On Learning Function Distinguishable Languages

Henning Fernau

WSI-2000-13

Henning Fernau

Wilhelm-Schickard-Institut für Informatik
Universität Tübingen
Sand 13
D-72076 Tübingen
Germany

E-Mail: `fernau@informatik.uni-tuebingen.de`
Telefon: (07071) 29-77565
Telefax: (07071) 29-5061

ISSN 0946-3852

On Learning Function Distinguishable Languages

Henning Fernau
Wilhelm-Schickard-Institut für Informatik
Universität Tübingen
Sand 13, D-72076 Tübingen, Germany
fernau@informatik.uni-tuebingen.de

June 28, 2000

Abstract

We show how appropriately chosen functions which we call *distinguishing* can be used to make deterministic finite automata backward deterministic. These ideas can be exploited to design regular language classes identifiable in the limit from positive samples. Special cases of this approach are the k-reversible and terminal distinguishable languages as discussed in [1, 7, 8, 14, 15].

1 Introduction

The learning model we use is *identification in the limit from positive samples* as proposed by Gold [10]. Here, a language class $\mathcal{L}$ (defined via a class of grammars or automata $\mathcal{G}$) is *identifiable* if there is a so-called *inference machine* I to which an arbitrary language $L \in \mathcal{L}$ may be enumerated (possibly with repetitions) in an arbitrary order, i.e., I receives an infinite input stream of words $w_1, w_2, \dots$ with $L = \{ w_i \mid i \in \mathbb{N} \}$, and I reacts with an output stream $G_i \in \mathcal{G}$ such that for all L, there is an $N(L)$ such that for all enumerations of L, for all corresponding response grammar sequences G_i and for all $n \geq N(L)$, we have $G_n = G_{N(L)}$ and, moreover, the language defined by $G_{N(L)}$ equals L.

This model is rather weak, since Gold already has shown [10] that any language class which contains all finite languages and one infinite language is not identifiable in the limit from positive samples. On the other hand, the model is very natural, since in most applications, negative samples are not available. There are two answers to this problem: (1) One could allow certain imprecision in the inference process; this has been done quite successfully within the PAC model proposed by Valiant [17] or, in another sense, by several heuristic approaches to the learning problem (including genetic algorithms or neural networks). (2) One could investigate how far one could get when maintaining the original "deterministic" model. The present paper makes some steps in the second direction.

The main point of this paper is to give a unified view on several identifiable language families through what we call f-distinguishing functions. In particular, to our knowledge this provides the first complete correctness proof of some published learning algorithms, as, e.g., in the case of terminal distinguishable languages. Among the language families which turn out to be special cases of our approach are the k-reversible languages [1] and the terminal-distinguishable languages [14, 15], which belong, according to Gregor [11], to the most popular identifiable regular language classes.

2 Definitions

2.1 Formal language prerequisites

Σ^* is the set of words over the alphabet Σ. Σ^k ($\Sigma^{<k}$) collects the words whose lengths are equal to (less than) k. λ denotes the empty word. Pref(L) is the set of prefixes of L and $u^{-1}L = \{v \in \Sigma^* | uv \in L\}$ is the quotient of $L \subseteq \Sigma^*$ by u.

We assume that the reader knows that regular languages can be characterized either (1) by left-linear grammars $G = (N, T, P, S)$, where N is the set of nonterminal symbols, T is the set of terminal symbols, $P \subset N \times (N \cup \{\lambda\})T^*$ is the rule set and $S \in N$ is the start symbol, or (2) by (deterministic) finite automata $A = (Q, T, \delta, q_0, Q_F)$, where Q is the state set, $\delta \subseteq Q \times T \times Q$ is the transition relation, $q_0 \in Q$ is the initial state and $Q_F \subseteq Q$ is the set of final states. As usual, δ^* denotes the extension of the transition relation to arbitrarily long input words. The language defined by a grammar G (or an automaton A) is written $L(G)$ (or $L(A)$, respectively). An automaton is called *stripped* iff all states are accessible from the initial state and all states lead to some final state.

We denote the minimal deterministic automaton of the regular language L by $A(L)$. Recall that $A(L) = (Q, T, \delta, q_0, Q_F)$ can be described as follows: $Q = \{u^{-1}L | u \in \text{Pref}(L)\}$, $q_0 = \lambda^{-1}L = L$; $Q_F = \{u^{-1}L | u \in L\}$; and $\delta(u^{-1}L, a) = (ua)^{-1}L$ with $u, ua \in \text{Pref}(L)$, $a \in T$.

Furthermore, we need two automata constructions in the following:

The *product automaton* $A = A_1 \times A_2$ of two automata $A_i = (Q_i, T, \delta_i, q_{0,i}, Q_{F,i})$ for $i = 1, 2$ is defined as $A = (Q, T, \delta, q_0, Q_F)$ with $Q = Q_1 \times Q_2$, $q_0 = (q_{0,1}, q_{0,2})$, $Q_F = Q_{F,1} \times Q_{F,2}$, $((q_1, q_2), a, (q'_1, q'_2)) \in \delta$ iff $(q_1, a, q'_1) \in \delta_1$ and $(q_2, a, q'_2) \in \delta_2$.

A *partition* of a set S is a collection of pairwise disjoint nonempty subsets of S whose union is S. If π is a partition of S, then, for any element $s \in S$, there is a unique element of π containing s, which we denote $B(s, \pi)$ and call the *block* of π containing s. A partition π is said to *refine* another partition π' iff every block of π' is a union of blocks of π. If π is any partition of the state set Q of the automaton $A = (Q, T, \delta, q_0, Q_F)$, then the *quotient automaton* $\pi^{-1}A = (\pi^{-1}Q, T, \delta', B(q_0, \pi), \pi^{-1}Q_F)$ is given by $\pi^{-1}\hat{Q} = \{ B(q, \pi) \mid q \in \hat{Q} \}$ (for $\hat{Q} \subseteq Q$) and $(B_1, a, B_2) \in \delta'$ iff $\exists q_1 \in B_1 \exists q_2 \in B_2 : (q_1, a, q_2) \in \delta$.

2.2 Distinguishing functions

In order to avoid cumbersome case discussions, let us fix now the terminal alphabet T of the left-linear grammars or automata we are going to discuss.

Definition 1 Let F be some finite set. A mapping $f : T^* \to F$ is called *distinguishing function* if $f(w) = f(z)$ implies $f(wu) = f(zu)$ for all $u, w, z \in T^*$.

In the literature, we can find the terminal function [15]

$$\text{Ter}(x) = \{ a \in T \mid \exists u, v \in T^* : uav = x \}$$

and, more generally, the k-terminal function [8]

$$\begin{aligned} \text{Ter}_k(x) &= (\pi_k(x), \mu_k(x), \sigma_k(x)), \quad \text{where} \\ \mu_k(x) &= \{ a \in T^{k+1} \mid \exists u, v \in T^* : uav = x \} \end{aligned}$$

and $\pi_k(x)$ $[\sigma_k(x)]$ is the prefix [suffix] of length k of x if $x \notin T^{<k}$, and $\pi_k(x) = \sigma_k(x) = x$ if $x \in T^{<k}$. The example $f(x) = \sigma_k(k)$ leads to the k-reversible languages, confer [1, 8] Other examples of distinguishing functions in the context of even linear languages can be found in [7].

Observe that every regular language R induces, via its Nerode equivalence classes a distinguishing function f_R, where $f_R(w)$ maps w to the equivalence class containing w. In some sense, these are the only distinguishing functions, since one can associate to every distinguishing function f a finite automaton $A_f = (F, T, \delta_f, f(\lambda), F)$ by setting $\delta_f(q, a) = f(wa)$, where $w \in f^{-1}(q)$ can be chosen arbitrarily, since f is a distinguishing function.

Definition 2 Let $G = (N, T, P, S)$ be a left-linear grammar with

$$P \subseteq (N \setminus \{S\}) \times ((N \setminus \{S\})T \cup \{\lambda\}) \cup \{S\} \times (N \setminus \{S\}).$$

Let $f : T^* \to F$ be a distinguishing function. We will say that G is *f-distinguishable* if:

1. G is *backward deterministic.* ($B \to w$ and $C \to w$ implies $B = C$).
2. For all $A \in N \setminus \{S\}$ and for all $x, y \in L(G, A)$,[1] we have $f(x) = f(y)$. (In other words, for $A \in N \setminus \{S\}$, $f(A) := f(x)$ for some $x \in L(G, A)$ is well-defined.)
3. If (a) $S \to B$ and $S \to C$ are in P or if (b) $A \to Ba$ and $A \to Ca$ are in P with $B \neq C$, then $f(B) \neq f(C)$.

A language is called *f-distinguishable* iff it can be generated by an f-distinguishable left-linear grammar.

The family of f-distinguishable languages is denoted by f-DL.

Remark 1 Our notation is adapted from the so-called terminal distinguishable languages introduced by Radhakrishnan and Nagaraja in [15]. We use left-linear grammars, while they use right-linear grammars in their definitions. This means that, e.g., the class Ter-DL coincides with the reversals (mirror images) of the class of terminal distinguishable languages, as exhibited in [7].[2]

[1] We will denote by $L(G, A)$ the language obtained by the grammar $G_A = (N, T, P, A)$.

[2] Note that their definition of terminal distinguishable right-linear grammar does not completely coincide with ours, but in order to maintain their results, their definition should be changed acccordingly.

Definition 3 Let $A = (Q, T, \delta, q_0, Q_F)$ be a finite automaton. Let $f : T^* \to F$ be a distinguishing function. A is called *f-distinguishable* if:

1. A is deterministic.
2. For all states $q \in Q$ and all $x, y \in T^*$ with $\delta^*(q_0, x) = \delta^*(q_0, y) = q$, we have $f(x) = f(y)$.

   (In other words, for $q \in Q$, $f(q) := f(x)$ for some x with $\delta^*(q_0, x) = q$ is well-defined.)
3. For all $q_1, q_2 \in Q$, $q_1 \neq q_2$, with either (a) $q_1, q_2 \in Q_F$ or (b) there exist $q_3 \in Q$ and $a \in T$ with $\delta(q_1, a) = \delta(q_2, a) = q_3$, we have $f(q_1) \neq f(q_2)$.

We need a suitable notion of a canonical automaton in the following.

Definition 4 Let $f : T^* \to F$ be a distinguishing function and let $L \subseteq T^*$ be a regular set. Let $A(L, f)$ be the stripped version of the product automaton $A(L) \times A_f$, i.e., delete all states that are not accessible from the initial state or does not lead to a final state of $A(L) \times A_f$. $A(L, f)$ is called *f-canonical automaton* of L.

Observe that an f-canonical automaton trivially obeys the first two restrictions of an f-distinguishing automaton. Clearly, $L(A(L, f)) = L$.

3 Characteristic Properties

In order to simplify the discussions below, we will always consider only the case of non-empty languages.

Remark 2 Let L be f-distinguishable. If $u_1v, u_2v \in L \subseteq T^*$ and $f(u_1) = f(u_2)$, then $\delta^*(q_0, u_1) = \delta^*(q_0, u_2)$ for any f-distinguishing automaton $A = (Q, T, \delta, q_0, Q_F)$ accepting L.

Proof. Consider the final states $q_i = \delta^*(q_0, u_iv)$ of A for $i = 1, 2$. Since $f(q_i) = f(u_iv)$ and since $f(u_1) = f(u_2)$ implies that $f(u_1v) = f(u_2v)$, condition 3a. in the definition of f-distinguishing automata yields $q_1 = q_2$.

By induction, and using condition 3b. in the induction step argument, one can show that $\delta^*(q_0, u_1v') = \delta^*(q_0, u_2v')$ for every prefix v' of v. This yields the desired claim. □

Theorem 5 (Characterization theorem) *The following conditions are equivalent for a regular language $L \subseteq T^*$ and a distinguishing function $f : T^* \to F$:*

1. *L is f-distinguishable.*
2. *For all $u, v, w, z \in T^*$ with $f(w) = f(z)$, we have $zu \in L \iff zv \in L$ whenever $\{wu, wv\} \subseteq L$.*
3. *For all $u, v, w, z \in T^*$ with $f(w) = f(z)$, we have $u \in z^{-1}L \iff v \in z^{-1}L$ whenever $u, v \in w^{-1}L$.*
4. *The f-canonical automaton of L is f-distinguishable.*
5. *L is accepted by an f-distinguishable automaton.*
6. *For all $u_1, u_2, v \in T^*$ with $f(u_1) = f(u_2)$, we have $u_1^{-1}L = u_2^{-1}L$ whenever $\{u_1v, u_2v\} \subseteq L$.*

Proof. '1. $\to$ 2.:' Assume firstly that L is generated by an f-distinguishable left-linear grammar $G = (N, T, P, S)$. Consider $\{wu, wv\} \subseteq L$. Since G is backward deterministic, there will be a single nonterminal A that will generate w, and both $S \Rightarrow^* Au$ and $S \Rightarrow^* Av$. More specifically, let $u = a_r \dots a_1$ and

$$S \Rightarrow X_0 \Rightarrow X_1 a_1 \Rightarrow X_2 a_2 a_1 \Rightarrow \dots \Rightarrow X_{r-1} a_{r-1} \dots a_1 \Rightarrow X_r a_r \dots a_1 = Au \quad (1)$$

be the above-mentioned derivation. Consider now a word zu. By definition of distinguishing functions, we have $f(zu) = f(wu)$, since $f(z) = f(w)$. This means that any derivation of zu via G must start with $S \Rightarrow X_0$, since otherwise the third condition (part (a)) of f-distinguishable grammars would be violated. By repeating this argument, taking now part (b) of the definition, we can conclude that any derivation of zu via G must start as depicted in Equation (1). Similarly, one can argue that the derivation of zv must start as any derivation of wv for the common suffix v. This means that any possible derivation of zu via G leads to the nonterminal A after processing the suffix u, and any possible derivation of zv via G leads to the nonterminal A after processing the suffix v, as well. Therefore, $zu \in L$ iff $zv \in L$ as required.

'2. $\leftrightarrow$ 3.' is trivial.

'3. $\to$ 4.:' We have to consider cases 3a. and 3b. of the definition of f-distinguishable automaton. We will prove that the f-canonical automaton

$A = A(L, f) = (Q, T, \delta, q_0, Q_F)$ of L is indeed f-distinguishable by using two similar contradiction arguments.

Assume firstly that there exist two different final states q_1, q_2 of A, i.e., $q_i = (w_i^{-1}L, X_i)$ with $w_1^{-1}L \neq w_2^{-1}L$ and $X = X_1 = X_2$. We may assume that $X = f(w_1) = f(w_2)$. Consider two strings $u, v \in w_1^{-1}L$. Since we may assume property 3., we know that either $u, v \in w_2^{-1}L$ or $u, v \notin w_2^{-1}L$. Since $u = \lambda \in w_1^{-1}L \cap w_2^{-1}L$, this means that $v \in w_1^{-1}L$ implies $v \in w_2^{-1}L$. Interchanging the roles of w_1 and w_2, we obtain $w_1^{-1}L = w_2^{-1}L$, a contradiction.

Secondly, consider two different states q_1, q_2 of A such that there is a third state q_3 with $\delta(q_1, a) = \delta(q_2, a) = q_3$. We have to treat the case that $q_i = (w_i^{-1}L, X_i)$ with $w_1^{-1}L \neq w_2^{-1}L$ and $X = X_1 = X_2$. We may assume that $X = f(w_1) = f(w_2)$. Since q_3 is not "useless", there exists a suffix s such that $w_1as, w_2as \in L$. Since f is a distinguishing function, $Y = f(w_1as) = f(w_2as)$ is the second (F-) component of the final state of A reached in this way. In particular, $as \in w_1^{-1}L \cap w_2^{-1}L$. This means that $v \in w_1^{-1}L$ implies $v \in w_2^{-1}L$. Interchanging the roles of w_1 and w_2, we obtain $w_1^{-1}L = w_2^{-1}L$, a contradiction.

'4. $\to$ 5.' is trivial.

'5. $\leftrightarrow$ 1.' is easy to see via the standard proof showing the equivalence of left-linear grammars and finite automata.

'4. $\to$ 6.' follows immediately by using Remark 2.

'6. $\to$ 5.': Let the regular language $L \subseteq T^*$ satisfy condition 6. Consider $A = A(L) \times A_f = (Q, T, \delta, q_0, Q_F)$. We have to verify condition 3. in the definition of f-distinguishing automata for A. If $u_1, u_2 \in L$ with $f(u_1) = f(u_2)$, then $u_1^{-1}L = u_2^{-1}L$. Hence, $\delta^*(q_0, u_1) = \delta^*(q_0, u_2)$, i.e., A satisfies condition 3*a*.

Consider two states $u_1^{-1}L$, $u_2^{-1}L$ of $A(L)$ with $f(u_1) = f(u_2)$. Assume that $(u_1a)^{-1}L = (u_2a)^{-1}L$ for some $a \in T$. Then, there is some $v' \in T^*$ such that $\{u_1av', u_2av'\} \subseteq L$. Hence, $\delta^*(q_0, u_1) = \delta^*(q_0, u_2)$, i.e., A satisfies condition 3*b*. □

Observe that the characterization theorem yields new characterizations for the special cases of both k-reversible and terminal distinguishable languages.

Lemma 6 *Let f be a distinguishing function. Any subautomaton of an f-distinguishable automaton is f-distinguishable.* □

Lemma 7 *Let f be a distinguishing function. The stripped version of an f-distinguishable automaton is isomorphic to the f-canonical automaton.*

Proof. Denote by $A' = (Q', T, \delta', q_0, Q'_F)$ the stripped subautomaton of some f-distinguishable automaton $A = (Q, T, \delta, q_0, Q_F)$. According to Lemma 6, A' is f-distinguishable. We have to show that, for all $q_1, q_2 \in Q'$ with $f(q_1) = f(q_2)$,

$$\{ v \in T^* \mid \delta^*(q_1, v) \in Q'_F \} = \{ v \in T^* \mid \delta^*(q_2, v) \in Q'_F \} \Rightarrow q_1 = q_2,$$

since then the mapping $q \mapsto (w^{-1}L(A), f(q))$ for some $w \in T^*$ with $\delta'^*(q_0, w) = q$ in A' will supply the required isomorphism.

Since A' is stripped, there exist strings $u_1, u_2, v \in T^*$ with $q_1 = \delta'^*(q_0, u_1)$, $q_2 = \delta'^*(q_0, u_2)$ and $\{u_1 v, u_2 v\} \subseteq L(A)$. By Remark 2, $q_1 = q_2$. □

4 Inferability

According to a theorem due to Angluin [12, Theorem 3.26], a language class $\mathcal{L}$ is inferable if any language $L \in \mathcal{L}$ has a characteristic sample, i.e., a finite subset $\chi(L) \subseteq L$ such that L is the smallest language from $\mathcal{L}$ containing $\chi(L)$.

For the language class f-DL and some language $L \in f$-DL, consider the corresponding f-canonical automaton $A(L, f) = (Q, T, \delta, q_0, Q_F)$ and define

$$\begin{aligned} \chi(L, f) &= \{ u(q)v(q) \mid q \in Q \} \\ &\cup \{ u(q)av(\delta(q, a)) \mid q \in Q, a \in T \}, \end{aligned}$$

where $u(q)$ and $v(q)$ are words of minimal length with $\delta^*(q_0, u(q)) = q$ and $\delta^*(q, v(q)) \in Q_F$.

Theorem 8 *For each distinguishing function f and each $L \in f$-DL, $\chi(L, f)$ is a characteristic sample of L.*

Proof. Consider an arbitrary language $L' \in f$-DL with $\chi(L, f) \subseteq L'$. Set $A = A(L, f) = (Q, T, \delta, q_0, Q_F)$ and $A' = A(L', f) = (Q', T, \delta', q'_0, Q'_F)$, cf. Theorem 5. We have to show $L \subseteq L'$. Therefore, we will prove:

(*) for all $w \in \mathrm{Pref}(L)$,

$$q = \delta^*(q_0, w) = (w^{-1}L', f(w)) = ((u(q))^{-1}L', f(u(q))).$$

(*) implies: If $w \in L$, i.e., $q_f = \delta^*(q_0, w)$ is final state of A, then, since $u(q_f) \in \chi(L, f) \subseteq L'$, $(u(q_f))^{-1}L'$ is an accepting state of the minimal automaton $A(L')$ of L'. This means that $(u(q_f)^{-1}L', f(u(q_f)))$ is an accepting state of A', i.e., $w \in L'$, since $f(w) = f(u(q))$. Hence, L is the smallest f-distinguishable language containing $\chi(L, f)$.

We prove (*) by induction over the length of the prefix w we have to consider.
If $|w| = 0$, then $w = u(q_0) = \lambda$. Hence, (*) is trivially verified.
We assume that (*) holds for all $w \in T^{\leq n}$, $n \geq 0$. We discuss the case where $wa \in T^{n+1}$, $w \in T^n$, $a \in T$ and $wa \in \mathrm{Pref}(L)$. Since $w \in \mathrm{Pref}(L)$, the induction hypothesis yields $(w^{-1}L', f(q)) = ((u(q))^{-1}L', f(q))$, where $q = \delta^*(q_0, w)$ and $f(w) = f(q) = f(u(q))$. Therefore, $(wa)^{-1}L' = (u(q)a)^{-1}L'$ and $f(wa) = f(u(q)a)$, since f is a distinguishing function. Consider $q' = \delta(q, a) = \delta^*(q_0, wa)$.

Since $\{u(q)av(q'), u(q')v(q')\} \subseteq \chi(L, f) \subseteq L'$ and $f(u(q)a) = f(u(q')) = f(wa)$, $\delta'^*(q_0', u(q)a) = \delta'^*(q_0', u(q'))$ due to Remark 2 and, hence, $(u(q'))^{-1}L' = (u(q)a)^{-1}L'$. The induction of (*) is finished. □

5 Inference algorithm

We sketch an algorithm which receives an input sample set $I_+ = \{w_1, \ldots, w_M\}$ (a finite subset of the language $L \in f$-DL to be identified) and finds the smallest language $L' \in f$-DL which contains I_+. In order to specify that algorithm more precisely, we need the following notions.

The *prefix tree acceptor* $PTA(I_+) = (Q, T, \delta, q_0, Q_F)$ of a finite sample set $I_+ = \{w_1, \ldots, w_M\} \subset T^*$ is a deterministic finite automaton which is defined as follows: $Q = \mathrm{Pref}(I_+)$, $q_0 = \lambda$, $Q_F = I_+$ and $\delta(v, a) = va$ for $va \in \mathrm{Pref}(I_+)$.

A simple merging state inference algorithm `f-Ident` for f-DL now starts with the automaton A_0 which is the stripped version of $PTA(I_+) \times A_f$[3] and merges two arbitrily chosen states q and q' which cause a conflict to the first or the third of the requirements for f-distinguishing automata. (One can show that the second requirement won't be violated ever when starting the merging process with A_0 which trivially satisfies that condition.) This yields an automaton A_1. Again, choose two conflicting states p, p' and merge them

[3]Of course, this automaton is equivalent to $PTA(I_+)$.

to obtain an automaton A_2 and so forth, until one comes to an automaton A_t which is f-distinguishable. In this way, we get a chain of automata $A_0, A_1, \ldots, A_t$. Speaking more formally, each automaton A_i in this chain can be interpreted as a quotient automaton of A_0 by the partition of the state set of A_0 induced by the corresponding merging operation. Observe that each A_i is stripped, since A_0 is stripped.

Completely analogous to [1, Lemma 1], one can prove:

Lemma 9 *Consider a distinguishing function f and some $L \in f$-DL. Let $I_+ \subseteq L \subseteq T^*$ be a finite sample. Let π be the partition of states of A_0 (the stripped version of $PTA(I_+) \times A_f$) given by: $(q_1, f(q_1))$, $(q_2, f(q_2))$ belong to the same block iff $q_1^{-1}L = q_2^{-1}L$ and $f(q_1) = f(q_2)$.*[4] *Then, the quotient automaton $\pi^{-1}A_0$ is isomorphic to a subautomaton of $A(L, f)$.* □

Theorem 10 *Let f be a distinguishing function. Consider a chain of automata $A_0, A_1, \ldots, A_t$ obtained by applying the sketched algorithm* `f-Ident` *on input sample I_+, where A_0 is the stripped version of $PTA(I_+) \times A_f$. Then, we have:*

1. *$L(A_0) \subseteq L(A_1) \subseteq \cdots \subseteq L(A_t)$.*
2. *A_t is f-distinguishable and stripped.*
3. *The partition π_t of the state set of A_0 corresponding to A_t is the finest partition π of the state set of A_0 such that the quotient automaton $\pi^{-1}A_0$ is f-distinguishable.*

Proof. 1. is clear, since `f-Ident` is a merging states algorithm.
2. follows almost by definition.
3. can be shown by induction, proving that each π_i corresponding to A_i refines π. Since this proof is analogous to [1, Lemma 25], we omit it; see also [5, Propriété 1.1]. □

Theorem 11 *In the notations of the previous theorem, $L(A_t)$ is the smallest f-distinguishable language containing I_+.*

Proof. The previous theorem states that $L(A_t) \in f$-DL and $I_+ = L(A_0) \subseteq L(A_t)$. Consider now an arbitary language L containing I_+. We consider

[4] Note that states of $PTA(I_+)$ are words over T.

the quotient automaton $\pi^{-1}A_0$ defined in Lemma 9. This Lemma shows that $L(\pi^{-1}A_0) \subseteq L = L(A(L,f))$. By Lemma 6, $\pi^{-1}A_0$ is f-distinguishable, because $A(L,f)$ is f-distinguishable due to Theorem 5. Theorem 10 yields that π_t refines π, so that $L(A_t) = L(\pi_t^{-1}A_0) \subseteq L(\pi^{-1}A_0) = L$. □

Theorem 12 *If $L \in f$-DL is enumerated to the algorithm f*`-Ident`*, it converges to the f-canonical automaton $A(L,f)$.*

Proof. At some point N of the enumeration process, the characteristic sample $\chi(L,f)$ will have been given to f`-Ident`. By combining Theorems 8 and 11, for all $n \geq N$, and all automata A_n output by f`-Ident`, we have $L(A_n) = L$. The argument of Theorem 11 shows that each A_n is isomorphic to a subautomaton of $A(L,f)$ generating $L = L(A(L,f))$. Since each A_n is stripped, it must be isomorphic to $A(L,f)$. □

We refrain from giving details of particular cases of f`-Ident`, since good implementations of f`-Ident` will depend on the choice of the distinguishing function f. We refer to [1, 8, 15] for several specific algorithms, including their time analysis. We only remark that the performance of the general algorithm f`-Ident` sketched above depends on the size of A_f (since the characteristic sample $\chi(L,f)$ we defined above depends on this size) and is in this sense "scalable", since "larger" A_f permit larger language families to be identified. More precisely:

Remark 3 Let f and g be distinguishing functions. If A_f is a homorphic image of A_g, then f-DL $\subseteq$ g-DL. □

As regards the time complexity, let us mention briefly that the f`-Ident` can be implemented to run in time $O(\alpha(|F|n)|F|n)$, where α is the inverse Ackermann function and n is the total length of all words in I_+ from language L, when L is the language presented to the learner for f-DL. This observation follows from the fact that f`-Ident` can be implemented similarly to the algorithm for 0-reversible languages exhibited by Angluin [1]. Moreover, her time analysis carries over to our situation. Observe that this leads to an $O((\alpha(|T|^k n)|T|^k n)$ algorithm for k-reversible languages, even if we output the deterministic minimal automaton as canonical object (instead of $A(L,f)$ as would be done by our algorithm), since $A(L)$ can be obtained by $A(L,f)$ by computationally simple projection. On the other hand, Angluin [1] presented an $O(kn^3)$ algorithm for the inference of k-reversible languages. When k is small compared to n (as it would be in realistic applications, where k could be considered even as fixed), our algorithm would turn out to be superior compared with Angluin's. Recall that this feature is prominent in so-called fixed-parameter algorithms, see [2, 3, 13].

We mention that f-`Ident` can be easily converted into an incremental algorithm, as sketched in the case of reversible languages in [1].

6 Discussion

We have proposed a large collection of families of languages, each of which is identifiable in the limit from positive samples, hence extending previous works. As the main technical contribution of the paper, we see the introduction of new canonical objects, namely the automata $A(L, f)$. This also simplifies correctness proofs of inference algorithms for k-reversible languages, $k > 0$, to some extent. It seems to be interesting to study these canonical automata also in the search-space framework of Dupont and Miclet [4, 6, 5].

We feel that deterministic methods (such as the one proposed in this paper) are quite important for practical applications, since they could be understood more precisely than mere heuristics, so that one can prove certain properties about the algorithms. Moreover, the approach of this paper allows one to make the bias (which each identification algorithm necessarily has) explicit and transparent to the user: The bias consists in (1) the restriction to regular languages and (2) the choice of a particular distinguishing function f.

We will provide a publicly accessible prototype learning algorithm for each f-DL in the near future. A user can then firstly look for an appropriate f by making learning experiments with typical languages he expects to be representative for the languages in his particular application. After this "bias training phase", the user may then use the such-chosen learning algorithm (or better, an improved implementation for the specific choice of f) for his actual application.

If the application suggests that the languages which are to be inferred are non-regular, methods such as those suggested in [14] can be transferred. This is done most easily by using the concept of *control languages* as undertaken in [7] or [16, Section 4] or by using the related concept of *permutations*, see [9].

Acknowledgments: We gratefully aknowledge discussions with J. Alber

and J. M. Sempere.

References

[1] D. Angluin. Inference of reversible languages. *Journal of the Association for Computing Machinery*, 29(3):741–765, 1982.

[2] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.

[3] R. G. Downey, M. R. Fellows, and U. Stege. Parameterized complexity: A framework for systematicall y confronting computational intractability. In *Contemporary Trends in Discrete Mathematics: From DIMACS and DIMATIA to the Future*, volume 49 of *AMS-DIMACS*, pages 49–99. AMS Press, 1999.

[4] P. Dupont. Incremental regular inference. In L. Miclet and C. de la Hieguera, editors, *Proceedings of the Third International Colloquium on Grammatical Inference (ICGI-96): Learning Syntax from Sentences*, volume 1147 of *LNCS/LNAI*, pages 222–237, Springer, 1996.

[5] P. Dupont and L. Miclet. Inférence grammaticale régulière: fondements théoriques et principaux algorithmes. Technical Report RR-3449, INRIA, 1998.

[6] P. Dupont, L. Miclet, and E. Vidal. What is the search space of the regular inference? In R. C. Carrasco and J. Oncina, editors, *Proceedings of the Second International Colloquium on Grammatical Inference (ICGI-94): Grammatical Inference and Applications*, volume 862 of *LNCS/LNAI*, pages 25–37, Springer, 1994.

[7] H. Fernau. Learning of terminal distinguishable languages. Technical Report WSI–99–23, Universität Tübingen (Germany), Wilhelm-Schickard-Institut für Informatik, 1999. Short version published in the proceedings of AMAI 2000, see `http://rutcor.rutgers.edu/~amai/AcceptedCont.htm`.

[8] H. Fernau. k-gram extensions of terminal distinguishable languages. In *Proc. International Conference on Pattern Recognition*. IEEE/IAPR, 2000. To appear.

[9] H. Fernau and J. M. Sempere. Permutations and control sets for learning non-regular language families. In *Proc. International Conference on Grammatical Inference*. Springer, 2000. To appear.

[10] E. M. Gold. Language identification in the limit. *Information and Control (now Information and Computation)*, 10:447–474, 1967.

[11] J. Gregor. Data-driven inductive inference of finite-state automata. *International Journal of Pattern Recognition and Artificial Intelligence*, 8(1):305–322, 1994.

[12] S. Jain, D. Osherson, J. S. Royer, and A. Sharma. *Systems That Learn.* MIT Press, 2nd edition, 1999.

[13] R. Niedermeier. Some prospects for efficent fixed parameter algorithms (invited paper). In B. Rovan, editor, *SOFSEM'98*, volume 1521 of *LNCS*, pages 168–185, Springer, 1998.

[14] V. Radhakrishnan. *Grammatical Inference from Positive Data: An Effective Integrated Approach.* PhD thesis, Department of Computer Science and Engineering, Indian Institute of Technology, Bombay (India), 1987.

[15] V. Radhakrishnan and G. Nagaraja. Inference of regular grammars via skeletons. *IEEE Transactions on Systems, Man and Cybernetics*, 17(6):982–992, 1987.

[16] Y. Takada. A hierarchy of language families learnable by regular language learning. *Information and Computation*, 123:138–145, 1995.

[17] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.