori [14] has proposed another definition for polynomial time identifiability in the limit from positive data, and he has proved that a class of languages accepted by strictly deterministic automata (SDA's for short) [11, 14], which is a proper subclass of regular languages, is polynomial time identifiable in the limit from

*This work was supported in part by Grants-in-Aid for Scientific Research Nos. 13680435, 16300001 and 18500108 from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

positive data. He has also proved that a class of very simple languages, which is a proper subclass of simple languages, is polynomial time identifiable in the limit from positive data [15, 16]. We have proved that a class of Szilard strict deterministic restricted one-counter automata (Szilard strict DROCA's for short), which is a proper subclass of deterministic pushdown automata, is also polynomial time identifiable in the limit from positive data [12] in the sense of Yokomori. Note that the class of languages accepted by Szilard strict DROCA's is incomparable to the class of very simple languages.

An SDA is an extended deterministic finite automaton, which is intuitively a state transition graph in which the set X of labels for edges is a finite subset of strings over an alphabet Σ, that satisfies the following conditions: for any x in X, there uniquely exists an edge (a pair of states) whose label is x, and for any distinct labels x_1, x_2 in X, the first symbol of x_1 differs from that of x_2. This SDA can be also represented by a pair (M, φ) of a corresponding deterministic finite automaton M and a homomorphism $\varphi : \Sigma'^* \to \Sigma^*$ such that $X = \varphi(\Sigma')$ for some alphabet Σ', where the language accepted by M is in the class of Szilard languages of linear grammars [8]. That is, the class of languages accepted by SDA's is the extended class of Szilard languages of linear grammars. In a similar way to this, some kind of language classes can be extended. Let $\mathcal{L}$ be a class of languages over Σ' to be based on and $\mathcal{X}$ a class of finite subsets of strings over Σ, where there exists a morphism $\varphi : \Sigma'^* \to X^*$ for some $X \in \mathcal{X}$. The extended class of $\mathcal{L}$, denoted by $\mathcal{C}(\mathcal{L}, \mathcal{X})$, is defined by these $\mathcal{L}$ and $\mathcal{X}$. Kobayashi and Yokomori [7] proved that for each $k \geq 0$, a class $\mathcal{C}(\mathcal{R}ev_k, \mathcal{X}_0)$ of languages, where $\mathcal{R}ev_k$ is a class of k-reversible languages and $\mathcal{X}_0$ is a class of codes [4], is identifiable in the limit from positive data. In [13], we have given a sufficient condition for the extended class $\mathcal{C}(\mathcal{L}, \mathcal{X})$ to be identifiable in the limit from positive data and have presented a unified identification algorithm for it.

A transducer is an automaton which has output mechanism. We can regard a class of transducers as an extended class of automata. Oncina et al. [9] have proved that a class of onward subsequential transducers (OST for short), which is a proper subclass of finite state transducers, is polynomial time identifiable in the limit from positive data.

The present report deals with a subclass of finite state transducers called *strict prefix deterministic finite state transducers* (*SPDFST*'s for short), and discusses the identification problem of the class of SPDFST's. The class of SDA's forms a proper subclass of associated automata with SPDFST's. Moreover, the class of languages represented by SPDFST's (that is, the class of sets of pairs of input strings accepted by SPDFST's and their corresponding output strings) is incomparable to the class of languages represented by OST's. After providing some properties of languages represented by SPDFST's, we show that the class of SPDFST's is polynomial time identifiable in the limit from positive data in the sense of Yokomori [14]. The main result in this report provides another interesting instance of a class of transducers which is polynomial time identifiable in the limit. This identifiability is proved by giving an exact *characteristic sample* of polynomial size for a language represented by an SPDFST.

2 Definitions

2.1 Basic Definitions and Notation

We assume that the reader is familiar with the basics of automata and formal language theory. For the definitions and notation not stated here, see, e.g., [6].

An *alphabet* Σ is a finite set of symbols. For any finite set S of finite-length strings over Σ, we denote by S^* (respectively, S^+) the set of all finite-length strings obtained by concatenating zero (one, resp.) or more elements of S, where the *concatenation* of strings u and v is simply denoted by uv. In particular, Σ^* denotes the set of all finite-length strings over Σ. The string of length 0 (the empty string) is denoted by ε. We denote by $|w|$ the length of a string w and by $|S|$ the cardinality of a set S. A *language* over Σ is any subset L of Σ^*. For a string $w \in \Sigma^+$, first(w) denotes the first symbol of w. For a string $w \in \Sigma^*$, alph(w) denotes the set of symbols appearing in w. Moreover, for a language $L \subseteq \Sigma^*$, let alph$(L) = \cup_{w \in L}$ alph(w). For a string $w \in \Sigma^*$ and its prefix $x \in \Sigma^*$, $x\backslash w$ denotes the string $y \in \Sigma^*$ such that $w = xy$. Moreover, for a language $L \subseteq \Sigma^*$ and a string $x \in \Sigma^*$, let $x\backslash L = \{y \mid xy \in L\}$. For a set $S \subseteq \Sigma^*$, lcp(S) denotes the *longest common prefix* of S.

Let Σ be any alphabet and suppose that Σ is totally ordered by some binary relation $\prec$. Let $x = a_1 \cdots a_r$, $y = b_1 \cdots b_s$, where $r, s \geq 0$, $a_i \in \Sigma$ for $1 \leq i \leq r$, and $b_i \in \Sigma$ for $1 \leq i \leq s$. We write that $x \prec y$ if (i) $|x| < |y|$, or (ii) $|x| = |y|$ and there exists $k \geq 1$ so that $a_i = b_i$ for $1 \leq i < k$ and $a_k \prec b_k$. The relation $x \preceq y$ means that $x \prec y$ or $x = y$.

2.2 Polynomial Time Identification in the Limit from Positive Data

In this report, we adopt Yokomori's definition in [14] for the notion of polynomial time identification in the limit from positive data.

For any class of languages to be identified, let $\mathcal{R}$ be a *class of representations* for a class of languages. Instances of such representations are automata, grammars, and so on. Given an r in $\mathcal{R}$, $L(r)$ denotes the language represented by r. A *positive presentation* of $L(r)$ is any infinite sequence of data such that every $w \in L(r)$ occurs at least once in the sequence and no other string not in $L(r)$ appears in the sequence. Each element of $L(r)$ is called a *positive example* (or simply, *example*) of $L(r)$.

Let r be a representation in $\mathcal{R}$. An algorithm $\mathcal{A}$ is said to *identify r in the limit from positive data* iff $\mathcal{A}$ takes any positive presentation of $L(r)$ as an input, and outputs an infinite sequence of representations in $\mathcal{R}$ such that there exist r' in $\mathcal{R}$ and $j > 0$ so that for all $i \geq j$, the i-th conjecture (representation) r_i is identical to r' and $L(r') = L(r)$. A class $\mathcal{R}$ is *identifiable in the limit from positive data* iff there exists an algorithm $\mathcal{A}$ that, for any r in $\mathcal{R}$, identifies r in the limit from positive data.

Let $\mathcal{A}$ be an algorithm for identifying $\mathcal{R}$ in the limit from positive data. Suppose that after examining i examples, the algorithm $\mathcal{A}$ conjectures some r_i. We say that $\mathcal{A}$ makes an *implicit error of prediction* at step i if r_i is not consistent with the $(i+1)$-st example w_{i+1}, i.e., if $w_{i+1} \notin L(r_i)$.

Definition 1.(Yokomori [14], pp.157-158, Definition 2) A class $\mathcal{R}$ is *polynomial time identifiable in the limit from positive data* iff there exists an algorithm $\mathcal{A}$ for identifying $\mathcal{R}$ in the limit from positive data with the property that there exist polynomials p and q such that for any n, for any r of size n, and for any positive presentation of $L(r)$, the time used by $\mathcal{A}$ between receiving the i-th example w_i and outputting the i-th conjecture r_i is at most $p(n, \sum_{j=1}^{i} |w_j|)$, and the number of implicit errors of prediction made by $\mathcal{A}$ is at most $q(n, l)$, where the size of r is the length of a description for r and $l = \mathrm{Max}\{|w_j| \mid 1 \leq j \leq i\}$. □

3 Strict Prefix Deterministic Finite State Transducers

Definition 2.([9],[3]) A *finite state* or *rational transducer* (*FST* for short) is defined as a 6-tuple $T = (Q, \Sigma, \Delta, \delta, q_0, F)$, where Q is a finite set of *states*, Σ is an *input alphabet*, Δ is an *output alphabet*, δ is a finite subset of $Q \times \Sigma^* \times \Delta^* \times Q$ whose elements are called *transitions* or *edges*, q_0 is the *initial state*, and $F (\subseteq Q)$ is a set of *final states*. □

Associated with a transition $(p, x, y, q) \in \delta$ in an FST, there is a transition $(p, x, q) \in Q \times \Sigma^* \times Q$, which corresponds to the conventional concept in finite automata. A finite automaton $M = (Q, \Sigma, \delta', q_0, F)$, where $\delta' \subseteq Q \times \Sigma^* \times Q$ and $(p, x, y, q) \in \delta$ implies that $(p, x, q) \in \delta'$, is called an *associated automaton* with an FST T. In a directed graph, called a *transition graph*, associated with an FST T, a transition $(p, x, y, q) \in \delta$ is associated with an edge from node p to node q labeled x/y.

A *sequential transducer* is an FST in which $\delta \subseteq Q \times \Sigma \times \Delta^* \times Q$, $F = Q$, and (p, a, u, q), $(p, a, v, r) \in \delta$ implies that $u = v$ and $q = r$ (*determinism condition*) [9]. Since F can be omitted, a sequential transducer is completely specified as a 5-tuple $T = (Q, \Sigma, \Delta, \delta, q_0)$. Sequential transducers are also called *generalized sequential machines* (*GSM*'s for short) [6], where Mealy and Moore machines are restricted instances of GSM's.

A *path* in an FST T is a sequence of transitions $\pi = (p_0, x_1, y_1, p_1)(p_1, x_2, y_2, p_2) \cdots (p_{n-1}, x_n, y_n, p_n)$, where $p_i \in Q$ for $0 \leq i \leq n$, and $x_i \in \Sigma^*, y_i \in \Delta^*$ for $1 \leq i \leq n$. When the intermediate states involved in a path are insignificant, a path is written as $\pi = (p_0, x_1 x_2 \cdots x_n, y_1 y_2 \cdots y_n, p_n)$. For $p, q \in Q$, $\Pi_T(p, q)$ denotes the set of all paths from p to q. By convention, we let $\varepsilon \in \Pi_T(p, p)$ for any $p \in Q$. We extend this notation by setting

$\Pi_T(p, Q') = \cup_{q \in Q'} \Pi_T(p, q)$ for any $Q' \subseteq Q$. A path π from p to q is *successful* iff $p = q_0$ and $q \in F$. Thus, the set of all successful paths is $\Pi_T(q_0, F)$. Here, for a state $p \in Q$, it is said to be *reachable* if $\Pi_T(q_0, p) \neq \emptyset$, and it is said to be *live* if $\Pi_T(p, F) \neq \emptyset$. For an FST T, the *language* represented by T is defined to be $L(T) = \{(x, y) \in \Sigma^* \times \Delta^* \mid (q_0, x, y, q) \in \Pi_T(q_0, F)\}$. Moreover, define $L_I(T) = \{x \in \Sigma^* \mid (x, y) \in L(T)\}$ and $L_O(T) = \{y \in \Delta^* \mid (x, y) \in L(T)\}$. Here, $L_I(T)$ (respectively, $L_O(T)$) is called an *input language* (*output language*, resp.) accepted by T.

Definition 3. Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be an FST. Then, T is a *strict prefix deterministic finite state transducer* (*SPDFST* for short) iff T satisfies the following conditions:
(1) $\delta \subseteq Q \times \Sigma^+ \times \Delta^+ \times Q$.
(2) for any $(p, x_1, y_1, q_1), (p, x_2, y_2, q_2) \in \delta$, if $\mathrm{first}(x_1) = \mathrm{first}(x_2)$, then $x_1 = x_2$, $y_1 = y_2$ and $q_1 = q_2$ (*determinism condition*).
(3) for any $(p, x_1, y_1, q_1), (p, x_2, y_2, q_2) \in \delta$, if $\mathrm{first}(x_1) \neq \mathrm{first}(x_2)$, then $\mathrm{first}(y_1) \neq \mathrm{first}(y_2)$. (We say that δ has the *strict prefix property*.)
(4) for any $(p_1, x_1, y_1, q_1), (p_2, x_2, y_2, q_2) \in \delta$ with $p_1 \neq p_2$ or $q_1 \neq q_2$, $\mathrm{first}(x_1) \neq \mathrm{first}(x_2)$ or $\mathrm{first}(y_1) \neq \mathrm{first}(y_2)$ (i.e., the uniqueness of labels). □

Example 1. Consider an FST $T = (Q, \Sigma, \Delta, \delta, q_0, F)$, where $Q = \{q_0, q_1, q_2, q_3, q_4, q_5\}$, $\Sigma = \{0, 1, 2\}$, $\Delta = \{a, b, c, d\}$,

$$\begin{aligned}\delta = \{ &(q_0, 0, ab, q_1), (q_0, 1, dd, q_3),\\ &(q_1, 0, b, q_2), (q_1, 1, c, q_0), (q_1, 2, ac, q_4),\\ &(q_2, 0, cc, q_1), (q_2, 1, a, q_3),\\ &(q_3, 1, bc, q_2),\\ &(q_5, 0, da, q_4), (q_5, 2, cb, q_2) \},\end{aligned}$$

and $F = \{q_2\}$. This FST T is an SPDFST since T satisfies the conditions in Definition 3. □

An SPDFST $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ is said to be in *canonical form* if, for any $p \in Q$, p is reachable and live, and for any $p \in Q - \{q_0\}$, it holds that $p \in F$ or $|\{(p, x, y, q) \in \delta \mid x \in \Sigma^+, y \in \Delta^+, q \in Q\}| \geq 2$ (i.e., the number of outgoing edges from node p in a transition graph is more than or equal to 2).

For any SPDFST T, there exists an SPDFST T' in canonical form. We can show this by using the following procedure.

Input: a given SPDFST $T = (Q, \Sigma, \Delta, \delta, q_0, F)$
Output: an SPDFST $T' = (Q', \Sigma', \Delta', \delta', q_0, F')$ in canonical form
Procedure
begin
  /* eliminate useless states and transitions */
  $Q_U := \{p \in Q \mid p$ is not reachable or p is not live$\}$;
  $Q' := Q - Q_U$; $F' := F - Q_U$;
  $\delta' := \delta - \{(p, x, y, q) \in \delta \mid p \in Q_U$ or $q \in Q_U\}$;
  $\Sigma' := \mathrm{alph}(\{x \in \Sigma^+ \mid (p, x, y, q) \in \delta'\})$;
  $\Delta' := \mathrm{alph}(\{y \in \Delta^+ \mid (p, x, y, q) \in \delta'\})$;
  /* transformation to an SPDFST in canonical form */
  while there exists $p \in Q' - \{q_0\}$ such that $p \notin F'$ and $|\{(p, x, y, q) \in \delta' \mid x \in \Sigma'^+, y \in \Delta'^+, q \in Q'\}| = 1$ **do**
    $\delta' := \delta' \cup \{(r, ux, vy, q) \mid (r, u, v, p) \in \delta'\} - \{(r, u, v, p), (p, x, y, q)\}$;
    $Q' := Q' - \{p\}$
  od
end

Example 2. Let us apply the above procedure to the SPDFST T in Example 1. The resulting SPDFST in canonical form is $T' = (Q', \Sigma', \Delta', \delta', q_0, F')$, where $Q' = \{q_0, q_1, q_2\}$, $\Sigma' = \{0, 1\}$, $\Delta' = \{a, b, c, d\}$,

$$\begin{aligned}\delta' = \{ &(q_0, 0, ab, q_1), (q_0, 11, ddbc, q_2),\\ &(q_1, 0, b, q_2), (q_1, 1, c, q_0),\\ &(q_2, 0, cc, q_1), (q_2, 11, abc, q_2) \},\end{aligned}$$

and $F' = \{q_2\}$. □

For the above SPDFST T', we can show that $L(T') = L(T)$ for any SPDFST T. Furthermore, T' has a minimum number of states and is unique up to isomorphism.

Hereafter, we are concerned with SPDFST's T in canonical form.

The next lemma immediately follows from Definition 3.

Lemma 1. Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be an SPDFST, and let $p, p', q, q' \in Q$, $x, x' \in \Sigma^+$, and $y, y' \in \Delta^+$.
(1) If $(p, x, y, q), (p', x', y', q') \in \delta$ such that $\mathrm{first}(x) = \mathrm{first}(x')$ and $\mathrm{first}(y) = \mathrm{first}(y')$, then it holds that $p = p'$, $x = x'$, $y = y'$ and $q = q'$.
(2) If $(p, x, y, q), (p, x', y', q') \in \delta$ such that $q \neq q'$, then it holds that $\mathrm{first}(x) \neq \mathrm{first}(x')$ and $\mathrm{first}(y) \neq \mathrm{first}(y')$. □

The following lemma is derived from Definition 3 and Lemma 1.

Lemma 2. Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be an SPDFST, and let $p, p', q, q' \in Q$, $x, x' \in \Sigma^+$, and $y, y' \in \Delta^+$. Then, the followings hold.
(1) If $(p, x, y, q) \in \Pi_T(p, q)$ and $(p, x, y', q') \in \Pi_T(p, q')$, then it holds that $y = y'$ and $q = q'$.
(2) If $(p, x, y, q) \in \Pi_T(p, q)$ and $(p', x, y, q') \in \Pi_T(p', q')$, then it holds that $p = p'$ and $q = q'$.
(3) For some $\pi = (p, x, y, q) \in \Pi_T(p, q)$ and $\pi' = (p, x', y', q') \in \Pi_T(p, q')$, if $\mathrm{first}(x) = \mathrm{first}(x')$ and $\mathrm{first}(y) = \mathrm{first}(y')$, then π can be divided into $(p, x_c, y_c, r)\,(r, x_c\backslash x, y_c\backslash y, q)$ and π' can be divided into $(p, x_c, y_c, r)\,(r, x_c\backslash x', y_c\backslash y', q')$, where $x_c = \mathrm{lcp}(\{x, x'\}) \in \Sigma^+$, $y_c = \mathrm{lcp}(\{y, y'\}) \in \Delta^+$, and $r \in Q$. □

Using Definition 3 and Lemmas 1 and 2, it is easy to obtain the following lemmas.

Lemma 3. Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be an SPDFST and let $(x, y), (x_1, y_1), (x_2, y_2) \in L(T)$. Then, for each $a, a_1, a_2 \in \Sigma$ $(a_1 \neq a_2)$, $b, b_1, b_2 \in \Delta$ $(b_1 \neq b_2)$, the followings hold.
(1) If $x = ax''$ and $y = by''$ for some $x'' \in \Sigma^*, y'' \in \Delta^*$, then there exists a transition $(q_0, u, v, p) \in \delta$ such that $\mathrm{first}(u) = a$ and $\mathrm{first}(v) = b$ for some $p \in Q$.
(2) If $x_1 = x'a_1x_1'', x_2 = x'a_2x_2''$, $y_1 = y'b_1y_1''$ and $y_2 = y'b_2y_2''$ for some $x', x_1'', x_2'' \in \Sigma^*$, $y', y_1'', y_2'' \in \Delta^*$, then there exist $p, q_1, q_2 \in Q$, $u_1, u_2 \in \Sigma^+$, and $v_1, v_2 \in \Delta^+$ such that $(p, u_1, v_1, q_1), (p, u_2, v_2, q_2) \in \delta$ with $\mathrm{first}(u_1) = a_1$, $\mathrm{first}(u_2) = a_2$, $\mathrm{first}(v_1) = b_1$ and $\mathrm{first}(v_2) = b_2$.
(3) If $x_2 = x_1ax_2''$ and $y_2 = y_1by_2''$ for some $x_2'' \in \Sigma^*$, $y_2'' \in \Delta^*$, then there exist $p \in F$, $q \in Q$, $u \in \Sigma^+$, and $v \in \Delta^+$ such that $(p, u, v, q) \in \delta$ with $\mathrm{first}(u) = a$ and $\mathrm{first}(v) = b$. □

Lemma 4. Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be an arbitrary SPDFST. Then, for $x, z \in \Sigma^*$, $y \in \Sigma^+$, $u, w \in \Delta^*$, $v \in \Delta^+$, if $(xz, uw), (xyz, uvw), (xy^2z, uv^2w)$ are in $L(T)$, then for each $i \geq 0$, $(xy^iz, uv^iw) \in L(T)$. □

Finally, we mention the relationships among the class of SPDFST's, the class of SDA's [14], and the class of OST's [9].

Definition 4.([14]) An *extended deterministic finite automaton* (*EDFA* for short) over Σ is defined as a 5-tuple $M = (Q, X, \delta, q_0, F)$, where $X \subseteq \Sigma^+$, $\delta \subseteq Q \times X \times Q$, and $(p, x, q_1), (p, x, q_2) \in \delta$ implies that $q_1 = q_2$. □

The class of EDFA's is a proper subclass of associated automata with FST's.

Definition 5.([14]) Let $M = (Q, X, \delta, q_0, F)$ be an EDFA. Then, M is a *strictly deterministic automaton* (*SDA* for short) iff M satisfies the following conditions:
(1) for any $x \in X$, there uniquely exists a pair $(p, q) \in Q \times Q$ such that $(p, x, q) \in \delta$.
(2) for any $x_1, x_2 \in X$ such that $x_1 \neq x_2$, $\mathrm{first}(x_1) \neq \mathrm{first}(x_2)$. □

From Definition 5, we can show that the class of SDA's is a proper subclass of associated automata with SPDFST's.

Definition 6.([9]) A *subsequential transducer* is defined as a 6-tuple $T = (Q, \Sigma, \Delta, \delta, q_0, \lambda)$, where $T' = (Q, \Sigma, \Delta, \delta, q_0)$ is a sequential transducer, and $\lambda : Q \rightarrow \Delta^*$ is a function that assigns output strings to the state of T'. The language represented by T is defined to be $L(T) = \{(x, yz) \in \Sigma^* \times \Delta^* \mid (q_0, x, y, q) \in \Pi_T(q_0, Q), \lambda(q) = z\}$. □

Definition 7.([9]) A subsequential transducer $T = (Q, \Sigma, \Delta, \delta, q_0, \lambda)$ is an *onward subsequential transducer* (*OST* for short) that satisfies the following condition: $\mathrm{lcp}(\{y \in \Delta^* \mid (p, a, y, q) \in \delta\} \cup \{\lambda(p)\}) = \varepsilon$ for any $p \in Q - \{q_0\}$, $a \in \Sigma$. □

From Definitions 6 and 7, we can show that the class of languages represented by OST's is incomparable to the class of languages represented by SPDFST's.

4 Identifying SPDFST's

4.1 A Characteristic Sample

Let $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ be any SPDFST in canonical form. A finite subset $R \subseteq \Sigma^* \times \Delta^*$ of $L(T)$ is called a *characteristic sample* of $L(T)$ if $L(T)$ is the smallest language represented by an SPDFST containing R, i.e., if for any SPDFST T', $R \subseteq L(T')$ implies that $L(T) \subseteq L(T')$.

For each $p \in Q$, define $\mathrm{pre}(p)$ as the *shortest* input string $x \in \Sigma^*$ from q_0 to p, i.e., $(q_0, x, y, p) \in \Pi_T(q_0, p)$ and $x \preceq x'$ for any x' such that $(q_0, x', y', p) \in \Pi_T(q_0, p)$. Moreover, for each $p \in Q$ and $q \in F$, define $\mathrm{post}(p, q)$ $(\in \Sigma^*)$

as the *shortest* input string from p to q. Then, define

$$\begin{aligned} R_I(T) = &\{\, \mathrm{pre}(p) \cdot \mathrm{post}(p,q) \mid p \in Q,\, q \in F \} \\ &\cup \{\, \mathrm{pre}(p) \cdot x \cdot \mathrm{post}(r,q) \mid \\ &\quad p \in Q,\, (p,x,y,r) \in \delta, q \in F \} \\ &\cup \{\, \mathrm{pre}(p) \cdot x_1 \cdot x_2 \cdot \mathrm{post}(s,q) \mid \\ &\quad p \in Q,\, (p,x_1,y_1,r), (r,x_2,y_2,s) \in \delta, \\ &\quad q \in F \} \,(\subseteq L_I(T)) \end{aligned}$$

and

$$\begin{aligned} R(T) = \{\, &(x,y) \in \Sigma^* \times \Delta^* \mid x \in R_I(T), \\ &(q_0,x,y,q) \in \Pi_T(q_0,F) \}. \end{aligned}$$

$R(T)$ is called a *representative sample* of T. Note that the cardinality $|R(T)|$ of a representative sample is at most $|Q|\,|F|\,(|\Sigma|^2 + |\Sigma| + 1)$, that is, $|R(T)|$ is polynomial with repect to the description length of T.

Example 3. Consider an SPDFST $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ in canonical form, where T is isomorphic to the SPDFST T' in Example 2. For Σ, let $0 \prec 1$. Then, we have the following.

$$\begin{aligned} &\mathrm{pre}(q_0) = \varepsilon, \quad &&\mathrm{post}(q_0,q_2) = 00, \\ &\mathrm{pre}(q_1) = 0, \quad &&\mathrm{post}(q_1,q_2) = 0, \\ &\mathrm{pre}(q_2) = 00, \quad &&\mathrm{post}(q_2,q_2) = \varepsilon. \end{aligned}$$

Therefore, we have that

$$\begin{aligned} R_I(T) = \{\, &00,\, 11,\, 0000,\, 0011,\, 0100, \\ &0111,\, 1100,\, 1111, \\ &000100,\, 001100,\, 001111 \,\}. \end{aligned}$$

Consequently, we obtain that

$$\begin{aligned} R(T) = \{\, &(00, abb),\, (11, ddbc), \\ &(0000, abbccb),\, (0011, abbabc), \\ &(0100, abcabb),\, (0111, abcddbc), \\ &(1100, ddbcccb),\, (1111, ddbcabc), \\ &(000100, abbcccabb), \\ &(001100, abbabcccb), \\ &(001111, abbabcabc) \,\}. \end{aligned}$$

□

The next lemma assures that the representative sample $R(T)$ is a characteristic sample of $L(T)$.

Lemma 5. The representative sample $R(T)$ of an SPDFST $T = (Q, \Sigma, \Delta, \delta, q_0, F)$ in canonical form is a characteristic sample of $L(T)$. That is, it holds that for any language L' ($\subseteq \Sigma^* \times \Delta^*$) represented by an SPDFST, $R(T) \subseteq L'$ implies that $L(T) \subseteq L'$.

Proof: By construction, it holds that $R(T) \subseteq L(T)$. Let $T' = (Q', \Sigma, \Delta, \delta', q_0', F')$ be an SPDFST representing L' (i.e., $L' = L(T')$). Then, it should follow that we have the following claim.

[Claim] Let $p \in Q$ be any state of T, and let $(q_0, u, v, p) \in \Pi_T(q_0, p)$ for some $u \in \Sigma^*$, $v \in \Delta^*$. Moreover, let $(q_0', u, v, p') \in \Pi_{T'}(q_0', p')$ for some $p' \in Q'$. If $\pi = (p, x, y, q) \in \Pi_T(p, F)$ with $x \in \Sigma^*$ and $y \in \Delta^*$ for some $q \in F$, then $(p', x, y, q') \in \Pi_{T'}(p', F')$ for some $q' \in F'$.

[Proof of Claim] Let $\pi \in \Pi_T(p, F)$ be a path such that $(p_0, u_1, v_1, p_1)\,(p_1, u_2, v_2, p_2) \cdots (p_{n-1}, u_n, v_n, p_n)$ with $p_0 = p$, $p_n = q$, $x = u_1 \cdots u_n$ and $y = v_1 \cdots v_n$, where $(p_{i-1}, u_i, v_i, p_i) \in \delta$ for $1 \leq i \leq n$. Since $R(T) \subseteq L(T')$, this claim can be proved by induction on $n \geq 0$ using Lemmas 2, 3 and 4. □

4.2 Identification Algorithm

Let $T_* = (Q_*, \Sigma_*, \Delta_*, \delta_*, q_0, F_*)$ be a target SPDFST. We now present an idenitification algorithm *IA*. This algorithm is given in the following.

Input: a positive presentation $(x_1, y_1), (x_2, y_2), \ldots$ of a language $L(T_*)$ for a target SPDFST T_*
Output: a sequence of SPDFST's $T_1, T_2, \ldots$
Procedure *IA*
begin
  initialize $i = 0$; $q_0 := p_{[\varepsilon]}$;
  let $T_0 = (\{q_0\}, \emptyset, \emptyset, \emptyset, q_0, \emptyset)$ be the initial SPDFST;
  repeat (forever)
    $i := i + 1$;
    let $T_{i-1} = (Q_{i-1}, \Sigma_{i-1}, \Delta_{i-1}, \delta_{i-1}, q_0, F_{i-1})$ be the current conjecture;
    read the next positive example (x_i, y_i);
    if $(x_i, y_i) \in L(T_{i-1})$ **then**
      output $T_i = T_{i-1}$ as the i-th conjecture
    else
      $Q_i := Q_{i-1}$; $\Sigma_i := \Sigma_{i-1}$; $\Delta_i := \Delta_{i-1}$;
      $\delta_i := \delta_{i-1}$; $F_i := F_{i-1}$;
      if $(x_i, y_i) = (\varepsilon, \varepsilon)$ **then**
        $F_i := F_{i-1} \cup \{q_0\}$;
        output $T_i = (Q_i, \Sigma_i, \Delta_i, \delta_i, q_0, F_i)$ as the i-th conjecture
      else
        /* the case where $x_i \neq \varepsilon$ and $y_i \neq \varepsilon$ */
        $\Sigma_i := \Sigma_i \cup \mathrm{alph}(x_i)$;
        $\Delta_i := \Delta_i \cup \mathrm{alph}(y_i)$;
        $Q_i := Q_i \cup \{p_{[x_i]}\}$; $F_i := F_i \cup \{p_{[x_i]}\}$;

```
      if δ_{i-1} = ∅ then
        δ_i := {(q_0, x_i, y_i, p_[x_i])}
      else
        /* parse the i-th example */
        p := q_0;
        u := x_i;   v := y_i;   z := ε;
        while u ≠ ε do
          if there exists (p, u', v', q) ∈ δ_i
             such that first(u') = first(u)
             then
            u_c := lcp({u', u});
            u_s := u_c\u;   u'_s := u_c\u';
            v_c := lcp({v', v});
            v_s := v_c\v;   v'_s := v_c\v';
            if u'_s = ε then p := q
            else
              Q_i := Q_i ∪ {p_[zu_c]};
              δ_i := δ_i ∪ {(p, u_c, v_c, p_[zu_c]),
                            (p_[zu_c], u'_s, v'_s, q)}
                     -{(p, u', v', q)};
              p := p_[zu_c]
            fi
            u := u_s;   v := v_s;   z := zu_c
          else
            δ_i := δ_i ∪ {(p, u, v, p_[x_i])};
            u := ε;   v := ε
          fi od fi
        T_i := CONSTRUCT(Q_i, Σ_i, Δ_i, δ_i,
                         q_0, F_i);
        output T_i as the i-th conjecture
    fi fi
  until (false)
end

Function CONSTRUCT(Q, Σ, Δ, δ, q_0, F)
  Procedure MERGE(p_[x_1], p_[x_2])
  begin
    let z_1 be a string so that m(p_[z_1]) = m(p_[x_1])
    and z_1 ⪯ z' for any z'
    such that m(p_[z']) = m(p_[z_1]);
    let z_2 be a string so that m(p_[z_2]) = m(p_[x_2])
    and z_2 ⪯ z' for any z'
    such that m(p_[z']) = m(p_[z_2]);
    if z_1 ⪯ z_2 then m(p_[x_2]) := m(p_[x_1])
    else m(p_[x_1]) := m(p_[x_2]) fi
  end
begin
  repeat
    flag := true;
    /* merge identical states */
    for each p ∈ Q do m(p) := p od;
    while there exist two distinct transitions
          (p_[u_1], x, y, p_[v_1]), (p_[u_2], x, y, p_[v_2]) ∈ δ
          do
      call procedure MERGE(p_[u_1], p_[u_2]);
      call procedure MERGE(p_[v_1], p_[v_2]);
      δ := {(m(p'), x', y', m(q')) |
            (p', x', y', q') ∈ δ};
      F := {m(p) | p ∈ F};
      Q := {m(p) | p ∈ Q}
    od
    /* modify transitions to be deterministic */
    while there exist two distinct transitions
          (p_[u_1], x_1, y_1, q_1), (p_[u_2], x_2, y_2, q_2) ∈ δ
          such that x_1 = x'x''_1, x_2 = x'x''_2,
          x' = lcp({x_1, x_2}) ∈ Σ^+, x''_2 ∈ Σ^+,
          y_1 = y'y''_1, y_2 = y'y''_2,
          y' = lcp({y_1, y_2}) ∈ Δ^+, y''_2 ∈ Δ^+ do
      call procedure MERGE(p_[u_1], p_[u_2]);
      let z be a string so that
      m(p_[z]) = m(p_[u_1]) and z ⪯ z' for any z'
      such that m(z') = m(z);
      δ := δ - {(p_[u_1], x_1, y_1, q_1),
                (p_[u_2], x_2, y_2, q_2)};
      if x''_1 = ε (and y''_1 = ε) then
        δ := δ ∪ {(p_[z], x', y', q_1), (q_1, x''_2, y''_2, q_2)}
      else /* the case where
              first(x''_1) ≠ first(x''_2) */
        Q := Q ∪ {p_[zx']};
        δ := δ ∪ {(p_[z], x', y', p_[zx']),
                  (p_[zx'], x''_1, y''_1, q_1),
                  (p_[zx'], x''_2, y''_2, q_2)}
      fi
      δ := {(m(p̄), x̄, ȳ, m(q̄)) | (p̄, x̄, ȳ, q̄) ∈ δ};
      F := {m(p) | p ∈ F};
      Q := {m(p) | p ∈ Q};
      flag := false
    od
  until (flag = true);
  return T = (Q, Σ, Δ, δ, q_0, F)
end
```

By analyzing the behavior of the identification algorithm IA in the similar way as in [14], we can prove the following lemma from Lemma 5.

Lemma 6. Let $R(\subseteq \Sigma^* \times \Delta^*)$ be a characteristic sample of a language represented by some SPDFST T and let R' be a finite set such that $R \subseteq R' \subset L(T)$. Then, whenever the identification algorithm IA receives all positive examples in R', IA outputs an SPDFST which is isomorphic to T. □

From Lemmas 5 and 6, the next lemma follows.

Lemma 7. Let $T_1, T_2, \ldots, T_i, \ldots$ be a sequence of conjectured SPDFST's produced by IA. Then, there exists $r \geq 1$ such that for all $j \geq 0$, $T_r = T_{r+j}$ and $L(T_r) = L(T_*)$. □

Thus, we have the following theorem.

Theorem 1. The class of SPDFST's is identifiable in the limit from positive data. □

Let $K_i = \sum_{j=1}^{i}(|x_i| + |y_i|)$, $l_i = \mathrm{Max}\{|x_i|, |y_i| \mid 1 \leq j \leq i\}$, and $\mathrm{size}(T_*) = |Q_*| + |\Sigma_*| + |\Delta_*| + |F_*| + \sum_{(p,x,y,q)\in\delta_*}(|x|+|y|+2)+1$. By analyzing the time complexity of IA in the same way as in [14], we can show that (1) the time for updating a conjecture is bounded by $\mathcal{O}({K_i}^2)$ and (2) the number of implicit errors of prediction IA makes is bounded by $\mathcal{O}(n^2\,|\Sigma_*|^2\,l_i)$, where $n = \mathrm{size}(T_*)$. Then, we have the following theorem.

Theorem 2. The class of SPDFST's is polynomial time identifiable in the limit from positive data in the sense of Yokomori (Definition 1). □

5 Conclusions

We have shown that the class of SPDFST's is identifiable in the limit from positive data, and have presented an algorithm that identifies any SPDFST in polynomial time in the sense of Yokomori.

Acknowledgments The authors would like to thank Takashi Nishita for his contribution in an early stage of this work.

References

[1] Angluin, D., *Inductive inference of formal languages from positive data*, Inform. and Control **45** (1980), 117–135.

[2] Angluin, D., *Inference of reversible languages*, J. ACM **29** (1982), 741–765.

[3] Berstel J., "Transductions and Context-Free Languages", Teubner Studienbücher, Stuttgart, 1979.

[4] Berstel, J. and D. Perrin, "Theory of Codes", Academic Press, Inc., 1985.

[5] Gold, E. M., *Language identification in the limit*, Inform. and Control **10** (1967), 447–474.

[6] Harrison, M. A., "Introduction to Formal Language Theory", Addison-Wesley, Reading, Massachusetts, 1972.

[7] Kobayashi, S. and T. Yokomori, *Identifiability of subspaces and homomorphic images of zero-reversible languages*, ALT'97, LNAI **1316** (1997), 48–61.

[8] Mäkinen, E., *The grammatical inference problem for the Szilard languages of linear grammars*, Inform. Process. Lett. **36** (1990), 203–206.

[9] Oncina, J., P. García, and E. Vidal, *Learning subsequential transducers for pattern recognition interpretation tasks*, IEEE Trans. on Pattern Analysis and Machine Intelligence, **15**(5) (1993), 448–458.

[10] Pitt, L., *Inductive inference, DFAs, and computational complexity*, Proc. 2nd Workshop on Analogical and Inductive Inference, LNAI **397** (1989), 18–44.

[11] Tanida, N. and T. Yokomori, *Polynomial-time identification of strictly regular languages in the limit*, IEICE Trans. on Information and Systems **E75-D** (1992), 125–132.

[12] Wakatsuki, M., K. Teraguchi, and E. Tomita, *Polynomial time identification of strict deterministic restricted one-counter automata in some class from positive data*, ICGI2004, LNAI **3264** (2004), 260–272.

[13] Wakatsuki, M., E. Tomita, and G. Yamada: *A unified algorithm for extending classes of languages identifiable in the limit from positive data*, ICGI2006, LNAI **4201** (2006), 161-174.

[14] Yokomori, T., *On polynomial-time learnability in the limit of strictly deterministic automata*, Machine Learning **19** (1995), 153–179.

[15] Yokomori, T., *Polynomial-time identification of very simple grammars from positive data*, Theoretical Computer Science **298** (2003), 179–206.

[16] Yokomori, T., *Erratum to "Polynomial-time identification of very simple grammars from positive data [Theoret. Comput. Sci. 298 (2003) 179–206]"*, Theoretical Computer Science **377** (2007), 282–283.