Synthesizing Enumeration Techniques For Language Learning

Ganesh R. Baliga
Computer Science Department
Rowan College of New Jersey
Mullica Hill, NJ 08024, USA
(Email: baliga@gboro.rowan.edu)

John Case
Department of CIS
University of Delaware
Newark, DE 19716, USA
(Email: case@cis.udel.edu)

Sanjay Jain
Department of ISCS
National University of Singapore
Singapore
(Email: sanjay@iscs.nus.sg)

Abstract

This paper provides positive and negative results on algorithmically synthesizing, from grammars and from decision procedures for classes of languages, learning machines for identifying, from positive data, grammars for the languages in those classes. In the process, the uniformly decidable classes of recursive languages that can be behaviorally correctly identified from positive data are surprisingly characterized by Angluin's 1980 Condition 2 (the *subset principle* for preventing overgeneralization).

1 Introduction

In the context of learning programs in the limit for functions [KW80, AS83, OSW86b], a variety of *enumeration techniques* [Gol67, BB75, Wie90, Ful90b] are important and ubiquitous. Here is the archetypal case. Suppose one has an r.e. set P of programs for computing total functions. One proceeds as follows. At each point that one is given data about an input function, one conjectures the *first* program, if any, listed in P that is correct on the data seen through that point. Clearly for functions f computed by some program in P, this procedure will eventually lock onto a fixed conjecture correct for f. For example, if P is the set of Meyer-Ritchie *loop-programs* [MR67, RC94], then the enumeration technique we just described learns, in the limit, a correct final program for each primitive recursive function. More generally, enumeration techniques involve some process of matching and elimination based on an initial r.e. set of programs for the objects to be learned. Enumeration techniques for function learning trivially have the crucial constructivity property that there *is* an algorithm for translating an enumeration procedure for P into a corresponding enumeration technique which learns, in the limit, a correct program for each function computed by a program in P.

Lets see what happens with enumeration techniques and their constructivity in the context of learning (formal) grammars for (formal) languages from positive information about those languages [OSW86b]. The present paper was inspired by the following amazingly negative result from [OSW88]. They show: there is *no* algorithm for translating any *pair* of grammars (g, g') into a learning procedure which, from positive data about any one of the languages generated by g and g', finds in the limit, for this language, a correct grammar! By the way, the problem is not that correct programs for finite classes of languages can't be learned in the limit; they can [OSW86b], and by an obvious enumeration technique. The problem is how to pass algorithmically from a list of grammars to a machine which so learns the corresponding languages.

We consider herein the extent to which one can assuage the negative result in the just previous paragraph by, among other means, loosening the requirements for successful learning. We let $\mathrm{card}(S)$ denote the cardinality of a set S. We show, for example, (Theorem 1 in Section 3 below) that there *is* an algorithm for translating any enumeration procedure for a *finite* set P of grammars into a learning procedure $\mathbf{M}_P$ which, given any listing of a language L generated by grammars in P, eventually converges to outputting over and over no more than $\mathrm{card}(P)$ grammars each of which is correct for L. The requirement for successful learning is loosened *from* requiring $\mathbf{M}_P$ to output eventually a single correct grammar for L (this is called **TxtEx**-*identification*) *to* merely requiring $\mathbf{M}_P$ to output eventually $\leq \mathrm{card}(P)$ grammars correct for L.[1] Furthermore, $\mathbf{M}_P$ does involve an enumeration technique, a procedure which does matching and elimination based on the grammars in P. Interestingly, too, $\mathbf{M}_P$, in this case, outputs conjectures

[1]The criterion requiring a machine, on positive data for a language L, to output eventually no more than n distinct grammars each of which is correct is called $\mathbf{TxtFex}_n$-*identification* [Cas88, Cas92]. $\mathbf{TxtFex}_1$-identification is just **TxtEx**-identification, but one can learn strictly larger classes of languages with $\mathbf{TxtFex}_{n+1}$-identification than with $\mathbf{TxtFex}_n$-identification [Cas88, Cas92].

from the "hypothesis space" P itself [LZ93].

Suppose x is a procedure for enumerating an r.e. (possibly infinite) set of grammars P. Let $\mathcal{C}_x$ be the set of languages generated by the grammars in P. Some interesting $\mathcal{C}_x$'s are learnable [Ang80a, Ang80b, Ang82] and some are not [Gol67]. In Section 3.1 we explore the problem of synthesizing learning machines for *learnable* $\mathcal{C}_x$'s from the corresponding x's.

One shot language identification (called $\mathbf{TxtEx}_0$-identification below) is just the important case of **TxtEx**-identification where the learning procedure makes but one conjecture (which must, then, be correct). The proof of Theorem 3 below (in Section 3.1) presents an algorithm for transforming any x such that $\mathcal{C}_x$ is $\mathbf{TxtEx}_0$-identifiable into a (behaviorally correct [CL82, OW82a][2] or **TxtBc**) learning procedure which, on any listing of a language in $\mathcal{C}_x$, eventually converges to an infinite sequence of grammars each of which generates the input language.[3]

For this, as well as our other results providing the synthesis of a learning machine, each synthesized learning machine can be construed as implementing a (perhaps complicated) enumeration technique; however, of necessity, in most cases the conjectures of the synthesized machines go beyond the original hypothesis space [LZ93].

Regarding the positive results about $\mathcal{C}_x$'s, we also present corresponding lower bound results showing, in many cases, the positive results to be best possible. For example, Theorem 4 below shows the necessity of the cost, from Theorem 3, of passing from *no* mind changes for the input classes to *infinitely* many in the *synthesized* learning machines.

There has been considerable interest in the computational learning theory community in the learnability, from positive data, of *uniformly decidable classes of recursive languages*. This begins with [Gol67], but especially spans [Ang80b] through [ZL95]. In the context of the present paper, one might hope to obtain synthesized learning machines with better mind change complexity if one input *decision procedures* in place of grammars for the languages to be learned from positive data. In Section 3.2 below, we see that this is indeed the case. For example, the proof of Theorem 9 below (in Section 3.2) presents an algorithm for transforming any uniform decision procedure for a class of recursive languages $\mathcal{L}$ that is $\mathbf{TxtEx}_0$-identifiable into a learning procedure which **TxtEx**-identifies $\mathcal{L}$. Theorem 10 shows the necessity of the cost, from Theorem 9, of passing from *no* mind changes for the input classes to *finitely* many in the *synthesized* learning machines.

Suppose $a \in N \cup \{*\}$, where $*$ is not a natural number. $\mathbf{TxtBc}^a$*-identification* [CL82] is just like **TxtBc**-identification except that, in the final sequence of "correct" programs, each program is allowed to make up to a mistakes (where $a = *$ denotes the unbounded, finite case). The proof of the Main Theorem of the present paper (Theorem 11) provides an algorithm for transforming any uniform decision procedure for a class of recursive languages $\mathcal{L}$ that is $\mathbf{TxtBc}^a$-identifiable into a learning procedure which $\mathbf{TxtBc}^a$-identifies $\mathcal{L}$. Hence, in passing from learnable uniformly decidable classes to algorithmically synthesized learning machines for them, we get a fixed point, for each a, at $\mathbf{TxtBc}^a$-identification! The last theorem of Section 3.2 (Theorem 12) shows that the cost of passing from even *one* mind change in the input classes to *infinitely* many in the *synthesized* learning machines is necessary.

The Main Corollaries (Corollaries 2 and 3) of the proof of the Main Theorem (Theorem 11) *completely characterize* the uniformly decidable classes in $\mathbf{TxtBc}^a$. In fact, for the $a = 0$ case, surprisingly, our characterizing condition is Angluin's Condition 2 from [Ang80b]! This condition is the *subset principle*, a necessary condition preventing overgeneralization in learning from positive data [Ang80b, Ber85, Cas92].[4] Her Condition 1, also from [Ang80b], was an constructive version of her Condition 2, providing her characterization of uniformly decidable classes in **TxtEx**.[5] Hence, a uniformly decidable class is in (**TxtBc** $-$ **TxtEx**) $\Leftrightarrow$ it satisfies Angluin's Condition 2 but not her Condition 1.

Finally, the discussion following the proof of Theorem 12 clarifies the connection between our learning machine synthesizing algorithm from the proof of Theorem 11 and one implicit in Angluin's proof [Ang80b] of her characterization theorem.

2 Preliminaries

2.1 Notation

N denotes the set of natural numbers, $\{0, 1, 2, 3, \ldots\}$. Unless otherwise specified, $e, i, j, k, m, n, p, s, w, x, y, z$, with or without decorations[6], range over N. $*$ denotes a non-member of N and is assumed to satisfy $(\forall n)[n < * < \infty]$. Furthermore, $2* \stackrel{\text{def}}{=} *$. a, b and c with or without decorations, ranges over $N \cup \{*\}$. $\emptyset$ denotes the empty set. $\subseteq$ denotes subset. $\subset$ denotes proper subset. $\supseteq$ denotes superset. $\supset$ denotes proper superset. P and S, with or without decorations, range over sets. We sometimes write $\mathrm{card}(S) \leq *$ to mean S is finite. We use $S_1 \mathbf{\Delta} S_2$ to denote the symmetric difference of the sets S_1 and S_2. $S_1 =^a S_2$ means that $\mathrm{card}(\{x \mid x \in S_1 \mathbf{\Delta} S_2\}) \leq a$. **ODD** denotes the set of odd natural numbers. **EVEN** denotes the set of even natural numbers.

[2] See also [Bar74, CS78, CS83].

[3] One can learn larger classes of languages with **TxtBc**-identification than with $\mathbf{TxtFex}_n$-identification for any n [Cas88, Cas92]; of course this is at a cost of outputting infinitely many distinct grammars in the limit.

[4] See [KLHM93, Wex93] for discussion regarding the possible connection between this subset principle and a more traditionally linguistically oriented one in [MW87].

[5] We restate these conditions in Section 3.2 below.

[6] Decorations are subscripts, superscripts and the like.

$\max(\cdot), \min(\cdot)$ denote the maximum and minimum of a set, respectively, where $\max(\emptyset) = 0$ and $\min(\emptyset)$ is undefined. Fix a *canonical indexing* of the finite sets 1-1 onto N [Rog67]. The $\min(\cdot)$ of a collection of finite sets is, then, the finite set in the collection with minimal canonical index. Also, when we compare finite sets by $<$ we are comparing their corresponding canonical indices.

We use the symbol $\downarrow$ to denote that a computation converges. f, g and h with or without decorations range over *total* functions with arguments and values from N. $\langle \cdot, \cdot \rangle$ stands for an arbitrary, computable, one-to-one encoding of all pairs of natural numbers onto N [Rog67].

φ denotes a fixed *acceptable* programming system for the partial computable functions: $N \to N$ [Rog58, MY78, Roy87]. φ_i denotes the partial computable function computed by program i in the φ-system. The set of all (total) recursive functions of one variable is denoted by $\mathcal{R}$. $\mathcal{R}_{0,1}$ denotes the class of all (total) recursive 0-1 valued functions.

W_i denotes domain(φ_i). W_i is, then, the r.e. set/language ($\subseteq N$) accepted (or equivalently, generated [HU79]) by the φ-program i. $W_i^s \stackrel{\text{def}}{=} \{x \leq s \mid x$ appears in W_i in $\leq s$ steps$\}$. For language $L \subseteq N$, $L[x]$ denotes $\{w \leq x \mid x \in L\}$, and we use χ_L to denote the characteristic function of L; $\overline{L}$ denotes complement of L. $\mathcal{L}$, with or without decorations, ranges over set of subsets of the r.e. sets.

We sometimes consider partial recursive functions with two arguments in the φ system. In such cases we implicitly assume that $\langle \cdot, \cdot \rangle$ is used to code the arguments, so, for example, $\varphi_i(x, y)$ stands for $\varphi_i(\langle x, y \rangle)$.

The quantifiers '$\forall^\infty$', and '$\exists^\infty$' essentially from [Blu67], mean 'for all but finitely many' and 'there exist infinitely many', respectively.

$f : N \to N$ is *limiting recursive* $\stackrel{\text{def}}{\Leftrightarrow}$ ($\exists$ recursive $g : (N \times N) \to N)(\forall x)[f(x) = \lim_{t \to \infty} g(x, t)]$. Intuitively, $g(x, t)$ is the output at discrete time t of a mind changing algorithm for f (acting on input x); hence, for f limiting recursive as just above, for all x, for all but finitely many times t, the output of the mind changing algorithm on input x is $f(x)$. It is easy to show that there is a limiting recursive function h such that ($\forall$ recursive $g)(\forall^\infty x)[h(x) > g(x)]$. Hence, the limiting recursive functions go way beyond the recursive ones; in fact, they have been known since Post ([Sha71]) to characterize the functions recursive in an oracle for the halting problem. The set of all (total) limiting recursive functions of one variable is denoted by $\mathcal{LR}$.

We sometimes use the symbol '$\neg$' for *negation*.

2.2 Learning Machines

We now consider language learning machines. We first introduce a notion that facilitates discussion about elements of a language being fed to a learning machine. A *sequence* σ is a mapping from an initial segment of N into $(N \cup \{\#\})$. The *content* of a sequence σ, denoted content(σ), is the set of natural numbers in the range of σ. The *length* of σ, denoted by $|\sigma|$, is the number of elements in σ. Λ denotes an empty sequence. Intuitively, #'s represent pauses in the presentation of data. We let σ and τ, with or without decorations, range over finite sequences. SEQ denotes the set of all finite sequences. The set of all finite sequences of natural numbers and #'s, SEQ, can be coded onto N. This latter fact will be used implicitly in some of our proofs.

A *language learning machine* is an algorithmic device which computes a mapping from SEQ into $N \cup \{?\}$. Intuitively the outputted ?s represent the machine not yet committing to an output *program*. The reason we want the ?s is so we can avoid biasing the number of *program mind changes* before a learning machine converges: if we allow initial outputs of ?s before, if ever, the first program is output, then we can learn more things within n mind changes than if we had to begin with a program (numerical) output. In this paper we assume, without loss of generality, that for all $\sigma \subseteq \tau$, $[\mathbf{M}(\sigma) \neq ?] \Rightarrow [\mathbf{M}(\tau) \neq ?]$.

$\mathbf{M}$ ranges over language learning machines. Lemma 4.2.2B in [OSW86b] easily generalizes to cover all learning criteria considered in this paper thereby providing a recursive enumeration $\mathbf{M}_0, \mathbf{M}_1, \ldots$ of (total) language learning machines such that, for any inference criteria $\mathbf{I}$, every $\mathcal{L} \in \mathbf{I}$ is $\mathbf{I}$-identified by some machine in this enumeration. Moreover, this enumeration satisfies the s-m-n property, i.e., given a description, algorithmic in x, of a behavior of a machine $\mathbf{M}$, one can algorithmically find a machine $\mathbf{M}_{f(x)}$ whose learning behavior is identical to that of $\mathbf{M}$. In the following we fix an arbitrary such enumeration $\mathbf{M}_0, \mathbf{M}_1, \ldots$.

2.3 Fundamental Language Identification Paradigms

A *text* T for a language L is a mapping from N into $(N \cup \{\#\})$ such that L is the set of natural numbers in the range of T. The *content* of a text T, denoted content(T), is the set of natural numbers in the range of T.

Intuitively, a text for a language is an enumeration or sequential presentation of all the objects in the language with the #'s representing pauses in the listing or presentation of such objects. For example, the only text for the empty language is just an infinite sequence of #'s.

We let T, with or without superscripts, range over texts. $T[n]$ denotes the finite initial sequence of T with length n. Hence, domain$(T[n]) = \{x \mid x < n\}$.

2.3.1 Explanatory Language Identification

Suppose $\mathbf{M}$ is a learning machine and T is a text. $\mathbf{M}(T)\!\downarrow$ (read: $\mathbf{M}(T)$ *converges*) $\stackrel{\text{def}}{\Leftrightarrow}$ $(\exists i)(\forall^\infty n)$ $[\mathbf{M}(T[n]) = i]$. If $\mathbf{M}(T)\!\downarrow$, then $\mathbf{M}(T)$ is defined $=$ the unique i such that $(\forall^\infty n)[\mathbf{M}(T[n]) = i]$.

We now introduce criteria for a learning machine to be considered *successful* on languages.

Definition 1 Recall that a and b range over $N \cup \{*\}$.

(1) $\mathbf{M}$ $\mathbf{TxtEx}^a$-*identifies* L (written: $L \in \mathbf{TxtEx}^a(\mathbf{M})$) $\stackrel{\mathrm{def}}{\Leftrightarrow}$ ($\forall$ texts T for L)$(\exists i \mid W_i =^a L)[\mathbf{M}(T)\downarrow = i]$.

(2) $\mathbf{M}$ $\mathbf{TxtEx}^a_b$-*identifies* $L \stackrel{\mathrm{def}}{\Leftrightarrow}$ (written:$L \in \mathbf{TxtEx}^a_b(\mathbf{M})$) $[L \in \mathbf{TxtEx}^a(\mathbf{M}) \;\wedge\;$ ($\forall$ texts T for L)$[\mathrm{card}(\{x \mid ? \neq \mathbf{M}(T[x]) \;\wedge\; \mathbf{M}(T[x]) \neq \mathbf{M}(T[x+1])\}) \leq b]]$.

(3) $\mathbf{TxtEx}^a_b = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtEx}^a_b(\mathbf{M})]\}$.

Gold [Gol67] introduced the criteria we call $\mathbf{TxtEx}^0_*$. The generalization to the $a > 0$ cases in Definition 1 is motivated by the observation that humans rarely learn a language perfectly, where the a represents an upper bound on the numer of *anomalies* permitted in final grammars. The $a > 0$ case is from [CL82], but [OW82a], independently, introduced the $a = *$ case. For these and the other success criteria of this paper, we have that tolerating more anomalies leads to being able to learn larger classes of languages [CL82, Cas88, Cas92, BC93]. Gold's model of language learning from text (positive information) and by machine [Gol67] has been very influential in contemporary theories of natural language and in mathematical work motivated by its possible connection to human language learning (see, for example, [Pin79, WC80, Wex82, OSW82, OSW84, Ber85, Gle86, Cas86, OSW86a, OSW86b, Ful85, Ful90a, Kir92, BCJ95]).

We sometimes write $\mathbf{TxtEx}$ for $\mathbf{TxtEx}^0_*$ and $\mathbf{TxtEx}^a$ for $\mathbf{TxtEx}^a_*$.

2.3.2 Vacillatory and Behaviorally Correct Language Identification

Definition 2 We say $\mathbf{M}(T)$ *finitely-converges* (written: $\mathbf{M}(T)\Downarrow$) $\stackrel{\mathrm{def}}{\Leftrightarrow}$ $\{\mathbf{M}(\tau) \mid \tau \subset T\}$ is a finite subset of N. If $\mathbf{M}(T)\Downarrow$, then $\mathbf{M}(T)$ is defined $= \{p \mid (\exists^\infty \tau \subset T)[\mathbf{M}(\tau) = p]\}$; otherwise, $\mathbf{M}(T)$ is undefined.

Definition 3 $\mathbf{M}$, $\mathbf{TxtFex}^a_b$-*identifies* an r.e. language L (written:$L \in \mathbf{TxtFex}^a_b(\mathbf{M})$) $\stackrel{\mathrm{def}}{\Leftrightarrow}$ ($\forall$ texts T for L)$[\mathbf{M}(T)\Downarrow =$ a set of cardinality $\leq b$ and $(\forall p \in \mathbf{M}(T))[W_p =^a L]]$. $\mathbf{TxtFex}^a_b$ denotes the class of all *classes* $\mathcal{L}$ of languages such that some learning machine $\mathbf{TxtFex}^a_b$-identifies each language in $\mathcal{L}$.

In $\mathbf{TxtFex}^a_b$-identification the b is a "bound" on the number of final grammars and the a a "bound" on the number of anomalies allowed in these final grammars. In general a "bound" of $*$ just means *un*bounded, but *finite*. Intuitively, $\mathcal{L} \in \mathbf{TxtFex}^a_b \stackrel{\mathrm{def}}{\Leftrightarrow}$ there is an algorithmic procedure $\mathbf{p}$ such that, if $\mathbf{p}$ is given any listing of any language $L \in \mathcal{L}$, it outputs a sequence of grammars *converging* in a non-empty *set* of no more than b grammars, and each of these grammars makes no more than a mistakes in generating L.

N.B. The b in $\mathbf{TxtEx}^a_b$ has a totally different meaning form the b in $\mathbf{TxtFex}^a_b$; the former is a bound on mind changes for convergence to a single final program, the latter is a bound on the number of different programs an associated machine eventually vacillates between in the limit.

We sometimes write $\mathbf{TxtFex}_b$ for $\mathbf{TxtFex}^0_b$.

$\mathbf{TxtFex}^a_1$-identification is clearly equivalent to $\mathbf{TxtEx}^a_*$. Osherson and Weinstein [OW82a] were the first to define the notions of $\mathbf{TxtFex}^0_*$ and $\mathbf{TxtFex}^*_*$, and the other cases of $\mathbf{TxtFex}^a_b$-identification are from [Cas86, Cas88, Cas92].

Definition 4 (1) $\mathbf{M}$ $\mathbf{TxtBc}^a$-*identifies* L (written: $L \in \mathbf{TxtBc}^a(\mathbf{M})$) $\stackrel{\mathrm{def}}{\Leftrightarrow}$ ($\forall$ texts T for L)$(\forall^\infty n)[W_{\mathbf{M}(T[n])} =^a L]$.

(2) $\mathbf{TxtBc}^a = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtBc}^a(\mathbf{M})]\}$.

In a completely computable universe all texts must be recursive (synonym: computable). This motivates the following

Definition 5 (1) $\mathbf{M}$ $\mathbf{RecTxtBc}^a$-*identifies* L (written: $L \in \mathbf{RecTxtBc}^a(\mathbf{M})$) $\stackrel{\mathrm{def}}{\Leftrightarrow}$ ($\forall$ *recursive* texts T for L)$(\forall^\infty n)[W_{\mathbf{M}(T[n])} =^a L]$.

(2) $\mathbf{RecTxtBc}^a = \{\mathcal{L} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{RecTxtBc}^a(\mathbf{M})]\}$.

Definition 4 is from [CL82]. The $a \in \{0, *\}$ cases were independently introduced in [OW82a, OW82b]. $\mathbf{RecTxtBc}^a \neq \mathbf{TxtBc}^a$ [CL82, Fre85]; however, the restriction to recursive texts doesn't affect learning power for $\mathbf{TxtFex}^a_b$-identification [Cas88, Cas92].

We sometimes write $\mathbf{TxtBc}$ for $\mathbf{TxtBc}^0$, etc.

Definition 6 σ is called a $\mathbf{TxtEx}^a$*-locking sequence* for $\mathbf{M}$ on L, $\stackrel{\mathrm{def}}{\Leftrightarrow}$ $\mathrm{content}(\sigma) \subseteq L$, $W_{\mathbf{M}(\sigma)} =^a L$, and $(\forall \tau \mid \sigma \subseteq \tau \;\wedge\; \mathrm{content}(\tau) \subseteq L)[\mathbf{M}(\sigma) = \mathbf{M}(\tau)]$.

σ is called a $\mathbf{TxtBc}^a$*-locking sequence* for $\mathbf{M}$ on L, iff $\mathrm{content}(\sigma) \subseteq L$, and $(\forall \tau \mid \sigma \subseteq \tau \;\wedge\; \mathrm{content}(\tau) \subseteq L)[W_{\mathbf{M}(\tau)} =^a L]$.

It can be shown ([BB75, OW82a, OSW86b, Cas92]) that if $\mathbf{M}$ $\mathbf{TxtEx}^a$-identifies L, then there exists a $\mathbf{TxtEx}^a$-locking sequence for $\mathbf{M}$ on L. Similarly it can be shown that, if $\mathbf{M}$ $\mathbf{TxtBc}^a$-identifies L, then there exists a $\mathbf{TxtBc}^a$-locking sequence for $\mathbf{M}$ on L.

3 Results

In Section 3.1 we present our positive and negative results on synthesizing learning machines from grammars.

In Section 3.2 we present such results on synthesizing learning machines from decision procedures.

3.1 Synthesizing From Grammars

Definition 7 $\mathcal{C}_x \stackrel{\text{def}}{=} \{W_p \mid p \in W_x\}$.

One can think of the x in '$\mathcal{C}_x$' as representing a *uniform* grammar for generating (accepting) the languages in $\mathcal{C}_x$.

The following theorem removes some of the sting from the negative result ([OSW88]) motivating the present paper. It does this by relaxation of the criterion for successful learning.

Theorem 1 $(\exists$ *recursive* $f)(\forall x \mid W_x$ *is finite*$)$ $[\mathcal{C}_x \subseteq \mathbf{TxtFex}_{\mathrm{card}(W_x)}(\mathbf{M}_{f(x)})]$.[7]

Proof. Let $\mathrm{match}(i, T[n]) = \min(\{n\} \cup (W_i^n \boldsymbol{\Delta} \mathrm{content}(T[n])))$. Intuitively match finds the minimum point of disagreement between W_i and $T[n]$. Note that [i is a grammar for content(T) $\Leftrightarrow$ $\lim_{n\to\infty} \mathrm{match}(i, T[n]) = \infty$]. Let

$\mathbf{M}_{f(x)}(T[n]) = i \in W_x^n$ which maximizes $\mathrm{match}(i, T[n])$.

Fix x such that W_x is finite. Let $L \in \mathcal{C}_x$ and T be a text for L. Since, W_x is finite, it is easy to verify that, for all but finitely many n, $\mathbf{M}_{f(x)}(T[n])$ is in W_x and is a grammar for $L = \mathrm{content}(T)$. The theorem follows. ∎

As a corollary of the proof of the above theorem, we have

Corollary 1 ([OSW88]) $(\exists$ recursive $f)(\forall x \mid W_x$ is finite $\wedge$ $(\forall$ distinct $i, j \in W_x)[W_i \neq W_j])[\mathcal{C}_x \subseteq \mathbf{TxtEx}(\mathbf{M}_{f(x)})]$.

That the bound in Theorem 1 above is tight is witnessed by the following strong lower bound

Theorem 2 *For all* $n \geq 1$, $\neg(\exists f \in \mathcal{LR})(\forall i_0, i_1, \ldots, i_n)$ $[\{W_{i_0}, W_{i_1}, \ldots W_{i_n}\} \subseteq \mathbf{TxtFex}_n^*(\mathbf{M}_{f(i_0,i_1,\ldots,i_n)})]$.

The $n = 1$ case of Theorem 2 just above, with 'limiting recursive' replaced by 'recursive' is just the negative result from [OSW88] that inspired the present paper, but, of course, Theorem 2 is stronger and more general. Theorem 2 follows by a direct n-ary recursion theorem argument and also quickly but indirectly from the fact from [Cas88, Cas92] that $(\mathbf{TxtFex}_{n+1} - \mathbf{TxtFex}_n^*) \neq \emptyset$.

The following theorem implies that it is possible to synthesize algorithmically, from uniform grammars, behaviorally correct learners for classes which can be learned in one-shot (i.e., without any mind changes). As Theorem 4 further below shows, the cost of passing from *no* mind changes in the input classes to *infinitely* many in the *synthesized* learning machines is in some cases necessary (but see the second paragraph in Section 4 below).

Recall that $2* \stackrel{\text{def}}{=} *$.

Theorem 3 $(\exists$ *recursive* $f)(\forall x \mid \mathcal{C}_x \in \mathbf{TxtEx}_0^a)[\mathcal{C}_x \subseteq \mathbf{TxtBc}^{2a}(\mathbf{M}_{f(x)})]$.

Proof. Fix x such that $\mathcal{C}_x \in \mathbf{TxtEx}_0^a$.

Claim 1 $(\forall L \in \mathcal{C}_x)(\exists S \subseteq L)(\forall L' \mid S \subseteq L')[L' \in \mathcal{C}_x \Rightarrow L' =^{2a} L]$.

Proof. Suppose $\mathbf{M}$ $\mathbf{TxtEx}_0^a$–identifies $\mathcal{C}_x$. Suppose by way of contradiction that $L \in \mathcal{C}_x$ is such that $(\forall S \subseteq L)(\exists L' \mid S \subseteq L')[(L' \in \mathcal{C}_x) \ \wedge \ (L' \neq^{2a} L)]$. Let σ be a $\mathbf{TxtEx}^a$ locking sequence for $\mathbf{M}$ on L [OSW86b]. Let $L' \in \mathcal{C}_x$ be such that $L' \supseteq \mathrm{content}(\sigma)$ and $L' \neq^{2a} L$. Since $W_{\mathbf{M}(\sigma)} =^a L$, it follows that $W_{\mathbf{M}(\sigma)} \neq^a L'$. Therefore, $L' \notin \mathbf{TxtEx}_0^a(\mathbf{M})$, a contradiction. □(Claim 1)

Let g be a recursive function such that for all x and σ, $W_{g(x,\sigma)} = W_y$, where $y \in W_x$ is an integer satisfying $\mathrm{content}(\sigma) \subseteq W_y$, if such an integer exists; otherwise, $W_{g(x,\sigma)} = \emptyset$. Let f be a recursive function such that for all x, $\mathbf{M}_{f(x)}(\sigma) = g(x, \sigma)$.

Fix $L \in \mathcal{C}_x$ and a text T for L. Let $S \subseteq L$ witness that L satisfies the above claim. Let n_0 be such that $\mathrm{content}(T[n_0]) \supseteq S$. It follows using Claim 1 that for all $n \geq n_0$, $W_{\mathbf{M}_{f(x)}(T[n])} =^{2a} L$. ∎ (Theorem 3)

It is open whether, for $a > 0$, the $2a$ in Theorem 3 just above is also a lower bound; however, we do know the following

Theorem 4 $\neg(\exists f \in \mathcal{LR})(\forall x \mid \mathcal{C}_x \in \mathbf{TxtEx}_0)[\mathcal{C}_x \subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(x)})]$.

Proof. We prove here the following restricted version of the theorem.

$\neg(\exists f \in \mathcal{R})(\forall x \mid \mathcal{C}_x \in \mathbf{TxtEx}_0)[\mathcal{C}_x \subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(x)})]$.

Suppose by way of contradiction that $f \in \mathcal{R}$ is such that $(\forall x \mid \mathcal{C}_x \in \mathbf{TxtEx}_0)[\mathcal{C}_x \subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(x)})]$. By the Operator Recursion Theorem [Cas74, Cas94], there exists a recursive function p such that the languages $W_{p(i)}$, $i \geq 0$, are defined in stages as follows. Initially, the $W_{p(i)}$'s are empty and σ is the empty sequence. Go to stage 1.

Stage s

1. For each i such that $1 \leq i \leq s$, enumerate $p(i)$ into $W_{p(0)}$ and let $W_{p(i)} = \mathrm{content}(\sigma)$.
2. Let $x_s = \min(\{x \mid \mathrm{content}(\sigma) \cap \{\langle x, i\rangle \mid i \geq 0\} = \emptyset\})$.

[7] Our proof actually demonstrates that $(\exists$ recursive $f)(\forall n, x \mid \max(\{\mathrm{card}(\{i \mid W_i = L\}) \mid L \in \mathcal{C}_x\}) = n)[\mathcal{C}_x \subseteq \mathbf{TxtFex}_n(\mathbf{M}_{f(x)})]$.
This nicely generalizes the special case from [OSW88] presented as Corollary 1 below.

3. Dovetail steps 4 and 5. If and when step 4 succeeds, go to step 6.
4. Search for $\sigma' \supset \sigma$ such that $\mathrm{card}(\{\mathbf{M}_{f(p(0))}(\sigma'') \mid \sigma'' \subseteq \sigma'\}) \geq s$.
5. For each i such that $1 \leq i \leq s$, enumerate more and more elements for the $\{\langle x_s + i - 1, j\rangle \mid j \geq 0\}$ into $W_{p(i)}$.
6. Let $\sigma \supset \sigma'$ be the least sequence such that $\mathrm{content}(\sigma) \supseteq \{j \mid j \leq 1 + \max(\bigcup_{1\leq i\leq s} W_{p(i)}$ enumerated until now$)\}$.
   Go to stage $s+1$.

End stage s

We consider 2 cases.

Case 1: All stages terminate.

In this case, it can be verified that $W_{p(0)} = \{p(i) \mid i \geq 1\}$ and for all $i \geq 1$, $W_{p(i)} = N$. Thus, $\mathcal{C}_{p(0)} \in \mathbf{TxtEx}_0$. But step 4 succeeds in every stage, hence $\mathcal{C}_{p(0)} \not\subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(p(0))})$.

Case 2: Stage s starts but does not terminate.

In this case, $W_{p(0)} = \{p(i) \mid 1 \leq i \leq s\}$. Let σ_s be the value of σ just before the start of stage s. From step 5 it is clear that for all i such that $1 \leq i \leq s$, $W_{p(i)} =^* \{\langle x_s + i - 1, j\rangle \mid j \geq 0\}$. Hence $(\forall i \leq s, j \leq s \mid i \neq j)[W_{p(i)} \neq^* W_{p(j)}]$. Thus, since step 4 did not succeed in stage s, $\mathcal{C}_{p(0)} \not\subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(p(0))})$. Furthermore, it is clear that $\mathcal{C}_{p(0)}$ is finite and the languages in it are pairwise incomparable (by $\subset$). Hence, $\mathcal{C}_{p(0)} \in \mathbf{TxtEx}_0$. ∎

It is interesting to generalize Theorem 3 above about synthesis from one-shot learnable $\mathcal{C}_x$'s to the case of *two*-shot learnable $\mathcal{C}_x$'s. The next two Theorems (Theorems 5 and 6) provide our best to date upper and lower bounds, respectively, for the two-shot cases. Other possibilities are open.

Theorem 5 $(\exists f \in \mathcal{R})(\forall x)[[\mathcal{C}_x \in \mathbf{TxtEx}_1 \wedge (\forall$ *distinct* $i, j \in W_x)[W_i \neq W_j]] \Rightarrow \mathcal{C}_x \in \mathbf{TxtBc}^*(\mathbf{M}_{f(x)})]$.

We omit the proof of the theorem due to space constraints.

Theorem 6 $(\forall f \in \mathcal{LR})(\forall n)(\exists x)[\mathcal{C}_x \in \mathbf{TxtEx}_1 - \mathbf{RecTxtBc}^n(\mathbf{M}_{f(x)})]$.

Proof. We prove here the following restricted version of the theorem only.

$(\forall f \in \mathcal{R})(\forall n)(\exists x)[\mathcal{C}_x \in \mathbf{TxtEx}_1 - \mathbf{RecTxtBc}^n(\mathbf{M}_{f(x)})]$.

Fix f and n. By the Operator Recursion Theorem [Cas74, Cas94], there exists a 1-1 increasing recursive function p such that the languages $W_{p(i)}$, $i \geq 0$, are defined as follows. Enumerate $p(1)$ in $W_{p(0)}$. $W_{p(1)}$ will be a subset of $\mathbf{ODD}$. The construction will use a set O. Initially, let $O = \emptyset$. Informally, O denotes the set of odd numbers we have decided to keep out of $W_{p(1)}$. Let σ_2 be the empty sequence. Go to stage 2.

Stage s

1. Enumerate $p(s)$ into $W_{p(0)}$. Dovetail the execution of steps 2 and 3. If and when step 3 succeeds, go to step 4.
2. Enumerate one-by-one, in increasing order, the elements of $\mathbf{ODD} - O$ into $W_{p(s)}$.
3. Search for $\sigma_{s+1} \supset \sigma_s$ and set P_s containing exactly $n+1$ distinct odd numbers such that $\mathrm{content}(\sigma_{s+1}) \subseteq \mathbf{ODD} - O$ and $P_s \subseteq (W_{\mathbf{M}_{f(p(0))}(\sigma_{s+1})} - \mathrm{content}(\sigma_{s+1}))$.
4.
   4.1 Enumerate $\mathrm{content}(\sigma_{s+1})$ into $W_{p(1)}$.
   4.2 Enumerate the (even) number $2s$ into $W_{p(s)}$.
   4.3 Let $O = O \cup P_s$, where P_s is the set found above in step 3.
   4.5 Go to stage $s+1$.

End stage s.

We consider two cases.

Case 1: Stage s starts but does not terminate.

In this case $W_{p(0)} = \{p(i) \mid 1 \leq i \leq s\}$. Observe that for each i such that $2 \leq i \leq s-1$, $2i$ is the only even number in $W_{p(i)}$. $W_{p(1)}$ is a finite subset of $\mathbf{ODD}$ and $W_{p(s)} = \mathbf{ODD} - O$, is an infinite subset of $\mathbf{ODD}$. It is then easy to verify that $\mathcal{C}_{p(0)} \in \mathbf{TxtEx}_1$.

Let $T \supset \sigma_s$ be a recursive text for $W_{p(s)}$. It is clear that for all $(\sigma \mid \sigma_s \subset \sigma \subset T)[W_{\mathbf{M}_{f(p(0))}(\sigma)} \cap \mathbf{ODD}$ is finite$]$. Thus, $\mathbf{M}$ does not $\mathbf{RecTxtBc}^n$-identify $W_{p(s)}$.

Case 2: All stages terminate.

In this case, clearly, for all $i > 1$, $W_{p(i)}$ is finite and contains exactly one even number, namely $2i$. Also, $W_{p(1)}$ contains only odd numbers. Thus $\mathcal{C}_{p(0)}$ belongs to $\mathbf{TxtEx}_1$.

We claim that $\mathbf{M}$ does not $\mathbf{RecTxtBc}^n$-identify $W_{p(1)}$. Let $T = \bigcup_{s\geq 2} \sigma_s$. Clearly, T is a recursive text with content exactly $W_{p(1)}$. Consider any stage $s \geq 2$. It is clear by steps 3 and 4 that, for all s, $\mathrm{card}(W_{\mathbf{M}_{f(p(0))}(\sigma_s)} - W_{p(1)}) \geq n+1$. Thus, T is a recursive text witnessing that $\mathbf{M}$ does not $\mathbf{RecTxtBc}^n$–identify $W_{p(1)}$. ∎

3.2 Synthesizing From Decision Procedures

Definition 8

(1) We say that x is a *uniform decision procedure* for a class $\mathcal{L}$ of recursive languages $\stackrel{\mathrm{def}}{\Leftrightarrow} [\varphi_x \in \mathcal{R}_{0,1} \ \wedge \ \mathcal{L} = \{L \mid (\exists i)[\chi_L(\cdot) = \varphi_x(i, \cdot)]\}]$.

(2) Suppose x *is* a uniform decision procedure. Then $\mathcal{U}_x$ is (by definition) the class of languages for which x is a uniform decision procedure.

(3) $\mathcal{L}$ is a *uniformly decidable class of languages* $\stackrel{\mathrm{def}}{\Leftrightarrow} (\exists x$, a uniform decision procedure$)[\mathcal{L} = \mathcal{U}_x]$.

(4) When a fixed uniform decision procedure x is understood, we sometimes then write U_i for the language whose characteristic function is $\varphi_x(i,\cdot)$.

(5) By $U_i[s]$ we mean $\{x \in U_i \mid x \leq s\}$.

As noted in Section 1, there has been considerable interest in the computational learning theory community in learnability, from positive data, of uniformly decidable classes of recursive languages [Ang80b, ZL95]. As Angluin [Ang80b] essentially points out, all of the formal language style example classes *are* uniformly decidable.

[Ang80b], for example, deals with so-called *exact* [LZ93] learning in which, for each learnable class, the programs learned derive naturally from the defining uniform decision procedure for that class.[8] Herein, we will synthesize learning machines whose hypothesis spaces, in some cases of necessity, go beyond ones naturally associated with the defining uniform decision procedures for the classes [LZ93].

The next two theorems (Theorems 7 and 8) deal with the special case where the uniformly decidable classes are *finite*. The first is an even more positive result than its analog for uniformly r.e. classes (Theorem 1 in Section 3.1 above). It's proof is straightforward, hence, omitted. The second shows the first is best possible.

Theorem 7 $(\forall n \geq 1)(\exists$ *recursive* $f)(\forall x \mid x$ *is a uniform decision procedure and* $\mathrm{card}(\mathcal{U}_x) = n)[\mathcal{U}_x \subseteq \mathbf{TxtEx}_{n-1}(\mathbf{M}_{f(x)})]$.

Theorem 8 $(\forall n \geq 1)(\exists i_0, i_1, \ldots, i_n$ *uniform decision procedures*$)$ $[\{U_{i_0}, U_{i_1}, \ldots U_{i_n}\} \notin \mathbf{TxtEx}^*_{n-1}]$.

PROOF. Consider $U_{i_j} = \{\langle x,y\rangle \mid y \in N \wedge x \leq j\}$. It is easy to verify that $[\{U_{i_0}, U_{i_1}, \ldots U_{i_n}\} \notin \mathbf{TxtEx}^*_{n-1}]$. ∎

The next two theorems (Theorems 9 and 10) concern synthesis from one-shot learnable uniformly decidable classes, and the first provides a much more positive result than its analog for uniformly r.e. classes (Theorem 3 in Section 3.1 above). The second shows the first is best possible and that the cost of passing from *no* mind changes in the input classes to *finitely* many in the *synthesized* learning machines is necessary.

Theorem 9 $(\forall a \in N \cup \{*\})(\exists f \in \mathcal{R})(\forall x \mid x$ *is a uniform decision procedure*$)[\mathcal{U}_x \in \mathbf{TxtEx}^a_0 \Rightarrow \mathcal{U}_x \in \mathbf{TxtEx}^a(\mathbf{M}_{f(x)})]$.

PROOF. Fix a. Let $\mathcal{U}_x \in \mathbf{TxtEx}^a_0$ be given. We first show the following Claim.

Claim 2 *For all* $L \in \mathcal{U}_x$*, there exist finite sets* S_L, S^1_L *and* S^2_L *such that*

(a) $S_L \subseteq L$*, and*

(b) for least i *such that* $S_L \subseteq U_i$*:*

$$(\forall L' \in \mathcal{U}_x \mid S_L \subseteq L')[(U_i - S^1_L) \cup S^2_L =^a L']$$

PROOF. Suppose $\mathcal{U}_x \in \mathbf{TxtEx}^a_0(\mathbf{M})$. Suppose $L \in \mathcal{U}_x$. Then there exists a sequence σ, such that $\mathrm{content}(\sigma) \subseteq L$ and $W_{\mathbf{M}(\sigma)} \neq ?$. This implies that for all $L' \in \mathcal{U}_x$ such that $\mathrm{content}(\sigma) \subseteq L'$, $W_{\mathbf{M}(\sigma)} =^a L'$. Let $S_L = \mathrm{content}(\sigma)$. Let i be the least number such that $S_L \subseteq U_i$. Let $S^1_L = U_i - W_{\mathbf{M}(\sigma)}$ and $S^2_L = W_{\mathbf{M}(\sigma)} - U_i$. It is easy to verify that (a) and (b) are satisfied. □

Claim 3 *Suppose* T *is a text for* $L \in \mathcal{U}_x$*. Let* S^1_T, S^2_T *and* S_T *be such that*

(a) $S_T \subseteq \mathrm{content}(T)$*, and*

(b) for least i *such that* $S_T \subseteq U_i$*:*

$$(\forall L' \in \mathcal{U}_x \mid S_T \subseteq L')[(U_i - S^1_T) \cup S^2_T =^a L']$$

Then, $[(U_i - S^1_T) \cup S^2_T] =^a L$.

PROOF. Since, $S_T \subseteq L$, it follows from clause (b) that $[(U_i - S^1_T) \cup S^2_T] =^a L$. □

Since one can verify in the limit, for given S^1_T, S^2_T and S_T, whether clauses (a) and (b) above are satisfied, one can algorithmically search for (some lexicographically least) such S^1_T, S^2_T, S_T. This is what gives us $\mathbf{M}_{f(x)}$ given below.

Let $\mathrm{Gram}(i, S_1, S_2)$ denote a grammar obtained algorithmically from i and finite sets S_1 and S_2, such that $W_{\mathrm{Gram}(i,S_1,S_2)} = (U_i - S_1) \cup S_2$.

$\mathbf{M}_{f(x)}(T[n])$:

1. Let $S_n, S^1_n, S^2_n \subseteq \{x \leq n\}$ be (lexicographically least (if any)) finite sets such that
   (a) $S_n \subseteq \mathrm{content}(T[n])$
   (b) for least i such that $S_n \subseteq U_i$:
   $$(\forall j \mid S_n \subseteq U_j)[(U_i[n] - S^1_n) \cup S^2_n =^a U_j[n]]$$
2. If no such S_n, S^1_n, S^2_n is found in the search above, then output ?. Else let S_n, S^1_n, S^2_n be the lexicographically least such set. For least i such that $S_n \subseteq U_i$, output $\mathrm{Gram}(i, S^1_n, S^2_n)$.

End $\mathbf{M}_{f(x)}$

Using Claim 2, it is easy to verify that, for any T for $L \in \mathcal{U}_x$, S_n, S^1_n, S^2_n as found in step 1 above converges to S_T, S^1_T, S^2_T, which satisfy (a) and (b) in Claim 3. It thus follows that $\mathbf{M}_{f(x)}$ $\mathbf{TxtEx}^a$-identifies $\mathcal{U}_x$. ∎

Theorem 10 $(\forall n)\neg(\exists f \in \mathcal{LR})(\forall x \mid x$ *is a uniform decision procedure* $)[\mathcal{U}_x \in \mathbf{TxtEx}_0 \Rightarrow \mathcal{U}_x \in \mathbf{TxtEx}^*_n(\mathbf{M}_{f(x)})]$.

[8] Typically, in the literature one uses i as a "program" for the U_i from Definition 8 above. S-m-n in the φ-system [Rog67] clearly yields corresponding φ-decision procedures or grammars as one wishes.

PROOF. Fix n. For simplicity of presentation, we give the proof only for recursive f. The proof can be easily generalized to limiting recursive f. By implicit use of recursion theorem there exists an x such that $\mathcal{U}_x$ may be described as follows.

Let σ_0 be a sequence, if any, such that $\mathbf{M}_{f(x)}(\sigma_0) \neq ?$. For $i \leq n$, if σ_i is defined, then try to define σ_{i+1} as follows: let σ_{i+1} be a sequence, if any, such that $\sigma_i \subseteq \sigma_{i+1}$ and $\mathbf{M}_{f(x)}(\sigma_i) \neq \mathbf{M}_{f(x)}(\sigma_{i+1})$.

Let $L_0 = \{\langle 0, y\rangle \mid y \in N\}$. For i such that σ_i is defined in the process above, define L_{2i+1}, L_{2i+2} as follows: $L_{2i+1} = \text{content}(\sigma_i) \cup \{\langle 2i+1, y\rangle \mid y \in N\}$. $L_{2i+2} = \text{content}(\sigma_i) \cup \{\langle 2i+2, y\rangle \mid y \in N\}$.

Let $\mathcal{U}_x = \{L_0\} \cup \{L_{2i+1}, L_{2i+2} \mid \sigma_i$ is defined in the process above $\}$. Since, $\mathcal{U}_x$ is finite and all languages in $\mathcal{U}_x$ are pairwise incomparable (by $\subset$), it follows that $\mathcal{U}_x \in \mathbf{TxtEx}_0$. We claim that $\mathcal{U}_x \not\subseteq \mathbf{TxtEx}^*_n(\mathbf{M}_{f(x)})$.

If σ_{n+1} is defined then, $\mathbf{M}_{f(x)}$ makes at least $n+1$ mind changes on σ_{n+1} and thus does not $\mathbf{TxtEx}^*_n$-identify $L_{2i+3}, L_{2i+4} \in \mathcal{U}_x$.

If σ_0 is not defined, then $L_0 \in \mathcal{U}_x$, but $\mathbf{M}_{f(x)}$ does not $\mathbf{TxtEx}^*$-identify L_0.

If σ_{i+1} is not defined, then $L_{2i+1}, L_{2i+2} \in \mathcal{U}_x$, but $\mathbf{M}_{f(x)}$ does not $\mathbf{TxtEx}^*$-identify at least one of L_{2i+1}, L_{2i+2} (since content$(\sigma_i) \subseteq L_{2i+1} \cap L_{2i+2}$, $L_{2i+1} \neq^* L_{2i+2}$ and $\mathbf{M}_{f(x)}$ on any extension of σ_i outputs $\mathbf{M}_{f(x)}(\sigma_i)$).

The theorem follows. ∎

Next we present our two Main Corollaries (Corollaries 2 and 3) which *completely characterize* the uniformly decidable classes in $\mathbf{TxtBc}^a$ and are easy consequences of Lemmas 1 and 3 following them.[9] The first corollary is a very important case of the second which we've separated out for special attention.

As we noted in Section 1 above, Angluin [Ang80b] completely characterized the uniformly decidable classes in $\mathbf{TxtEx}$.[10] Essentially she showed that, for any fixed uniform decision procedure x, $\mathcal{U}_x \in \mathbf{TxtEx} \Leftrightarrow$ Condition 1 holds, where Condition 1 states:[11]

> There is an *r.e.* sequence of (r.e. indices of) finite sets $S_0, S_1, \ldots$ (called *tell tales*) such that,
>
> $$(\forall i)[S_i \subseteq U_i \ \wedge\ (\forall j \mid S_i \subseteq U_j)[U_j \not\subset U_i]]. \quad (1)$$

She also considered a Condition 2 just like Condition 1 *except* that the sequence of finite sets (tell tales) is *not* required to be r.e. and showed that Condition 2 is *not* sufficient. Our characterization of uniformly decidable classes in $\mathbf{TxtBc}$ is, surprisingly, just Angluin's Condition 2! As mentioned above, it is referred to as the *subset principle*, a necessary condition preventing overgeneralization in learning from positive data [Ang80b, Ber85, Cas92].

Corollary 2 $\mathcal{U}_x \in \mathbf{TxtBc} \Leftrightarrow (\forall U \in \mathcal{U}_x)(\exists S \subseteq U \mid S$ *is finite*$)(\forall U' \in \mathcal{U}_x \mid S \subseteq U')[U' \not\subset U]$.

It is surprising and important that the subset principle alone (Angluin's Condition 2) *without the added constructivity conditions of Angluin's Condition 1* characterizes the uniformly decidable classes $\mathcal{U}_x \in \mathbf{TxtBc}$.

[OSW86b] note that a class of r.e. languages $\mathcal{U}$ can be learned in the limit from positive data *by a not necessarily algorithmic procedure* iff Angluin's Condition 2 holds for $\mathcal{U}$. Hence, Corollary 2 together with this observation entails that *for uniformly decidable classes* $\mathcal{U}$, $\mathcal{U}$ can be learned in the limit from positive data *by a not necessarily algorithmic procedure* iff $\mathcal{U} \in \mathbf{TxtBc}$. Therefore, for uniformly decidable classes, algorithmicity of the learning procedure doesn't matter for behaviorally correct identification! It is open whether there are other types of classes $\mathcal{U}$ (besides uniformly decidable) for which algorithmicity of the learning procedure doesn't matter for behaviorally correct identification.

Suppose x is a uniform decision procedure. Corollary 2, immediately above also provides the following characterization.

$$\mathcal{U}_x \in (\mathbf{TxtBc} - \mathbf{TxtEx}) \Leftrightarrow \mathcal{U}_x \text{ satisfies Condition 2 but not Condition 1.}$$

Hence, since Angluin [Ang80b] provided an example $\mathcal{U}_x$ satisfying Condition 2 and not Condition 1, her example is a uniformly decidable class witnessing that $(\mathbf{TxtBc} - \mathbf{TxtEx}) \neq \emptyset$.

Our characterization for $\mathbf{TxtBc}^a$ more generally is a variant of Angluin's Condition 2.[12]

Corollary 3 $\mathcal{U}_x \in \mathbf{TxtBc}^a \Leftrightarrow (\forall U \in \mathcal{U}_x)(\exists S \subseteq U \mid S$ *is finite*$)(\forall U' \mid S \subseteq U' \in \mathcal{U}_x)[U' \not\subset U$ *or* $\text{card}(U - U') \leq 2a]$.

As we will see, our Main Theorem (Theorem 11) below is an immediate consequence of Lemmas 1 and 3 to follow.

Lemma 1 *Suppose* $\mathcal{U}_x \in \mathbf{TxtBc}^a$. *Then* $(\forall L \in \mathcal{U}_x)(\exists X_L \subseteq L \mid X_L$ *is finite*$)(\forall L' \in \mathcal{U}_x \mid X_L \subseteq L')[L' \not\subset L \ \vee\ \text{card}(L - L') \leq 2a]$.

PROOF. Suppose $\mathcal{U}_x \in \mathbf{TxtBc}^a(\mathbf{M})$. Suppose $L \in \mathcal{U}_x$. Then there exists a $\mathbf{TxtBc}^a$-locking sequence σ for $\mathbf{M}$

[9] It seems pedagogically useful to present the results in this order.

[10] Mukouchi [Muk92] characterized the uniformly decidable classes in $\mathbf{TxtEx}_0$. [dJK96] surprisingly characterizes the *r.e.* classes in $\mathbf{TxtEx}$ and presents other interesting results.

[11] Recall that the U_i's are defined in Definition 8 above.

[12] We also have a variant of Angluin's characterization above, but for $\mathbf{TxtEx}^*$ in place of $\mathbf{TxtEx}$, which characterization is just like hers *except* that (1) above is replaced by

$$(\forall i)[S_i \subseteq U_i \ \wedge\ (\forall j \mid S_i \subseteq U_j \subseteq U_i)[U_j =^* U_i]]. \quad (2)$$

on L. Let $X_L = \text{content}(\sigma)$. Note that $W_{\mathbf{M}(\sigma)} =^a L$. Now, for any $L' \in \mathcal{U}_x$ such that $X_L \subseteq L' \subseteq L$, we must have $W_{\mathbf{M}(\sigma)} =^a L'$ (otherwise, $\mathbf{M}$ does not $\mathbf{TxtBc}^a$-identify L'). It follows that $L =^{2a} L'$. ∎

Before presenting our Main Lemma (Lemma 3), we present a slightly weaker version of it: in Lemma 3, the $\mathbf{TxtBc}^{2a}$ in Lemma 2, is replaced by just $\mathbf{TxtBc}^a$.

Lemma 2 *There exists an $f \in \mathcal{R}$ such that the following satisfied.*

Suppose x is a uniform decision procedure. Further suppose $(\forall L \in \mathcal{U}_x)(\exists X_L \subseteq L \mid X_L$ is finite$)(\forall L' \in \mathcal{U}_x \mid X_L \subseteq L')[L' \not\subseteq L \ \vee\ \text{card}(L - L') \leq 2a]$. Then, $[\mathcal{U}_x \subseteq \mathbf{TxtBc}^{2a}(\mathbf{M}_{f(x)})]$.

Proof. Suppose x is a uniform decision procedure satisfying the hypothesis of the theorem. We describe the construction of $\mathbf{M}_{f(x)}$. It is easy to see that the construction is algorithmic in x.

$\mathbf{M}_{f(x)}(T[n]) = \text{Proc}(T[n])$, where $W_{\text{Proc}(T[n])}$ is defined as follows.

$W_{\text{Proc}(T[n])}$
1. Let $P_n = \{i \leq n \mid \text{content}(T[n]) \subseteq U_i\}$.
2. Go to stage 0.

Begin Stage s
3. Let $DelS_n^s = \{i \in P_n \mid (\exists i' \in P_n \mid i' < i)[U_i[s] \not\subseteq U_{i'}[s]]\}$.
(* Note that $DelS_n^s \subseteq DelS_n^{s+1}$. Intuitively $DelS_n^s$ consists of grammars we want to delete from P_n, since they seem to be bad (see analysis below) *).
4. Let $S_n^s = P_n - DelS_n^s$
(* Note that $(\forall i, i' \in S_n^s \mid i' < i)[U_i[s] \subseteq U_{i'}[s]]$ *)
5. Let $i_n^s = \max(S_n^s)$. Enumerate $U_{i_n^s}[s]$.
6. Go to stage $s+1$.

End stage s
End $W_{\text{Proc}(T[n])}$

Suppose $L \in \mathcal{U}_x$ and T is a text for L. We claim that for all but finitely many n, $W_{\text{Proc}(T[n])} =^{2a} L$. This will prove the theorem. Let X_L be as given in the hypothesis. It follows that for all $L' \in \mathcal{U}_x$, $[X_L \subseteq L'] \Rightarrow [L' \not\subseteq L \ \vee\ L' =^{2a} L]$. Let j be the minimal number such that $U_j = L$.

Let n be large enough so that,

(i) For all $i < j$ such that $U_i \not\supseteq U_j$, $\text{content}(T[n]) \not\subseteq U_i$.

(ii) $j < n$.

(iii) $X_L \subseteq \text{content}(T[n])$.

We claim that $W_{\text{Proc}(T[n])} =^{2a} U_j = L$. Let $P_n, S_n^s, DelS_n^s, i_n^s$ be as defined in $Proc(T[n])$ above. They satisfy the following properties:

(a) $j \in P_n$.

(b) $(\forall i < j)[i \in P_n \Rightarrow U_i \supset L]$.

(c) $(\forall i > j)[i \in P_n \Rightarrow [U_i \not\subseteq L \ \vee\ U_i =^{2a} L]]$.

(d) for all s: $j \notin DelS_n^s$; thus $j \in S_n^s$.

It immediately follows from (d) above and the comment after step (4) that $W_{\text{Proc}(T[n])} \subseteq U_j$.

Now note that, since $DelS_n^s \subseteq DelS_n^{s+1}$, we have $S_n^s \supseteq S_n^{s+1}$. Thus $\lim_{s\to\infty} i_n^s$ is defined. Let this limiting value be i_n. It follows that $U_{i_n} \subseteq W_{Proc(T[n])} \subseteq U_j = L$. Now since, $U_{i_n} \subseteq U_j$, property (c) above implies that $U_{i_n} =^{2a} U_j$. Thus, $W_{Proc(T[n])} =^{2a} L$. It follows that $\mathbf{M}_{f(x)}$ $\mathbf{TxtBc}^{2a}$-identifies L. ∎

Lemma 3 *There exists a recursive f satisfying the following. Suppose x is a uniform decision procedure. Further suppose $(\forall L \in \mathcal{U}_x)(\exists X_L \subseteq L \mid X_L$ is finite$)(\forall L' \in \mathcal{U}_x \mid X_L \subseteq L')[L' \not\subseteq L \ \vee\ \text{card}(L - L') \leq 2a]$. Then, $[\mathcal{U}_x \subseteq \mathbf{TxtBc}^{a}(\mathbf{M}_{f(x)})]$.*

The proof of the lemma, is a careful modification of the proof of Lemma 2 to reduce the errors. The weaker version, Lemma 2, above suffices to obtain the $a \in \{0, *\}$ cases. We omit the details due to space constraints.

Next we have our Main Theorem (Theorem 11) which follows immediately from Lemmas 1 and 3 above. It says that we *can* synthesize, from uniform decision procedures for classes in $\mathbf{TxtBc}^a$, $\mathbf{TxtBc}^a$-learning machines! Hence, in passing from learnable uniformly decidable classes to algorithmically synthesized learning machines for them, we get a fixed point, for each a, at $\mathbf{TxtBc}^a$-identification!

Theorem 11 $(\forall a \in N \cup \{*\})(\exists f \in \mathcal{R})[\mathcal{U}_x \in \mathbf{TxtBc}^a \Rightarrow \mathcal{U}_x \subseteq \mathbf{TxtBc}^a(\mathbf{M}_{f(x)})]$.

The next and last theorem of this section contrasts nicely with Theorems 9 and 11 above. It also shows that the cost of passing from *one* mind change in the input classes to *infinitely* many in the *synthesized* learning machines is necessary. This is so, as in our other lower bound results above, even if we employed the stronger limiting recursive procedures for synthesis of learning machines from algorithmic descriptions of the class to be learned!

Theorem 12 $\neg(\exists f \in \mathcal{LR})[\mathcal{U}_x \in \mathbf{TxtEx}_1 \Rightarrow \mathcal{U}_x \subseteq \mathbf{TxtFex}_*^*(\mathbf{M}_{f(x)})]$.

Proof. We prove a simpler version of the theorem:

$$\neg(\exists f \in \mathcal{R})[\mathcal{U}_x \in \mathbf{TxtEx}_1 \Rightarrow \mathcal{U}_x \subseteq \mathbf{TxtEx}^*(\mathbf{M}_{f(x)})]. \tag{3}$$

The proof can be easily generalized to take care of $\mathbf{TxtFex}_*^*$ and limiting recursive f.

Suppose by way of contradiction otherwise. Let f be given. By the Operator Recursion Theorem [Cas74, Cas94], there exists a 1-1 increasing, p such that the following holds.

We let $\mathcal{L} = \{L \mid (\exists i \geq 1)[\chi_L = \varphi_{p(i)}]\}$.

It will be easily seen that, for $i > 0$, $\varphi_{p(i)}$ is either a characteristic function or an empty function. Hence, it follows that a uniform decision procedure for $\mathcal{L}$ (defined just above) exists and can be found. Let $p(0)$ be uniform decision procedure for $\mathcal{L}$.

We note that all $\varphi_{p(i)}$, $i \geq 1$, *which are considered in the construction below* will be characteristic functions.

If $\varphi_{p(i)}$ is a characteristic function, we will abuse notation slightly and refer to the language for which it's a characteristic function as $U_{p(i)}$.

We will have that $\mathcal{U}_{p(0)} \in \mathbf{TxtEx}_1$. Thus $\mathbf{M}_{f(p(0))}$ $\mathbf{TxtEx}^*$ identifies $\mathcal{U}_{p(0)}$. We let $U_{p(1)} = \mathbf{ODD}$.

We will have that, for all $i > 1$, $U_{p(i)}$ is finite. In addition the construction will ensure that at least one of the following properties (A) and (B) is satisfied.

(A) For $i > 1$, $U_{p(i)}$ contains exactly one even number. Moreover, for all $i > i' > 1$, $U_{p(i)} \cap U_{p(i')} \cap \mathbf{EVEN} = \emptyset$.

(B) There exists a $j > 1$ such that, B.1, B.2 and B.3 are satisfied.

(B.1) for all $i > j$, $\varphi_{p(i)}(0)\uparrow$.

(B.2) for all i such that $1 < i < j$, $U_{p(i)}$ contains exactly one even number. Moreover, for all $1 < i < i' < j$, $W_{p(i)} \cap W_{p(i')} \cap \mathbf{EVEN} = \emptyset$.

(B.3) $W_{p(j)}$ is finite, does not contain any even number, and for all $1 < i < j$, $W_{p(j)} \not\subseteq W_{p(i)}$.

Note that the above properties imply that $\mathcal{C}_{p(0)} \in \mathbf{TxtEx}_1$. In case (A) clearly, a machine can first output a grammar for $\mathbf{ODD}$. Then if it sees an even number it can output a grammar for the corresponding finite set using appropriate $p(i)$. In case (B) $\mathcal{C}_{p(0)}$ is finite and machine can $\mathbf{TxtEx}_1$-identify by first waiting until it gets an even number or sees $W_{p(j)}$ in the input and then output the corresponding grammar. The machine now needs to change its conjecture only if the input is for $\mathbf{ODD}$, requiring at most 1 mind change.

We now proceed to define $W_{p(i)}$ for $i > 1$. Let σ_0 be a sequence such that content$(\sigma) = \{1\}$. Go to stage 0.

Stage s

1. Start defining $p(s+2)$ so that it will be a characteristic function for content(σ_s), unless step 2 below succeeds.
2. Search for an extension σ of σ_s such that content$(\sigma) \subseteq \mathbf{ODD}$ and $\mathbf{M}_{f(p(0))}(\sigma) \neq \mathbf{M}_{f(p(0))}(\sigma_s)$.
3. If and when such a σ is found,

    Let e be an even number large enough so that $\varphi_{p(s+2)}(e)$ has not yet been defined and e is bigger than all the even numbers considered in previous stages.

    Let $\varphi_{p(s+2)}$ be characteristic function for content$(\sigma_s) \cup \{e\}$.

    Let σ_{s+1} be an extension of σ such that content$(\sigma_{s+1}) \supset$ content$(\sigma_s) \cup \{2x+1 \mid x \leq \max(\text{content}(\sigma_s))\}$.
4. Go to stage $s+1$.

End stage s

Now consider the following cases.

Case 1: All stages terminate.

In this case clearly, for all $i > 1$, $U_{p(i)}$ is finite, and $p(i)$ satisfy the property (A) above. Moreover, $\mathbf{M}_{f(p(0))}$ makes infinitely many mind changes on $\bigcup_s \sigma_s$ which is a text for $\mathbf{ODD}$.

Case 2: stage s starts but does not terminate.

In this case clearly, property (B) above is satisfied with $j = s+2$. Moreover, $\mathbf{M}_{f(p(0))}$ can $\mathbf{TxtEx}^*$-identify at most one of $U_{p(s+2)}$ and $U_{p(1)} = \mathbf{ODD}$, since step 2 does not succeed.

This proves the theorem. ∎

N.B. Angluin's proof of her characterization (by Condition 1) of uniformly decidable classes in $\mathbf{TxtEx}$ essentially provides an algorithm for transforming *two* things into a learning machine which $\mathbf{TxtEx}$-identifies $\mathcal{L}$:[13]

(1) a uniform decision procedure for a class of recursive languages $\mathcal{L}$ that is $\mathbf{TxtEx}$-identifiable *and*

(2) *a program for generating an associated r.e. sequence of tell tale sets $S_0, S_1, \ldots$ as featured in Equation (1) above.*

The *additional input information* of a program for generating the tell tales therefore makes a huge difference in the mind change complexity of the synthesized learning machine!

4 Future Directions

Barzdin [BF72] first considered improvements of archetypal enumeration techniques, involving a majority vote strategy which has vastly better mind-change complexity. See also [LM89]. It would be interesting to look into variants of our algorithms above for synthesizing learning machines with improved mind-change complexity at least for interesting special cases.

Corollary 1 above and, more generally, the footnote to Theorem 1 above suggest to us ones exploring the relevance to our paper's topics of r.e. classes of languages which can be enumerated with $\leq n+1$ duplications but

[13] Actually Angluin's synthesized machine learns decision procedures rather than grammars, and, for $\mathbf{TxtEx}$-identification of uniformly decidable classes of languages, one can learn grammars $\Leftrightarrow$ one can learn decision procedures. We can prove this latter is *not* the case for $\mathbf{TxtBc}$-identification (of uniformly decidable classes of languages).

not with $\leq n$ [PEH64, PEP65]. In this interest we have a preliminary result complementing Theorem 3 above (in Section 3.1) as follows.

Theorem 13 $(\exists$ *recursive* $f)(\forall x \mid \mathcal{C}_x \in \mathbf{TxtEx}_0 \ \wedge \ (\forall$ *distinct* $i, j \in W_x)[W_i \neq W_j])[\mathcal{C}_x \subseteq \mathbf{TxtEx}(\mathbf{M}_{f(x)})]$.

We also know that, in this result, $\mathbf{TxtEx}(\mathbf{M}_{f(x)})$ cannot be improved to $\mathbf{TxtEx}_n(\mathbf{M}_{f(x)})$.

It is interesting to explore extensions of the present paper for the cases of adding small amounts of negative information to the input data [BCJ95]. In the light of Theorem 22 in [BCJ95] and the discussion following the proof of Theorem 12 above, it is reasonable to hope for some resultant improvements in mind change complexity for synthesized machines.

References

[Ang80a] D. Angluin. Finding patterns common to a set of strings. *Journal of Computer and System Sciences*, 21:46–62, 1980.

[Ang80b] D. Angluin. Inductive inference of formal languages from positive data. *Information and Control*, 45:117–135, 1980.

[Ang82] D. Angluin. Inference of reversible languages. *Journal of the ACM*, 29:741–765, 1982.

[AS83] D. Angluin and C. Smith. A survey of inductive inference: Theory and methods. *Computing Surveys*, 15:237–289, 1983.

[Bar74] J. Barzdin. Two theorems on the limiting synthesis of functions. *In Theory of Algorithms and Programs, Latvian State University, Riga*, 210:82–88, 1974.

[BB75] L. Blum and M. Blum. Toward a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

[BC93] G. Baliga and J. Case. Learnability: Admissible, co-finite, and hypersimple sets. In *Proceedings of the 20th International Colloquium on Automata, Languages and Programming*, volume 700 of *Lecture Notes in Computer Science*, pages 289–300. Springer-Verlag, Berlin, Lund, Sweden, July 1993. Journal version in press for *Journal of Computer and System Sciences*, 1996.

[BCJ95] G. Baliga, J. Case, and S. Jain. Language learning with some negative information. *Journal of Computer and System Sciences*, 51:273–285, 1995.

[Ber85] R. Berwick. *The Acquisition of Syntactic Knowledge.* MIT Press, Cambridge, MA, 1985.

[BF72] J. Barzdin and R. Freivalds. On the prediction of general recursive functions. *Soviet Mathematics Doklady*, 13:1224–1228, 1972.

[Blu67] M. Blum. A machine independent theory of the complexity of recursive functions. *Journal of the ACM*, 14:322–336, 1967.

[Cas74] J. Case. Periodicity in generations of automata. *Mathematical Systems Theory*, 8:15–32, 1974.

[Cas86] J. Case. Learning machines. In W. Demopoulos and A. Marras, editors, *Language Learning and Concept Acquisition.* Ablex Publishing Company, 1986.

[Cas88] J. Case. The power of vacillation. In D. Haussler and L. Pitt, editors, *Proceedings of the Workshop on Computational Learning Theory*, pages 133–142. Morgan Kaufmann Publishers, Inc., 1988. Expanded in [Cas92].

[Cas92] J. Case. The power of vacillation in language learning. Technical Report 93-08, University of Delaware, 1992. Expands on [Cas88]; journal version revised for possible publication.

[Cas94] J. Case. Infinitary self-reference in learning theory. *Journal of Experimental and Theoretical Artificial Intelligence*, 6:3–16, 1994.

[CL82] J. Case and C. Lynes. Machine inductive inference and language identification. In M. Nielsen and E. Schmidt, editors, *Proceedings of the 9th International Colloquium on Automata, Languages and Programming*, volume 140, pages 107–115. Springer-Verlag, Berlin, 1982.

[CS78] J. Case and C. Smith. Anomaly hierarchies of mechanized inductive inference. In *Symposium on the Theory of Computation*, pages 314–319, 1978.

[CS83] J. Case and C. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.

[dJK96] D. de Jongh and M. Kanazawa. Angluin's thoerem for indexed families of r.e. sets and applications. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory, Desenzano del Garda, Italy.* ACM Press, July 1996.

[Fre85] R. Freivalds. Recursiveness of the enumerating functions increases the inferrability of recursively enumerable sets. *Bulletin of the European Association for Theoretical Computer Science*, 27:35–40, 1985.

[Ful85] M. Fulk. *A Study of Inductive Inference machines.* PhD thesis, SUNY at Buffalo, 1985.

[Ful90a] M. Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.

[Ful90b] M. Fulk. Robust separations in inductive inference. In *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, pages 405–410, St. Louis, Missouri 1990.

[Gle86] L. Gleitman. Biological dispositions to learn language. In W. Demopoulos and A. Marras, editors, *Language Learning and Concept Acquisition.* Ablex Publ. Co., 1986.

[Gol67] E. Gold. Language identification in the limit. *Information and Control*, 10:447–474, 1967.

[HU79] J. Hopcroft and J. Ullman. *Introduction to Automata Theory Languages and Computation.* Addison-Wesley Publishing Company, 1979.

[Kir92] D. Kirsh. PDP learnability and innate knowledge of language. In S. Davis, editor, *Connectionism: Theory and Practice*, pages 297–322. Oxford University Press, NY, 1992.

[KLHM93] S. Kapur, B. Lust, W. Harbert, and G. Martohardjono. Universal grammar and learnability theory: The case of binding domains and the 'subset principle'. In E. Reuland and W. Abraham, editors, *Knowledge and Language*, volume I, pages 185–216. Kluwer, 1993.

[KW80] R. Klette and R. Wiehagen. Research in the theory of inductive inference by GDR mathematicians – A survey. *Information Sciences*, 22:149–169, 1980.

[LM89] N. Littlestone and Warmuth M. The weighted majority algorithm. In *Symposium on the Theory of Computation*, pages 256–261, 1989.

[LZ93] S. Lange and T. Zeugmann. Language learning in dependence on the space of hypotheses. In *Proc. 6th Annual Conf. on Comp. Learning Theory*, pages 156–166, 1993.

[MR67] A. Meyer and D. Ritchie. The complexity of loop programs. In *Proceedings of the 22nd National ACM Conference*, pages 465–469. Thomas Book Co., 1967.

[Muk92] Y. Mukouchi. Characterization of finite identification. In K. P. Jantke, editor, *Proceedings of the Third International Workshop on Analogical and Inductive Inference, Dagstuhl Castle, Germany*, pages 260–267, October 1992.

[MW87] R. Manzini and K. Wexler. Parameters, binding theory and learnability. *Linguistic Inquiry*, 18:413–444, 1987.

[MY78] M. Machtey and P. Young. *An Introduction to the General Theory of Algorithms.* North Holland, New York, 1978.

[OSW82] D. Osherson, M. Stob, and S. Weinstein. Ideal learning machines. *Cognitive Science*, 6:277–290, 1982.

[OSW84] D. Osherson, M. Stob, and S. Weinstein. Learning theory and natural language. *Cognition*, 17:1–28, 1984.

[OSW86a] D. Osherson, M. Stob, and S. Weinstein. An analysis of a learning paradigm. In W. Demopoulos and A. Marras, editors, *Language Learning and Concept Acquisition.* Ablex Publ. Co., 1986.

[OSW86b] D. Osherson, M. Stob, and S. Weinstein. *Systems that Learn, An Introduction to Learning Theory for Cognitive and Computer Scientists.* MIT Press, Cambridge, Mass., 1986.

[OSW88] D. Osherson, M. Stob, and S. Weinstein. Synthesizing inductive expertise. *Information and Computation*, 77:138–161, 1988.

[OW82a] D. Osherson and S. Weinstein. Criteria of language learning. *Information and Control*, 52:123–138, 1982.

[OW82b] D. Osherson and S. Weinstein. A note on formal learning theory. *Cognition*, 11:77–88, 1982.

[PEH64] M. Pour-El and W. Howard. A structural criterion for recursive enumeration without repetition. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 10:105–114, 1964.

[PEP65] M. Pour-El and H. Putnam. Recursively enunumerable classes and their application to recursive sequences of formal theories. *Arch. f. Math. Log. Grund.*, 8:104–121, 1965.

[Pin79] S. Pinker. Formal models of language learning. *Cognition*, 7:217–283, 1979.

[RC94] J. Royer and J. Case. *Subrecursive Programming Systems: Complexity and Succinctness.* Progress in Theoretical Computer Science. Birkhäuser Boston, 1994.

[Rog58] H. Rogers. Gödel numberings of partial recursive functions. *Journal of Symbolic Logic*, 23:331–341, 1958.

[Rog67] H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw Hill, New York, 1967. Reprinted, MIT Press, 1987.

[Roy87] J. Royer. *A Connotational Theory of Program Structure.* Lecture Notes in Computer Science 273. Springer Verlag, 1987.

[Sha71] N. Shapiro. Review of "Limiting recursion" by E.M. Gold and "Trial and error predicates and the solution to a problem of Mostowski" by H. Putnam. *Journal of Symbolic Logic*, 36:342, 1971.

[WC80] K. Wexler and P. Culicover. *Formal Principles of Language Acquisition.* MIT Press, Cambridge, Mass, 1980.

[Wex82] K. Wexler. On extensional learnability. *Cognition*, 11:89–95, 1982.

[Wex93] K. Wexler. The subset principle is an intensional principle. In E. Reuland and W. Abraham, editors, *Knowledge and Language*, volume I, pages 217–239. Kluwer, 1993.

[Wie90] R. Wiehagen. A thesis in inductive inference. In P. Schmitt J. Dix, K. Jantke, editor, *Nonmonotonic and Inductive Logic, 1st International Workshop*, volume 543 of *Lecture Notes in Artificial Intelligence*, pages 184–207. Springer Verlag, Karlsruhe, Germany 1990.

[ZL95] T. Zeugmann and S. Lange. A guided tour across the boundaries of learning recursive languages. In Klaus P. Jantke and Steffen Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 190–258. Springer-Verlag, 1995.