On a Generalized Notion of Mistake Bounds

Sanjay Jain
School of Computing
National University of Singapore
Singapore 119260, Republic of Singapore
Email: sanjay@comp.nus.edu.sg

Arun Sharma
School of Computer Science and Engineering
The University of New South Wales
Sydney, NSW 2052, Australia
Email: arun@cse.unsw.edu.au

Abstract

This paper proposes the use of constructive ordinals as mistake bounds in the on-line learning model. This approach elegantly generalizes the applicability of the on-line mistake bound model to learnability analysis of very expressive concept classes like pattern languages, unions of pattern languages, elementary formal systems, and minimal models of logic programs.

The main result in the paper shows that the topological property of effective finite bounded thickness is a sufficient condition for on-line learnability with a certain ordinal mistake bound.

An interesting characterization of the on-line learning model is shown in terms of the identification in the limit framework. It is established that the classes of languages learnable in the on-line model with a mistake bound of α are exactly the same as the classes of languages learnable in the limit from both positive and negative data by a Popperian, consistent learner with a mind change bound of α. This result nicely builds a bridge between the two models.

1 Introduction

There has been considerable work in the on-line model of learning where a learning algorithm's behavior is evaluated by counting the worst-case number of mistakes that it makes while learning a function (see Barzdin and Freivalds [BF72] and Littlestone [Lit88]). This approach has been successfully applied in analyzing learnability of several classes of functions.

Restricting the mistake bounds to natural numbers, however, limits the use of this model as it cannot shed light on learnability of more expressive concept classes like logic programs and pattern languages. In the present paper, we show that the use of constructive ordinals as mistake bounds provides an elegant extension to this model enabling the analysis of learnability of rich concept classes.

The use of constructive ordinals here is not new, they have been used in the inductive inference framework to count the number of mind changes by Freivalds and Smith [FS93] and by Jain and Sharma [JS97] (see also [SSV97, AFS96]). What is new here is how elegantly their use extends the applicability of the on-line mistake bound model of learnability. The rest of this section is devoted to an informal description of these ideas.

Let N be the set of natural numbers. Let $L \subseteq N$ be a language.

Our setting is the on-line learning of characteristic functions of languages. The learning takes place in a sequence of trials with the following order of events:

(a) The learner receives an example $x \in N$.

(b) The learner makes a response of 1 to conjecture that $x \in L$ or a response of 0 to conjecture that $x \notin L$.

(c) The learner is told whether its response in step (b) to the example received in step (a) was correct or not.

A new trial begins after the previous one has ended.

Let m be a natural number. The learner is said to learn a class of languages $\mathcal{L}$ with a mistake-bound of m just in case for any L from $\mathcal{L}$, the learner, presented with instances from N in any order, makes no more than m mistakes before it always conjectures the status of instances in L correctly.

Now, to see how allowing only natural numbers to be mistake bounds constrains the applicability of this learning model, we consider COINIT, the class of coinitial languages defined below.

$$\begin{aligned} \mathsf{COINIT} \quad = \quad \{&\{0,1,2,3,4\ldots\},\\ &\{1,2,3,4\ldots\},\\ &\{2,3,4,\ldots\},\\ &\{3,4,\ldots\},\\ &\ldots\} \end{aligned}$$

In the on-line model, no natural number is a mistake bound for this class. However, it is not too difficult to see that an on-line learner is in a position to provide a mistake bound as soon as it has correctly seen a positive example in the target language. This situation, though quite intuitive, cannot be modeled by any fixed natural number as mistake bound.

Similarly, there is no natural number that is a mistake bound for PATTERN, the class of pattern languages introduced by Angluin [Ang80a]. Again a learner is in a position to provide an upper bound on the number of mistakes it will make as soon as it sees the first positive string from the target language.

In the present paper, we present a generalized notion of mistake bounds, where the mistake bound for COINIT (and for PATTERN) is the first limit ordinal, ω. A mistake bound of $\omega \times 2$ models the situation where the learner conjectures a mistake bound, but reserves the right to revise the bound once; in case of $\omega \times 3$, it reserves the right to revise the bound twice, and so on. A mistake bound of ω^2 means that the learner first conjectures an upper bound on the number of times it will conjecture a mistake bound, and so on.

After incorporating ordinals into the on-line learning model, we derive a sufficient condition for a class of languages to have a mistake bound of the form $\omega \times n$, where n is a natural number. This condition is established in terms of the concept class satisfying the topological property of effective finite bounded thickness [Ang80b, Wri89, Shi91, JS99].

As a consequence of our sufficient condition, the mistake bound of unions of no more than n pattern languages is $\omega \times n$. Also, as a consequence of this sufficient condition, it follows that for the class of linear logic programs (see Shapiro [Sha81a]) with no more than m clauses, the mistake-bound is $\omega \times m$. A similar result can also be established for linearly-covering logic programs of

Arimura and Shinohara [AS94] and for linearly-moded logic programs of Krishna Rao [KR96] if the body length of the clauses is also bounded.

Finally, we give a characterization of learnability with a mistake bound of ordinal α in terms of identification in the limit. We show that the classes of languages learnable in the on-line model with a mistake bound of α are exactly those classes of languages that can be learned in the identification in the limit setting from both positive and negative data by a learner that is Popperian (one which outputs only total programs), is consistent, and makes no more than α mind changes. This result is not very difficult to show if the order of presentation of the positive and negative data is fixed (e.g., the case of canonical informants); however, the proof turns out to be somewhat complicated if the data can be presented in any order. This result establishes a nice bridge between the on-line mistake bound model and the identification in the limit framework.

We now proceed formally.

2 Notation

N denotes the set of natural numbers, $\{0, 1, 2, \ldots\}$. Any unexplained recursion theoretic notation is from [Rog67]. Cardinality of a set S is denoted $\mathrm{card}(S)$. The maximum and minimum of a set are represented by $\max(\cdot)$ and $\min(\cdot)$, respectively. The symbols $\subseteq, \supseteq, \subset, \supset$, and $\emptyset$ respectively stand for subset, superset, proper subset, proper superset, and the emptyset. Λ denotes the empty sequence.

We let π_1 and π_2 denote projections of an ordered pair such that $\pi_1(a, b) = a$ and $\pi_2(a, b) = b$. We denote the fact that a function f is defined on input x by $f(x)\downarrow$. χ_L denotes the characteristic function of the language L (i.e., for $n \in N$, $\chi_L(n) = 1$ if $n \in L$; $\chi_L(n) = 0$ otherwise).

In this paper we employ constructive ordinals to count mistakes in the on-line learning model and to count mind changes in the identification in the limit model. We assume a fixed notation system, O, and partial ordering of ordinal notations as used by, for example, Kleene [Kle38, Rog67, Sac90]. $\preceq, \prec, \succeq$ and $\succ$ on ordinal notations below refer to the partial ordering of ordinal notations in this system. We do not go into the details of the notation system used, but instead refer the reader to [Kle38, Rog67, Sac90, CJS95, FS93]. In the sequel, we are somewhat informal and use $+$, $\times$, and for all $m \in N$ as notation for the same.

A class of languages $\mathcal{L} = \{L_0, L_1, \ldots\}$ is said to be uniformly decidable iff there exists a recursive function f such that $f(i, x) = \chi_{L_i}(x)$.

3 Ordinals as mistake counters

Although our main concern is prediction of $\{0, 1\}$-valued functions, i.e., characteristic functions of languages, we first describe information sequences for functions.

Definition 1

(a) An *information sequence* is an infinite sequence of ordered pairs. A typical variable for an information sequence is I.

(b) The $(n + 1)^{th}$ member of the information sequence I is referred to as $I(n)$ (we start with $I(0)$).

(c) The set of ordered pairs in the information sequence I is denoted $\mathit{content}(I)$.

(d) An information sequence I is said to be for a function f just in case content$(I) = \{(n, f(n)) \mid n \in N\}$.

(e) The *finite initial segment* of an information sequence I of length n is denoted $I[n]$ (so, $I[0] = \Lambda$). We let σ and τ range over finite initial segments. We write $\sigma \subseteq \tau$ if σ is an initial segment of τ. Also, *content*(σ) denotes the set of ordered pairs in σ.

We now describe information sequences for languages.

Definition 2

(a) An information sequence I is said to be for a language L just in case I is an information sequence for χ_L, the characteristic function of L.

I is also referred to as an *informant* for L—modeling the presentation of both positive and negative data for L.

(b) σ *is consistent with* L just in case σ is an initial segment of an informant for L.

(c) The set of all initial segments of informants is denoted InfSEQ.

(d) Let $\sigma \in \mathsf{InfSEQ}$. Then, $\mathsf{PosInfo}(\sigma) = \{x \mid (x, 1) \in \text{content}(\sigma)\}$ and $\mathsf{NegInfo}(\sigma) = \{x \mid (x, 0) \in \text{content}(\sigma)\}$.

Definition 3 A *mistake counter* is a computable mapping $\mathbf{F}$ from InfSEQ into constructive ordinals such that for all informants I, for all $n \in N$, $\mathbf{F}(I[n+1]) \preceq \mathbf{F}(I[n])$.

We are now ready to describe the behavior of a learner in the on-line learning model. In any given trial, the learner has two pieces of information:

- A sequence of instances and information about their membership in the target language. This can be modeled as an initial segment of some informant for the target language.
- An instance whose membership question in the target language the learner is required to conjecture.

Given the above two pieces of information, the learner has to conjecture 0 or 1.

Hence, a learner in the on-line model can be defined as follows:

Definition 4 A learner $\mathbf{M}$ in the on-line model computes a mapping from $\mathsf{InfSEQ} \times N$ into $\{0, 1\}$.

The next definition describes what it means for a learner in the on-line model to successfully learn a language with no more than an ordinal number of mistakes.

Definition 5 $\mathbf{M}$, with associated mistake counter $\mathbf{F}$, is said to $\mathbf{Mistake}_\alpha$-*predict* L just in case the following properties are satisfied:

(a) $\mathbf{F}(\Lambda) = \alpha$.

(b) For all informants I and for all $n \in N$, $\mathbf{M}(I[n], \pi_1(I(n)))\downarrow$.

(c) For all informants I for L and for all $n \in N$, $\mathbf{M}(I[n], \pi_1(I(n))) \neq \pi_2(I(n)) \Rightarrow \mathbf{F}(I[n+1]) \prec \mathbf{F}(I[n])$.

We say that a class of languages $\mathcal{L} \in \mathbf{Mistake}_\alpha$ just in case there exists an $\mathbf{M}$ and a mistake counter $\mathbf{F}$ such that $\mathbf{M}$ with $\mathbf{F}$ $\mathbf{Mistake}_\alpha$-predicts each language in $\mathcal{L}$.

Note that clause (b) in the above definition requires that $\mathbf{M}$ predict for *all inputs* from InfSEQ (not just on initial segments drawn from languages in $\mathcal{L}$). Also, note that the learner $\mathbf{M}$ need not be explicitly told whether its answer for $\pi_1(I(n))$ was right or wrong, since $I[n+1]$ contains the correct answer for $\pi_1(I(n))$.

4 A sufficient condition for ordinal mistake bound

In this section, we present a sufficient condition for when a class of languages belongs to $\mathbf{Mistake}_{\omega \times m}$, for $m \in N$. The condition is a topological property of language classes described in terms of formal systems. We describe this formalism next.

Let U and R be two recursively enumerable sets. Members of U are referred to as *objects* and members of R are refereed to as *expressions*, *rules*, *clauses*, or *productions*. For this section we take a *language* to be a subset of U.[1] A *formal system* is a finite subset of R. We let Γ, with or without decorations, range over formal systems. A *semantic mapping* is a mapping from formal systems to languages. We use **Lang** to denote a typical semantic mapping. A formal system Γ is said to define the language L just in case $\mathbf{Lang}(\Gamma) = L$. We assume that $\mathbf{Lang}(\emptyset) = \emptyset$. One can easily adapt the notion of informants from the previous section to the setting of formal systems. A *language defining framework* is a triple $\langle U, R, \mathbf{Lang}\rangle$, where U, R are recursively enumerable sets and **Lang** is a semantic mapping as above.

We now formally define what it means for a language defining framework to have the property of *effective bounded finite thickness*. A semantic mapping **Lang** is *monotonic* just in case for any two formal systems Γ and Γ', $\Gamma \subseteq \Gamma' \Rightarrow \mathbf{Lang}(\Gamma) \subseteq \mathbf{Lang}(\Gamma')$. In the sequel, we only consider language defining frameworks that are monotonic, and where there exists a recursive function g such that, for all finite $\Gamma \subseteq R$, $x \in U$, $g(\Gamma, x) = 1 \iff x \in \mathbf{Lang}(\Gamma)$. Furthermore, we assume that for all $x \in U$, there exists a finite $\Gamma \subseteq R$ such that $x \in \mathbf{Lang}(\Gamma)$ (otherwise one can just drop such x from U).

Definition 6 Let $\langle U, R, \mathbf{Lang}\rangle$ be a language defining framework such that **Lang** is monotonic. For any $X \subseteq U$, let

$$\mathsf{Gen}_X \stackrel{\text{def}}{=} \{\Gamma \mid \Gamma \subseteq R \ \wedge\ \text{card}(\Gamma) < \infty \ \wedge\ X \subseteq \mathbf{Lang}(\Gamma)\},$$

$$\mathsf{Min}_X \stackrel{\text{def}}{=} \{\Gamma \in \mathsf{Gen}_X \mid (\forall \Gamma' \in \mathsf{Gen}_X)[\Gamma' \not\subset \Gamma]\},$$

and

$$\mathsf{Min}_X^m \stackrel{\text{def}}{=} \{\Gamma \in \mathsf{Min}_X \mid \text{card}(\Gamma) \leq m\}.$$

$\langle U, R, \mathbf{Lang}\rangle$ is said to have *effective m-bounded finite thickness* just in case for all finite $X \subseteq U$, Min_X^m is finite and can be obtained effectively in X (i.e., there are functions, recursive in X, for enumerating Min_X^m, and for finding cardinality of Min_X^m).

$\langle U, R, \mathbf{Lang}\rangle$ is said to have *effective bounded finite thickness* just in case it has effective m-bounded finite thickness for each $m \in N$.

[1] One can easily identify elements of U with members of N.

Note that $\mathsf{Min}_X = \bigcup_m \mathsf{Min}_X^m$. Also, if $\langle U, R, \mathbf{Lang}\rangle$ has effective $(m+1)$-bounded finite thickness, then it has effective m-bounded finite thickness.

In Angluin's definition, a class $\mathcal{L}$ has finite thickness, if any element belongs to at most finitely many languages in $\mathcal{L}$. Bounded finite thickness as defined by Shinohara [Shi91] is a generalization of Angluin's concept of finite thickness. Concept of Effective bounded finite thickness is obtained by imposing the effectiveness constraint on Shinohara's definition.

Intuitively, if $\langle U, R, \mathbf{Lang}\rangle$ has m-effective bounded finite thickness then for any finite $X \subseteq U$, there are at most finitely many *minimal* $\Gamma \subseteq R$ of size $\leq m$ such that $X \subseteq \mathbf{Lang}(\Gamma)$ (here *minimal* is used in the sense that no proper subset Γ' of Γ satisfies $X \subseteq \mathbf{Lang}(\Gamma')$). Moreover, these Γ can be effectively found from X.

Since our interest is only on languages $L \subseteq U$, which can be generated by finite Γ (that is, for some finite Γ, $\mathbf{Lang}(\Gamma) = L$), Min_L would be non-empty for all such L and their subsets.

Proposition 1 *Suppose $X \subseteq X' \subseteq U$, such that $\mathsf{Min}_{X'}$ is nonempty. Then, Min_X is not empty, and for every $\Gamma' \in \mathsf{Min}_{X'}$, there exists a $\Gamma \in \mathsf{Min}_X$ such that $\Gamma \subseteq \Gamma'$.*

Proof. Suppose $\Gamma' \in \mathsf{Min}_{X'}$. Then clearly, $X \subseteq X' \subseteq \mathbf{Lang}(\Gamma')$. Since Γ' is finite, there exists a finite subset Γ of Γ' such that $X \subseteq \mathbf{Lang}(\Gamma)$, but $X \not\subseteq \mathbf{Lang}(\Gamma'')$ for any $\Gamma'' \subset \Gamma$. It follows that $\Gamma \in \mathsf{Min}_X$. ∎

Proposition 2 *Suppose $X \subseteq U$, such that Min_X is nonempty. Then, for any $\Gamma \in \mathsf{Min}_X$, there exists a finite $X' \subseteq X$ such that $\Gamma \in \mathsf{Min}_{X'}$.*

Proof. Proposition is trivial for finite X. So let X be infinite. Let $\Gamma \in \mathsf{Min}_X$. Let $x_0, x_1, \ldots$ be a listing of elements of X. Let $S_i = \{\Gamma' \mid \Gamma' \subseteq \Gamma \ \wedge \ \{x_0, \ldots, x_i\} \subseteq \mathbf{Lang}(\Gamma')\}$. Note that each S_i is nonempty (since Γ belongs to every S_i). Moreover, $S_i \supseteq S_{i+1}$. Thus, $\lim_{i\to\infty} S_i$ converges to a set S. Now, for every $\Gamma' \in S$, $X \subseteq \mathbf{Lang}(\Gamma')$ (by definition of S). Thus, $S = \{\Gamma\}$ (since $\Gamma \in \mathsf{Min}_X$).

Let i be such that $S = S_i$. Hence, it follows that $\{x_0, \ldots, x_i\} \subseteq X \subseteq \mathbf{Lang}(\Gamma)$, and for all $\Gamma' \subset \Gamma$, $\{x_0, \ldots, x_i\} \not\subseteq \mathbf{Lang}(\Gamma')$ (by definition of S_i). It follows that $\Gamma \in \mathsf{Min}_{\{x_0,\ldots,x_i\}}$. ∎

Our first theorem links the notion of effective bounded finite thickness and prediction with ordinal mistake bounds.

Theorem 1 *Fix $m > 0$. Let $\langle U, R, \mathbf{Lang}\rangle$ be a language defining framework with effective m-bounded finite thickness. Let*

$$\mathcal{L}^m \stackrel{\text{def}}{=} \{\mathbf{Lang}(\Gamma) \mid \Gamma \subseteq R \ \wedge \ \mathrm{card}(\Gamma) \leq m\}.$$

Then $\mathcal{L}^m \in \mathbf{Mistake}_{\omega\times m}$.

Our proof of the above result employs some technical machinery which we introduce next.

Let $\mathsf{Pos} \subseteq U$ and $\mathsf{Neg} \subseteq U$ be two disjoint finite sets such that $\mathsf{Pos} \neq \emptyset$. Then let

$Z_i^{\mathsf{Pos},\mathsf{Neg}} \stackrel{\text{def}}{=} \{\Gamma \subseteq R \mid \mathrm{card}(\Gamma) = i \ \wedge \ [\mathsf{Pos} \subseteq \mathbf{Lang}(\Gamma)] \ \wedge \ [\mathsf{Neg} \subseteq U - \mathbf{Lang}(\Gamma)]\}$.

The next lemma and corollary shed light on computation of $Z_i^{\mathsf{Pos},\mathsf{Neg}}$.

Lemma 1 *Suppose $i \in N$. Let $\langle U, R, \mathbf{Lang}\rangle$ be a language defining framework with effective $(i+1)$-bounded finite thickness. Let $\mathsf{Pos} \neq \emptyset$ and Neg be two disjoint finite subsets of U. Suppose $(\forall j \leq i)[Z_j^{\mathsf{Pos},\mathsf{Neg}} = \emptyset]$. Then, $Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}}$ is finite and can be computed effectively from Pos and Neg.*

PROOF. Let Pos, Neg, and i be as given in the hypothesis of the lemma.

We claim that $Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}} \subseteq \{\Gamma \mid \Gamma \in \mathsf{Min}_{\mathsf{Pos}}^{i+1}\}$. To see this, suppose $\Gamma \in Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}}$. Clearly, $\mathsf{Pos} \subseteq \mathbf{Lang}(\Gamma)$. Suppose there exists a $\Gamma' \subset \Gamma$ such that $\mathsf{Pos} \subseteq \mathbf{Lang}(\Gamma')$. Then, clearly, $\mathbf{Lang}(\Gamma') \subseteq \mathbf{Lang}(\Gamma)$. Thus, $\mathsf{Neg} \cap \mathbf{Lang}(\Gamma') \subseteq \mathsf{Neg} \cap \mathbf{Lang}(\Gamma) = \emptyset$. Thus, $\Gamma' \in Z_{\mathrm{card}(\Gamma')}^{\mathsf{Pos},\mathsf{Neg}}$, a contradiction to the hypothesis of the lemma. Thus, for all $\Gamma' \subset \Gamma$, $\mathsf{Pos} \not\subseteq \mathbf{Lang}(\Gamma')$. Thus, $\Gamma \in \mathsf{Min}_{\mathsf{Pos}}^{i+1}$.

It follows that $Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}} = \{\Gamma \in \mathsf{Min}_{\mathsf{Pos}}^{i+1} \mid \mathsf{Neg} \cap \mathbf{Lang}(\Gamma) = \emptyset\}$.

$Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}}$ is finite, since $\mathsf{Min}_{\mathsf{Pos}}^{i+1}$ is finite. Also, since $\mathsf{Min}_{\mathsf{Pos}}^{i+1}$ is obtainable effectively from Pos, it follows that $Z_{i+1}^{\mathsf{Pos},\mathsf{Neg}}$ is obtainable effectively from Pos and Neg. ∎

Corollary 1 *Suppose $m > 0$, $m \in N$. Let $\langle U, R, \mathbf{Lang}\rangle$ be a language defining framework with effective m-bounded finite thickness. Let $\mathsf{Pos} \neq \emptyset$ and Neg be two disjoint finite subsets of U. Then, effectively from Pos, Neg one can determine $i = \min(\{j \mid Z_j^{\mathsf{Pos},\mathsf{Neg}} \neq \emptyset\} \cup \{m+1\})$; furthermore if $i \leq m$, then corrsponding $Z_i^{\mathsf{Pos},\mathsf{Neg}}$ (which is finite) can also be determined effectively.*

PROOF. Note that $\mathbf{Lang}(\emptyset)$ is empty. The corollary now follows by repeated use of Lemma 1 until one finds an i such that $Z_i^{\mathsf{Pos},\mathsf{Neg}} \neq \emptyset$ or discovers that $Z_m^{\mathsf{Pos},\mathsf{Neg}} = \emptyset$. ∎

PROOF. (Theorem 1) We use Lemma 1 and Corollary 1 to prove the theorem.

Fix m. Let I be an informant. Then for $n \in N$, $x \in U$, $\mathbf{M}(I[n], x)$ and $\mathbf{F}(I[n])$ are defined as follows.

Let $\mathsf{Pos} = \mathsf{PosInfo}(I[n])$ and $\mathsf{Neg} = \mathsf{NegInfo}(I[n])$.

If $\mathsf{Pos} = \emptyset$, then $\mathbf{M}(I[n], x) = 0$ and $\mathbf{F}(I[n]) = \omega \times m$.

If $\mathsf{Pos} \neq \emptyset$, then let $j = \min(\{m+1\} \cup \{j' \mid Z_{j'}^{\mathsf{Pos},\mathsf{Neg}} \neq \emptyset\})$. Note that j (and corresponding $Z_j^{\mathsf{Pos},\mathsf{Neg}}$, if $j \leq m$) can be found effectively in $I[n]$, using Corollary 1. Now define $\mathbf{M}(I[n], x)$ and $F(I[n])$ based on the following cases.

If $j > m$, then let $\mathbf{M}(I[n], x) = 0$, $\mathbf{F}(I[n]) = 0$ (output doesn't matter in this case, since I is not an informant for any $L \in \mathcal{L}^m$ — this is so since $Z_i^{\mathsf{Pos},\mathsf{Neg}} = \emptyset$, for $i \leq m$, and thus for all Γ such that $\mathrm{card}(\Gamma) \leq m$, either $\mathsf{Pos} \not\subseteq \mathbf{Lang}(\Gamma)$ or $\mathsf{Neg} \cap \mathbf{Lang}(\Gamma) \neq \emptyset$).

If $j \leq m$, then let $\mathbf{M}(I[n], x) = \chi_{\mathbf{Lang}(\Gamma)}(x)$, where Γ is the lexicographically least element in $Z_j^{\mathsf{Pos},\mathsf{Neg}}$, and let $\mathbf{F}(I[n]) = \omega \times k + \ell$, where $k = m - j$, and $\ell = \mathrm{card}(Z_j^{\mathsf{Pos},\mathsf{Neg}}) - 1$.

It is easy to verify that $\mathbf{M}, \mathbf{F}$ witness that $\mathcal{L}^m \in \mathbf{InfEx}_{\omega \times m}$. ∎

Corollary 2 *Let $\langle U, R, \mathbf{Lang}\rangle$ be a language defining framework with effective bounded finite thickness. For $m > 0$, let*

$$\mathcal{L}^m \stackrel{\mathrm{def}}{=} \{\mathbf{Lang}(\Gamma) \mid \Gamma \subseteq R \ \wedge \ \mathrm{card}(\Gamma) \leq m\}.$$

Then, for all $m > 0$, $\mathcal{L}^m \in \mathbf{Mistake}_{\omega \times m}$.

5 Examples of classes in $\mathbf{Mistake}_{\omega \times n}$

Theorem 1 can be used to show that the class of unions of up to n pattern languages [Wri89] belongs to $\mathbf{Mistake}_{\omega \times n}$ and so does the class of length-bounded formal systems [Shi91] consisting of no more than n clauses. In the present section, we show how Theorem 1 can be employed to establish mistake bound results for classes of minimal models of logic programs. These classes are known to

have bounded finite thickness. It turns out that the proof of bounded finite thickness can easily be modified to show effective bounded finite thickness. However, for the sake of completeness, we present the definitions and the results for two representative classes. We first describe the preliminaries from logic programming; the reader is referred to Lloyd [Llo87] for any unexplained notation.

Let $\Pi, \Sigma, \mathcal{X}$ be mutually disjoint sets such that Π and Σ are finite. Π is the set of *predicate symbols*, Σ is the set of *function symbols*, and $\mathcal{X}$ is the set of *variables*. The arity of a function or a predicate symbol p is denoted $\mathsf{arity}(p)$. The set of terms constructed from the function symbols in Σ and variables in $\mathcal{X}$ is denoted $\mathsf{Terms}(\Sigma, \mathcal{X})$. $\mathsf{Atoms}(\Pi, \Sigma, \mathcal{X})$ denotes the set of atoms formed from predicate symbols in Π and terms in $\mathsf{Terms}(\Sigma, \mathcal{X})$. The set of ground atoms for a predicate symbol p, then is $\mathsf{Atoms}(\{p\}, \Sigma, \emptyset)$; we denote this set by $B(p)$. The size of a term t, denoted $|t|$, is the number of symbols other than punctuation symbols in t. A definite clause is a clause of the form $H \leftarrow A_1 A_2 \ldots A_n$ where H, $A_1, \ldots, A_n$ are atoms; H is called the head and $A_1, \ldots, A_n$ is called the body of the clause. The *body length* of a definite clause is the number of atoms in its body. The *length of a logic program* P, denoted $\mathsf{Length}(P)$, is just the number of clauses in P.

Following the treatment of [KR96], we take the least Herbrand model semantics of logic programs as our monotonic semantic mapping in the present paper. We will refer to the target predicate being learned by the symbol p. It should be noted that the treatment can be generalized to take into account the situation of multiple predicates in an obvious way. Then our language defining frameworks will be of the form $\langle B(p), LP, M_p \rangle$, where LP is the class of Prolog clauses being considered and M_p denotes the semantic mapping such that $M_p(P)$ is the set of all atoms of the target predicate p in the least Herbrand model of P.

We next describe linear Prolog programs introduced by Shapiro [Sha81b].

Definition 7 [Sha81b] A definite clause $p(t_1, \ldots, t_n) \leftarrow q_1(s_{1_1}, \ldots, s_{1_{n_1}}), \ \ldots, \ q_k(s_{k_1}, \ldots, s_{k_{n_k}})$ is called *linear* just in case for each i, $1 \leq i \leq k$, $|t_1\theta| + \cdots + |t_n\theta| \geq |s_{i_1}\theta| + \cdots + |s_{i_{n_i}}\theta|$ for any substitution θ. A logic program P is said to be *linear* just in case each clause in P is linear.

Shinohara [Shi91] showed the following.

Theorem 2 [Shi91] *The class of least Herbrand models of linear Prolog programs is a uniformly decidable family of computable languages.*

Let LC denote the class of all linear clauses and M_p be a semantic mapping such that $M_p(P)$ is the set of all atoms of the target predicate p in the least Herbrand model of P. Then we have the following.

Theorem 3 *The language defining framework $\langle B(p), \mathsf{LC}, M_p \rangle$ has effective bounded finite thickness.*

PROOF. Shinohara's proof of $\langle B(p), \mathsf{LC}, M_p \rangle$ having bounded finite thickness can easily be modified to show that it is effective. ∎

As a corollary of Theorem 1 and above theorem we get

Corollary 3 *For each $n \geq 1$, $\mathcal{L}^n = \{M_p(P) \mid P \in \mathsf{LC} \ \wedge \ \mathsf{Length}(P) \leq n\} \in \mathbf{Mistake}_{\omega \times n}$.*

We note that a similar result can be shown for the class of hereditary logic programs [MSS91, MSS93] and reductive logic programs [KR96].

The above results were for classes of logic programs that did not allow local variables. We now turn our attention to the mistake bound complexity of learning classes of logic programs that allow local variables. We show that the language defining frameworks associated with the class of linearly-covering Prolog programs of Arimura and Shinohara and the class of linearly-moded Prolog programs of Krishna Rao have effective bounded finite thickness if the body length of the clauses is bounded. Since the class of linearly-covering programs are subsumed by the class of linearly-moded programs, we show the result for only the latter class. We first introduce some terminology about parametric size of terms and moded logic programs.

Let $\langle\ \rangle$ denote the empty list.

Definition 8 The *parametric size* of a term t, denoted $\text{Psize}(t)$, is defined inductively as follows:

(a) if t is a variable x then $\text{Psize}(t)$ is the linear expression x;

(b) if t is the empty list, then $\text{Psize}(t)$ is 0;

(c) if $t = f(t_1, \ldots, t_n)$ and $f \in \Sigma - \{\langle\ \rangle\}$, then $\text{Psize}(t)$ is the linear expression $1 + \text{Psize}(t_1) + \cdots + \text{Psize}(t_n)$.

We usually denote a sequence of terms $t_1, \ldots, t_n$ by $\mathbf{t}$. The parametric size of a sequence of terms $t_1, \ldots, t_n$ is the sum $\text{Psize}(t_1) + \cdots + \text{Psize}(t_n)$.

The definition of linearly-moded programs requires the notion of modes associated with each argument in a predicate.

Definition 9 (a) A *mode declaration* for an n-ary predicate p is a mapping from $\{1, \ldots, n\}$ to the set $\{+, -\}$.

(b) Let md be a mode declaration for the predicate p. Then the sets $+(p) = \{j \mid \mathsf{md}(j) = +\}$ and $-(p) = \{j \mid \mathsf{md}(j) = -\}$ are the sets of input and output positions of p, respectively.

If each predicate in a logic program has a unique mode declaration, the program is referred to as a moded program. In dealing with moded programs, it is useful to group together the input and output arguments, i.e., $p(\mathbf{s}; \mathbf{t})$ is an atom with input terms $\mathbf{s}$ and output terms $\mathbf{t}$.

The definition of linearly-moded logic programs requires the following technical notion.

Definition 10 [KR96] Let P be a moded logic program and let J be a mapping from the set of predicates occurring in P to sets of input positions such that $J(p) \subseteq +(p)$ for each predicate p in P. Then for an atom $A = p(\mathbf{s}; \mathbf{t})$, the following linear inequality is denoted $\mathsf{LI}(A, J)$.

$$\Sigma_{i \in J(p)} \text{Psize}(s_i) \geq \Sigma_{j \in -(p)} \text{Psize}(t_j).$$

Intuitively, for any atom $A = p(s; t)$ if we consider the input size of A (restricted by J) as the sum of the parametric size of terms in s which belong to $J(p)$, and the output size of A as the sum of the parametric size of terms in t, then $LI(A, J)$ denotes that the input size (restricted by J) of A is at least as large as the output size of A.

We now define Krishna Rao's notion of what it means for a logic program to be linearly-moded.

Definition 11 [KR96]

(a) Let P be a moded logic program and let J be a mapping from the set of predicates in P to the sets of input positions satisfying $J(p) \subseteq +(p)$ for each predicate p in P. P is said to be *linearly-moded with respect to* J if each clause

$$p_0(\mathbf{s}_0; \mathbf{t}_0) \leftarrow p_1(\mathbf{s}_1; \mathbf{t}_1), \ldots, p_k(\mathbf{s}_k; \mathbf{t}_k)$$

in P satisfies the following two conditions:

(i) $\mathsf{LI}(A_1, J), \ldots, \mathsf{LI}(A_{j-1}, J)$ together imply $\mathrm{Psize}(\mathbf{s}_0) \geq \mathrm{Psize}(\mathbf{s}_j)$, for each $j \geq 1$, and

(ii) $\mathsf{LI}(A_1, J), \ldots, \mathsf{LI}(A_k, J)$ together imply $\mathsf{LI}(A_0, J)$,

where A_j is the atom $p_j(\mathbf{s}_j; \mathbf{t}_j)$ for each $j \geq 0$.

(b) A logic program P is said to be *linearly-moded* just in case it is linearly-moded with respect to some mapping J.

Intuitively, a clause (where the atoms in the body are ordered) being linearly-moded means that, for some J, input size of the head of the clause (restricted by J) is at least as large as the input size of each atom in the body (under the assumption that output size of each of preceding atoms in the body is not larger than the input size (restricted by J) of the corresponding atoms). Furthermore, the input size (restricted by J) of the head is at least as large as the output size of the head (under the assumption that output size of each of the atoms in the body is not larger than the input size (restricted by J) of the corresponding atoms).

We now introduce the language defining framework of linearly-moded clauses. For $k > 0$, let LMC_k denote the set of all linearly-moded clauses of body length at most k. Then the language defining framework associated with linearly-moded clauses is $\langle B(p), \mathsf{LMC}_k, M_p \rangle$.

Theorem 4 [KR96] *For $k \geq 1$, the class of least Herbrand models of logic programs with clauses in LMC_k is a uniformly decidable family of computable languages.*

Theorem 5 *For $k \geq 1$, the language defining framework $\langle B(p), \mathsf{LMC}_k, M_p \rangle$ has effective bounded finite thickness.*

PROOF. Krishna Rao's [KR96] proof of $\langle B(p), \mathsf{LMC}_k, M_p \rangle$ having bounded finite thickness can easily be made effective. ∎

As a consequence of the above theorem and Theorem 1, we have the following corollary.

Corollary 4 *For each $n \geq 1$, for each $k \geq 1$, $\mathcal{L}_k^n = \{M_p(P) \mid P \in \mathsf{LMC}_k \ \wedge \ \mathsf{Length}(P) \leq n\} \in$ $\mathbf{Mistake}_{\omega \times n}$.*

The reader should note that the bound k on the body length of clauses is crucial for the effective bounded finite thickness property. It can be shown that without such a restriction the class of least Herbrand models of length-bounded linearly-moded programs does not have the property of effective bounded finite thickness. Krishna Rao [KR96] has shown that the class of linearly-covering clauses is included in the class of linearly-moded clauses. So the mistake bound learnability result on linearly-moded programs is applicable to linearly-covering programs too.

6 A characterization of Mistake$_\alpha$

We now present a characterization of **Mistake**$_\alpha$ in terms of identification in the limit from both positive and negative data. In this section, we revert back to treating languages as subsets of N. By φ we denote a fixed *acceptable* programming system for the partial computable functions: $N \to N$ [Rog67, MY78]. By φ_i we denote the partial computable function computed by the program with number i in the φ-system. If φ_i is a $\{0,1\}$-valued total function, then we take **Lang**(i) to be $\{x \in N \mid \varphi_i(x) = 1\}$; else **Lang**$(i) = \emptyset$.

The next definition defines a learning machine that learns languages in the identification in the limit framework from both positive and negative data.

Definition 12

(a) A learning machine from informants is an algorithmic mapping from InfSEQ into $N \cup \{?\}$. We again take a typical variable for learning machines in this framework to be **M**.

(b) **M** is said to *converge* on informant I to i (written: **M**(I) converges to i or **M**$(I)\downarrow = i$) just in case for all but finitely many n, **M**$(I[n]) = i$.

An output of i is interpreted as the conjecture that the characteristic function of the target language is φ_i. A conjecture of "?" by a machine is interpreted as "no guess at this moment." This is useful to avoid biasing the number of mind changes of a machine. For this paper, we assume, without loss of generality, that $\sigma \subseteq \tau$ and **M**$(\sigma) \neq ?$ implies **M**$(\tau) \neq ?$.

We next introduce ordinals as mind change counters in the context of identification in the limit from informants.

Definition 13 **F**, an algorithmic mapping from InfSEQ into ordinal notations, is an *ordinal mind change counter function* just in case $(\forall \sigma \subseteq \tau)[\mathbf{F}(\sigma) \succeq \mathbf{F}(\tau)]$.

The next definition describes what it means for a machine to identify a class of languages in the limit from informants with an ordinal mind change bound.

Definition 14 [FS93, AJS97] Let α be an ordinal notation.

(a) We say that **M**, with associated ordinal mind change counter function **F**, **InfEx**$_\alpha$-*identifies* an informant I just in case the following three conditions hold:

(i) **M**$(I)\downarrow = i$ and φ_i is the characteristic function of the language whose informant is I.

(ii) $\mathbf{F}(\Lambda) = \alpha$, and

(iii) $(\forall n)[[? \neq \mathbf{M}(I[n]) \ \wedge \ \mathbf{M}(I[n]) \neq \mathbf{M}(I[n+1])] \Rightarrow \mathbf{F}(I[n]) \succ \mathbf{F}(I[n+1])]$.

(b) **M**, with associated ordinal mind change counter function **F**, **InfEx**$_\alpha$-*identifies* L (written: $L \in$ **InfEx**$_\alpha(\mathbf{M}, \mathbf{F})$) just in case **M**, with associated ordinal mind change counter function **F**, **InfEx**$_\alpha$-identifies each informant for L.

(c) **InfEx**$_\alpha = \{\mathcal{L} \mid (\exists \mathbf{M}, \mathbf{F})[\mathcal{L} \subseteq \mathbf{InfEx}_\alpha(\mathbf{M}, \mathbf{F})]\}$.

We are now almost ready to present our characterization of **Mistake**$_\alpha$ in terms of **InfEx**$_\alpha$, but need a few properties of learning machines.

Definition 15 [CJNM94] A learning machine $\mathbf{M}$ is *Popperian* just in case for all $\sigma \in \mathsf{InfSEQ}$ (including the $\sigma = \Lambda$), $\mathbf{M}(\sigma)$ is a total program, i.e., $\varphi_{\mathbf{M}(\sigma)}$ is a total function.

Definition 16 [WZ95] A learning machine $\mathbf{M}$ is *consistent* just in case for all $\sigma \in \mathsf{InfSEQ}$, $\mathsf{PosInfo}(\sigma) \subseteq \mathbf{Lang}(\mathbf{M}(\sigma))$ and $\mathsf{NegInfo}(\sigma) \cap \mathbf{Lang}(\mathbf{M}(\sigma)) = \emptyset$.

We now state our characterization:

Theorem 6 *$\mathcal{L} \in \mathbf{Mistake}_\alpha$ iff a Popperian and consistent learning machine $\mathbf{InfEx}_\alpha$-identifies $\mathcal{L}$.*

Proof. We first show that if a Popperian and consistent learning machine $\mathbf{InfEx}_\alpha$-identifies $\mathcal{L}$, then $\mathcal{L} \in \mathbf{Mistake}_\alpha$. Suppose a Popperian and consistent machine $\mathbf{M}$, with mind change counter $\mathbf{F}$, $\mathbf{InfEx}_\alpha$-identifies $\mathcal{L}$. Consider a predictor $\mathbf{M}'$ defined as follows: $\mathbf{M}'(I[n], x) = \varphi_{\mathbf{M}(I[n])}(x)$. Then it is easy to verify that $\mathbf{M}'$ with mind change counter $\mathbf{F}$, $\mathbf{Mistake}_\alpha$-predicts each language in $\mathcal{L}$.

We next show that if $\mathcal{L} \in \mathbf{Mistake}_\alpha$, then there exists a Popperian and consistent learning machine which $\mathbf{InfEx}_\alpha$-identifies $\mathcal{L}$. Suppose predictor $\mathbf{M}$, with mind change counter $\mathbf{F}$, $\mathbf{Mistake}_\alpha$-predicts each language in $\mathcal{L}$. We will then define a Popperian and consistent machine $\mathbf{M}'$ and associated ordinal mind change counter $\mathbf{F}'$ witnessing $\mathcal{L} \in \mathbf{InfEx}_\alpha$. For any I, we define a partial function h_I and an informant I' as follows. It will be the case that the domain of h_I is either N or an initial segment of N. It can be easily seen that $h_I(n)$ (if defined) can be computed effectively from I. Similarly, if $h_I(n)$ is defined then $I'[h_I(n)]$ can be computed effectively from I.

Let $h_I(0) = 0$. $I'[h_I(0)] = \Lambda$. For $n \geq 0$, $h_I(n+1)$ and $I'[h_I(n+1)]$ is defined in stages as follows. Go to stage 0.

Stage n:

(* $h_I(n)$, and $I'[h_I(n)]$ have already been defined *)

(* $h_I(m)$, for $m > n$, has not yet been defined *)

1. Case 1: There exists an $m \geq h_I(n)$, such that $I(m) = (x, y)$, and $\mathbf{M}(I'[h_I(n)], x) \neq y$.
   Let $h_I(n+1)$ be 1+(the least such m). Let $I'(h_I(n)) = I(h_I(n+1) - 1)$, $I'(w) = I(w-1)$, for $h_I(n) < w < h_I(n+1)$.
   Go to stage $n+1$.
2. Case 2: Not Case 1.
   In this case $h_I(m)$ is undefined for all $m \geq n+1$. Let $I'(w) = I(w)$, for $w \geq h_I(n)$.
   No more stages are executed.

End Stage n

It is easy to verify by induction that:
(a) If $h_I(n)$ is defined then $I'[h_I(n)]$ is a rearrangement of $I[h_I(n)]$.
(b) If $h_I(n+1)$ is defined then, $\mathbf{M}(I'[h_I(n)], \pi_1(I'(h_I(n)))) \neq \pi_2(I'(h_I(n)))$.
(c) For all m, maximum n such that $h_I(n) \leq m$, can be effectively determined from m.

Now define $\mathbf{M}'$ and $\mathbf{F}'$ as follows:

$\mathbf{M}'(I[m]) = \mathsf{Prog}(I'[h_I(n)])$, and $\mathbf{F}'(I[m]) = \mathbf{F}(I'[h_I(n)])$, where n is the maximum n' such that $h_I(n') \leq m$, and Prog is a computable function such that

$$\varphi_{\mathsf{Prog}(\sigma)}(x) = \begin{cases} y, & \text{if } (x,y) \in \text{content}(\sigma); \\ \mathbf{M}(\sigma, x), & \text{otherwise.} \end{cases}$$

It is easy to verify that $\mathbf{M}'$ is Popperian and consistent for all I. Suppose now that I is an informant for $L \in \mathcal{L}$. Mind changes for $\mathbf{M}'$ on I only happen at the positions $I[h_I(n)]$ (that is, $\mathbf{M}'(I[z]) \neq \mathbf{M}'(I[z+1])$ implies $z+1 = h_I(n)$, for some $n > 0$).

Suppose $h_I(n+1)\downarrow$. Note that $\mathbf{F}'(I[m]) = \mathbf{F}'(I[h_I(n)])$, for $h_I(n) \leq m < h_I(n+1)$. Also, since, $h_I(n+1)\downarrow$ implies that $\mathbf{M}(I'[h_I(n)], \pi_1(I'(h_I(n)))) \neq \pi_2(I'(h_I(n)))$, it follows that $\mathbf{F}'(I[h_I(n+1)]) = \mathbf{F}(I'[h_I(n+1)]) \prec \mathbf{F}(I'[h_I(n)]) = \mathbf{F}'(I[h_I(n)]) = \mathbf{F}'(I[h_I(n+1)-1])$. The theorem follows. ∎

7 Conclusion

We employed the notion of constructive ordinals to extend the applicability of the on-line mistake bound learning model to expressive language classes. We gave a sufficient condition for learnability in this model. We also presented a characterization of on-line mistake bound learning model in terms of identification in the limit from both positive and negative data. This result bridges the gap between the on-line model and the identification in the limit model. In this connection, we also note the work of Kaufmann and Stephan [KS97].

8 Acknowledgements

The research of Arun Sharma is supported in part by the Australian Research Council Grant A49803051. The research of Sanjay Jain was supported in part by NUS grant number RP3992710. A preliminary version of this paper was presented at COLT'99; we thank the anonymous referees of COLT'99 for several useful comments.

References

[AFS96] A. Ambainis, R. Freivalds, and C. Smith. General inductive inference types based on linearly-ordered sets. In *Proceedings of Symposium on Theoretical Aspects of Computer Science*, volume 1046 of *Lecture Notes in Computer Science*, pages 243–253. Springer-Verlag, 1996.

[AJS97] A. Ambainis, S. Jain, and A. Sharma. Ordinal mind change complexity of language identification. In S. Ben-David, editor, *Third European Conference on Computational Learning Theory*, volume 1208 of *Lecture Notes in Artificial Intelligence*, pages 301–315. Springer-Verlag, 1997.

[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.

[AS94] H. Arimura and T. Shinohara. Inductive inference of Prolog programs with linear data dependency from positive data. In H. Jaakkola, H. Kangassalo, T. Kitahashi,

and A. Markus, editors, *Proc. Information Modelling and Knowledge Bases V*, pages 365–375. IOS Press, 1994.

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

[CJNM94] J. Case, S. Jain, and S. Ngo Manguelle. Refinements of inductive inference by Popperian and reliable machines. *Kybernetika*, 30:23–52, 1994.

[CJS95] J. Case, S. Jain, and M. Suraj. Not-so-nearly-minimal-size program inference. In K. Jantke and S. Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 77–96. Springer-Verlag, 1995.

[FS93] R. Freivalds and C. Smith. On the role of procrastination in machine learning. *Information and Computation*, pages 237–271, 1993.

[JS97] S. Jain and A. Sharma. Elementary formal systems, intrinsic complexity, and procrastination. *Information and Computation*, 132:65–84, 1997.

[JS99] S. Jain and A. Sharma. Mind change complexity of learning logic programs. In P. Fischer and H.U. Simon, editors, *Fourth European Conference on Computational Learning Theory*, volume 1572 of *Lecture Notes in Artificial Intelligence*, pages 198–213. Springer-Verlag, 1999.

[Kle38] S. Kleene. Notations for ordinal numbers. *Journal of Symbolic Logic*, 3:150–155, 1938.

[KR96] M. Krishna Rao. A class of Prolog programs inferable from positive data. In A. Arikawa and A. Sharma, editors, *Algorithmic Learning Theory: Seventh International Workshop (ALT '96)*, volume 1160 of *Lecture Notes in Artificial Intelligence*, pages 272–284. Springer-Verlag, 1996.

[KS97] S. Kaufmann and F. Stephan. Resource bounded next value and explanatory identification: learning automata, patterns and polynomials on-line. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 263–274. ACM Press, 1997.

[Lit88] N. Littlestone. Learning quickly when irrelevant attributes abound - a new linear threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[Llo87] J.W. Lloyd. *Foundation of Logic Programming (Second Edition)*. Springer, New York, 1987.

[MSS91] S Miyano, A. Shinohara, and T. Shinohara. Which classes of elementary formal systems are polynomial-time learnable? In *Proceedings of the Second Workshop on Algorithmic Learning Theory*, pages 139–150, 1991.

[MSS93] S. Miyano, A. Shinohara, and T. Shinohara. Learning elementary formal systems and an application to discovering motifs in proteins. Technical Report RIFIS-TRCS-37, RIFIS, Kyushu University, 1993.

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

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

[Sac90] G. Sacks. *Higher Recursion Theory.* Springer-Verlag, 1990.

[Sha81a] E. Shapiro. An algorithm that infers theory from facts. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, pages 446–451, 1981.

[Sha81b] E. Shapiro. Inductive inference of theories from facts. Technical Report 192, Computer Science Department, Yale University, 1981.

[Shi91] T. Shinohara. Inductive inference of monotonic formal systems from positive data. *New Generation Computing*, 8:371–384, 1991.

[SSV97] A. Sharma, F. Stephan, and Y. Ventsov. Generalized notions of mind change complexity. In *Proceedings of the Tenth Annual Conference on Computational Learning Theory*, pages 96–108. ACM Press, 1997.

[Wri89] K. Wright. Identification of unions of languages drawn from an identifiable class. In R. Rivest, D. Haussler, and M. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory*, pages 328–333. Morgan Kaufmann, 1989.

[WZ95] R. Wiehagen and T. Zeugmann. Learning and consistency. In K. P. Jantke and S. Lange, editors, *Algorithmic Learning for Knowledge-Based Systems*, volume 961 of *Lecture Notes in Artificial Intelligence*, pages 1–24. Springer-Verlag, 1995.