Strong Monotonic and Set Driven Inductive Inference

Sanjay Jain
Department of Information Systems and Computer Science
National University of Singapore
Singapore 0511, Republic of Singapore
Email: sanjay@iscs.nus.sg

February 22, 2011

Abstract

In an earlier paper, Kinber and Stephan posed an open problem about whether every class of languages, which can be identified strong monotonically, can also be identified by a set-driven machine. We solve this question in this paper. The answer to the question depends on whether the machines are required to be total or not! The solution of this result uncovers a finer gradation of the notion of set-drivenness.

1 Introduction

Consider the identification of formal languages from positive data. A machine is fed all the strings and no nonstrings of a language L, in any order, one string at a time. The machine, as it is receiving strings of L, outputs a sequence of grammars. The machine is said to identify L just in case the sequence of grammars converges to a grammar for L. This is essentially the paradigm of identification in the limit (called **TxtEx**-identification) introduced by Gold [Gol67].

Since only strings belonging to the language are available, if a learning machine conjectures a grammar for some superset of the target language, it may not be "rational" for the machine to revise this conjecture as data about the complement of the language is not available. This is the problem of overgeneralization in learning formal languages from positive data. To address this problem Jantke [Jan90] introduced the notion of strong monotonic identification, in which a machine is only allowed to output 'improvements' of its previous conjectures in the sense that each later conjectured grammar is for a language which contains languages enumerated by earlier conjectured grammars (see formal definition in Section 2).

Another, issue in language identification is set-drivenness. This constraint requires that the machine base its conjecture solely on the strings it has seen and not on the order in which they appear (or on whether the strings have appeared repeatedly in the text) (see formal definition in Section 2).

In an earlier paper [KS95], Kinber and Stephan posed an open problem about whether every class of languages, which can be identified strong monotonically, can also be identified by a set-driven machine. We solve this question in this paper. The answer to the question depends on whether the machines are required to be total or not! We show that there exists a class of languages which can be **TxtEx**-identified by a strong monotonic machine, but which cannot be **TxtEx**-identified by any total set-driven machine. On the other hand we show that every class of languages which can be **TxtEx**-identified by some strong monotonic machine can also be **TxtEx**-identified by some set-driven machine (this machine however may not be total). We now proceed formally. Section 2 describes the notation and the identification paradigms. Section 3 gives the results.

2 Preliminaries

Any unexplained notion is from [Rog67]. N denotes the set of natural numbers. i, j, k, m, n, p, s, t, x, with or without decorations (decorations are subscripts, superscripts, primes and the like), range over N. $\in, \subseteq, \subset, \supseteq, \supset$ denote membership, subset, proper subset, superset and proper superset relationship for sets. $\emptyset$ denotes empty set. S, X, with or without decorations, range over sets. $\mathrm{card}(S)$ denotes the cardinality of set S. $\min(S)$ and $\max(S)$ respectively denote the minimum and maximum of set S. For an infinite set S, we let $\max(S) = \infty$. Also, by convention, $\min(\emptyset) = \infty$ and $\max(\emptyset) = 0$.

η, with or without decorations, ranges over partial recursive functions. $\mathrm{domain}(\eta)$ and $\mathrm{range}(\eta)$ denote the domain and range of partial function η. f, g, h, with or without decorations, range over total recursive functions.

$\langle \cdot, \cdot \rangle$ denotes a pairing function, a computable bijective mapping from $N \times N$ onto N.

φ denotes a standard acceptable programming system. φ_i denotes the partial function computed by the i-th program in φ system. $\mathcal{R}$ denotes the set of all total computable functions. Φ denotes a Blum complexity measure [Blu67] for programming system φ. W_i denotes the set domain(φ_i). Intuitively, W_i denotes the i-th recursively enumerable (r.e.) set. We let $\mathcal{E}$ denote the set of all r.e. sets. We let $W_i^t = \{x < t \mid \Phi_i(x) < t\}$.

For an overview of the field of inductive inference of languages we refer the reader to [OSW86, CL82, KW80]. We only discuss the portion directly needed for the paper.

A *sequence* is a mapping from an initial segment of N to $N \cup \{\#\}$. We denote the set of all finite sequences by SEQ. We let σ, τ, γ, with or without decorations, range over SEQ. $|\sigma|$ denotes the length of sequence σ. A text is a mapping from N to $N \cup \{\#\}$. The initial sequence of T with length n is denoted by $T[n]$. content(T) denotes the set of natural numbers in the range of T. Similarly content(σ) denotes the set of natural numbers in the range of σ. A text T is for a language L, iff content(T) $= L$. Intuitively, a text T for L denotes a listing of elements L, where # denotes pauses in such a listing. The need for # comes mainly for empty languages, since the only text for empty language is a sequence of #'s. We say that $\sigma \subseteq \tau$, if σ is an initial sequence of τ. In this case we also say that τ is an extension of σ.

A *learning machine* is a (possibly partial) algorithmic mapping from SEQ to N, Where the output of the learning machines is interpreted as a grammar in some standard acceptable numbering. We usually refer to learning machines as just machines. We let $\mathbf{M}$, with or without decorations, range over learning machines.

We say that $\mathbf{M}$ on T converges or $\mathbf{M}(T)$ converges (written $\mathbf{M}(T)\downarrow$) iff there exists an i such that, for all but finitely many n, $\mathbf{M}(T[n]) = i$. Otherwise we say that $\mathbf{M}(T)$ diverges (written $\mathbf{M}(T)\uparrow$). In case $\mathbf{M}(T)\downarrow$, we say that $\mathbf{M}(T) = i$, for the unique i such that, for all but finitely many n, $\mathbf{M}(T[n]) = i$.

Definition 1 [Gol67]

1. We say that $\mathbf{M}$ **TxtEx**-*identifies* L (written: $L \in \mathbf{TxtEx}(\mathbf{M})$), iff ($\forall$ Texts T for L)[$\mathbf{M}(T)\downarrow \ \wedge \ W_{\mathbf{M}(T)} = L$].
2. We say that $\mathbf{M}$ **TxtEx**-identifies a class $\mathcal{L}$ of languages, iff $\mathbf{M}$ **TxtEx**-identifies each language in $\mathcal{L}$.
3. $\mathbf{TxtEx} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})]\}$.

From any machine $\mathbf{M}$ one can effectively construct a total machine $\mathbf{M}'$ such that $\mathbf{TxtEx}(\mathbf{M}) \subseteq \mathbf{TxtEx}(\mathbf{M}')$ (see for example [OSW86]). Thus requiring machines to be total does not restrict the class **TxtEx**.

The proof presented in this paper depends on the technical notion of locking sequence. The next two definitions and a lemma introduce this concept.

Definition 2 [Ful85, Ful90] σ is a *stabilizing sequence* for $\mathbf{M}$ on L iff
(a) content(σ) $\subseteq L$, and
(b) ($\forall \tau \mid \sigma \subseteq \tau \ \wedge$ content(τ) $\subseteq L$)[$\mathbf{M}(\sigma) = \mathbf{M}(\tau)$].

Definition 3 [BB75] σ is a *locking sequence* for $\mathbf{M}$ on L, iff σ is a stabilizing sequence for $\mathbf{M}$ on L and $W_{\mathbf{M}(\sigma)} = L$.

The following locking sequence lemma was proved by Blum and Blum [BB75].

Lemma 4 *[BB75] Suppose* $\mathbf{M}$ **TxtEx**-*identifies* L. *Then there exists a stabilizing sequence for* $\mathbf{M}$ *on* L. *Moreover, every stabilizing sequence for* $\mathbf{M}$ *on* L *is a locking sequence for* $\mathbf{M}$ *on* L.

We now consider strong monotonic and set-driven machines.

Definition 5 [Jan90] $\mathbf{M}$ is *strong monotonic* iff $(\forall \sigma, \tau \mid \sigma \subseteq \tau)[W_{\mathbf{M}(\sigma)} \subseteq W_{\mathbf{M}(\tau)}]$.

Definition 6 [Jan90] $\mathbf{SMON} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathbf{M}$ is strong monotonic and $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})]\}$.

It is easy to show that for any strong monotonic machine $\mathbf{M}$ there exists a total strong monotonic machine $\mathbf{M}'$ such that $\mathbf{TxtEx}(\mathbf{M}) \subseteq \mathbf{TxtEx}(\mathbf{M}')$. Thus the class **SMON** is not restricted by requiring learning machines to be total.

Also note that if $\mathbf{M}$ is strong monotonic and $\mathbf{M}$ **TxtEx**-identifies L, then, for all σ such that $\text{content}(\sigma) \subseteq L$, $W_{\mathbf{M}(\sigma)} \subseteq L$.

Definition 7 [OSW82] $\mathbf{M}$ is *set-driven* iff $(\forall \sigma, \tau \mid \text{content}(\sigma) = \text{content}(\tau))[\mathbf{M}(\sigma)\downarrow = \mathbf{M}(\tau)\downarrow$ or $\mathbf{M}(\sigma)\uparrow = \mathbf{M}(\tau)\uparrow]$.

Thus the output of a set-driven machine depends only on the content of its input. Note that unlike strong monotonicity or **TxtEx**-identification, identification by set driven machines may be restricted if we require machines to be total. (This follows as a corollary to our results). Thus we have two versions of set-driven identification.

Definition 8

1. $\mathbf{SDT} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathbf{M}$ is total and set-driven and $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})]\}$.
2. $\mathbf{SDNT} = \{\mathcal{L} \subseteq \mathcal{E} \mid (\exists \mathbf{M})[\mathbf{M}$ is set-driven and $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M})]\}$.

In general set-driven constraint restricts **TxtEx**-identification (see [SR84, Ful85]). However if one only considers language classes which do not contain finite languages, set-driven constraint is not a restriction (see for example [OSW86, Ful85]). Our result regarding total set-driven machines does not change even if one considers the class version of totality, i.e. one only requires the set-driven machine to be defined on σ such that $\text{content}(\sigma) \subseteq L$, for L **TxtEx**-identified by the machine.

3 Results

3.1 Set-driven identification by total learning machines

Theorem 9 $\mathbf{SMON} - \mathbf{SDT} \neq \emptyset$.

PROOF. Let $\mathbf{M}_0, \mathbf{M}_1, \ldots$ denote an algorithmic listing of all the learning machines.

Let $L_i = \{\langle i, x\rangle \mid x \in N\}$. Let $S_i^n = \{\langle i, x\rangle \mid x < n\}$.

Let T_i be a text for L_i, such that content$(T_i[n]) = S_i^n$. Intuitively, T_i just lists the elements of L_i in a canonical order.

Now let $\mathcal{L}_1 = \{L_i \mid i \in N \ \wedge \ \mathbf{M}_i \text{ total } \ \wedge (\forall n)[\mathbf{M}_i(T_i[n]) \neq \mathbf{M}_i(T_i[n+1])]\}$.

$\mathcal{L}_2 = \{S_i^m \mid i \in N \ \wedge \ \mathbf{M}_i \text{ total } \ \wedge (\exists n)[\mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])] \ \wedge \ m = \min(\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\})\}$.

$\mathcal{L}_3 = \{S_i^{m+1} \mid i \in N \ \wedge \ \mathbf{M}_i \text{ total } \ \wedge (\exists n)[\mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])] \ \wedge \ m = \min(\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\})\}$.

$\mathcal{L} = \mathcal{L}_1 \cup \mathcal{L}_2 \cup \mathcal{L}_3 \cup \{N\}$.

Claim 10 $\mathcal{L} \notin \mathbf{SDT}$.

PROOF. Suppose by way of contradiction that $\mathbf{M}_i$ is total, set-driven, and $\mathcal{L} \subseteq \mathbf{TxtEx}(\mathbf{M}_i)$. We consider two cases.

Case 1: There exists an n such that $\mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])$.

Let $m = \min(\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\})$. Now both content$(T_i[m]) = S_i^m$ and content$(T_i[m+1]) = S_i^{m+1}$ belong to $\mathcal{L}$. However, since $\mathbf{M}_i$ is set-driven and $\mathbf{M}_i(T_i[m]) = \mathbf{M}_i(T_i[m+1])$, we have that $\mathbf{M}_i$ does not **TxtEx**-identify at least one of S_i^m and S_i^{m+1}.

Case 2: For all n, $\mathbf{M}_i(T_i[n]) \neq \mathbf{M}_i(T_i[n+1])$.

In this case $L_i \in \mathcal{L}$, T_i is a text for L_i, but $\mathbf{M}_i(T_i)\uparrow$. Thus $\mathbf{M}_i$ does not **TxtEx**-identify L_i.

From the above cases we have that $\mathcal{L} \notin \mathbf{SDT}$. □

Claim 11 $\mathcal{L} \in \mathbf{SMON}$.

PROOF. Let p_i be a grammar, obtained effectively from i, such that $W_{p_i} = \{\langle i, x\rangle \mid (\forall n \leq x)[\mathbf{M}_i(T_i[n])\downarrow \neq \mathbf{M}_i(T_i[n+1])\downarrow]\}$.

Suppose $\mathbf{M}_i$ is total. Then the following property of W_{p_i} is easy to verify: if $\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\} = \emptyset$, then $W_{p_i} = L_i$; otherwise $W_{p_i} = S_i^m$, where $m = \min(\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\})$. Also note that if $L \in \mathcal{L}$ is such that $L \neq \emptyset$ and $L \subseteq L_i$, then either $L = W_{p_i}$, or $W_{p_i} \subset L = S_i^{m+1}$, where $m = \min(\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\})$. We use these facts to construct a strong monotonic machine $\mathbf{M}$ which **TxtEx**-identifies $\mathcal{L}$.

Consider the following machine $\mathbf{M}$. Let G_N denote a grammar for N. Let $G_\emptyset$ denote a grammar for $\emptyset$. Let G_i^m denote a grammar, obtained effectively from i and m, for S_i^m.

Let $X_i^s = \{n \leq s \mid (\forall m \leq n+1)[\mathbf{M}_i(T_i[m])\downarrow \text{ in} \leq s \text{ steps }] \wedge \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\}$. Intuitively, for total $\mathbf{M}_i$, X_i^s attempts to approximate the set $\{n \mid \mathbf{M}_i(T_i[n]) = \mathbf{M}_i(T_i[n+1])\}$ from below.

$$\mathbf{M}(\sigma) = \begin{cases} G_{\emptyset}, & \text{if content}(\sigma) = \emptyset; \\ p_i, & \text{if content}(\sigma) \neq \emptyset \text{ and content}(\sigma) \subseteq L_i \text{ and} \\ & X_i^{|\sigma|} = \emptyset; \\ p_i, & \text{if content}(\sigma) \neq \emptyset \text{ and content}(\sigma) \subseteq L_i \text{ and} \\ & X_i^{|\sigma|} \neq \emptyset \text{ and content}(\sigma) \subseteq S_i^m, \text{ where} \\ & m = \min(X_i^{|\sigma|}); \\ G_i^{m+1}, & \text{if content}(\sigma) \neq \emptyset \text{ and content}(\sigma) \subseteq L_i \text{ and} \\ & X_i^{|\sigma|} \neq \emptyset \text{ and content}(\sigma) \not\subseteq S_i^m, \text{ where} \\ & m = \min(X_i^{|\sigma|}); \\ G_N, & \text{otherwise.} \end{cases}$$

From the discussion above it is easy to see that $\mathbf{M}$ is strong monotonic and **TxtEx**-identifies each language in $\mathcal{L}$. □

The theorem follows from the above two claims. ∎

3.2 Set-driven identification by non-total learning machines

Theorem 12 **SMON** $\subseteq$ **SDNT**.

Proof. Suppose a machine $\mathbf{M}$ is given such that $\mathbf{M}$ is strong monotonic. Without loss of generality assume that $\mathbf{M}$ is total. We now construct a machine $\mathbf{M}'$ such that, $\mathbf{M}'$ is set-driven (possibly non-total) and $\mathbf{M}'$ **TxtEx**-identifies each language **TxtEx**-identified by $\mathbf{M}$.

We assume some well-ordering of elements of SEQ; thus we can talk about the least element of SEQ satisfying certain constraints.

We say that L is closed with respect to σ and $\mathbf{M}$, if the following properties are satisfied:
(a) content$(\sigma) \subseteq L$.
(b) $(\forall \tau \mid \text{content}(\tau) \subseteq L)[W_{\mathbf{M}(\tau)} \subseteq L]$.

Let closure$(\sigma, \mathbf{M})$, denote the smallest (with respect to containment) language which is closed with respect to σ and $\mathbf{M}$. Note that a grammar for closure$(\sigma, \mathbf{M})$ can be effectively obtained from σ and $\mathbf{M}$. Let gram be a computable function such that $W_{\text{gram}(\sigma,\mathbf{M})} =$ closure$(\sigma, \mathbf{M})$ (such a function clearly exists by s-m-n theorem [Rog67]).

Our construction of $\mathbf{M}'$ depends on the following property of closure.

Lemma 13 *Suppose* $L \in$ **TxtEx**$(\mathbf{M})$ *and* content$(\sigma) \subseteq L$. *Then* closure$(\sigma, \mathbf{M}) \subseteq L$.

Proof. Since $\mathbf{M}$ is strong monotonic, for all τ such that content$(\tau) \subseteq L$, we have that $W_{\mathbf{M}(\tau)} \subseteq L$. Lemma now follows from the definition of closure$(\sigma, \mathbf{M})$. □

We will now give a set-driven (potentially non-total) machine $\mathbf{M}'$ which **TxtEx**-identifies each language **TxtEx**-identified by $\mathbf{M}$. The proof that $\mathbf{M}'$ **TxtEx**-identifies each language **TxtEx**-identified by $\mathbf{M}$ uses Lemma 13.

Begin $\mathbf{M}'(\sigma)$

1. Let $S = \text{content}(\sigma)$.
2. Search for τ such that 2.1, 2.2 and 2.3 are satisfied.
   2.1. For all $\tau' < \tau$ (in the well ordering of SEQ), either $\text{content}(\tau') \not\subseteq S$, or τ' is not a stabilizing sequence for $\mathbf{M}$ on S.
   2.2. $\text{content}(\tau) \subseteq S$.
   2.3. $(\exists t)[S \subseteq W^t_{\mathbf{M}(\tau)}$ and $(\forall \gamma \mid \tau \subseteq \gamma \ \wedge \ \text{content}(\gamma) \subseteq S \ \wedge \ |\gamma| \leq t)[\mathbf{M}(\tau) = \mathbf{M}(\gamma)]]$.
   Note that a τ satisfying 2.1, 2.2, and 2.3 may not exist. If no τ satisfying 2.1, 2.2 and 2.3 exists then $\mathbf{M}'(\sigma)\uparrow$; otherwise choose one such τ for step 3 (note that if such a τ exists, then one such τ can be found algorithmically).
3. Output $\text{gram}(\tau, \mathbf{M})$.
   Note that $\text{gram}(\tau, \mathbf{M})$ is a grammar for $\text{closure}(\tau, \mathbf{M})$. Thus from property 2.3 above we have $\text{content}(\sigma) \subseteq W_{\mathbf{M}'(\sigma)}$.

End $\mathbf{M}'(\sigma)$

It is easy to see that $\mathbf{M}'$ defined above is set-driven. Suppose $L \in \mathbf{TxtEx}(\mathbf{M})$ and T is a text for L. Clearly, by Lemma 13 we have that, for all n, $W_{\mathbf{M}'(T[n])} \subseteq L$. We thus just need to show that $\mathbf{M}'(T)\downarrow$ and $W_{\mathbf{M}'(T)} \supseteq L$. We consider two cases.
Case 1: L is finite.

Let n be such that $L = \text{content}(T[n])$. Now clearly, $\mathbf{M}'(T)\downarrow = \mathbf{M}'(T[n])$ (since $\mathbf{M}'$ is set-driven). Let τ be as found in step 2 of $\mathbf{M}'(T[n])$. Note that such a τ must exist, since $\mathbf{M}$ **TxtEx**-identifies L: the least locking sequence for $\mathbf{M}$ on L, satisfies 2.1, 2.2, and 2.3. Now by property 2.3. checked in step 2 of the construction of $\mathbf{M}'$ we have, $L = \text{content}(T[n]) \subseteq \text{closure}(\tau, \mathbf{M}) = W_{\mathbf{M}'(T[n])}$. Thus from the discussion before the case analysis, we have that $\mathbf{M}'$ **TxtEx**-identifies L.

Case 2: L is infinite.

Suppose σ is the least stabilizing (locking) sequence (in the well-ordering of SEQ) for $\mathbf{M}$ on L. We claim that for all but finitely many n, $\mathbf{M}'(T[n]) = \text{gram}(\sigma, \mathbf{M})$. Since $\text{closure}(\sigma, \mathbf{M}) \supseteq L$, using the discussion before the case analysis, we have that $\mathbf{M}'$ **TxtEx**-identifies L.

So let m be so large that the following are satisfied.
(a) $\text{content}(\sigma) \subseteq \text{content}(T[m])$.
(b) For all $\sigma' < \sigma$, such that $\text{content}(\sigma') \subseteq L$, there exists an extension σ'' of σ' such that

$\mathbf{M}(\sigma') \neq \mathbf{M}(\sigma'')$, $\text{content}(\sigma'') \subseteq \text{content}(T[m])$ and
$\text{content}(T[m]) \not\subseteq W^{|\sigma''|}_{\mathbf{M}(\sigma')}$.

Note that such an m exists since each $\sigma' < \sigma$ is not a stabilizing sequence for $\mathbf{M}$ on L and $\text{card}(L) = \infty$. Thus for all $n > m$, τ chosen in the algorithm for $\mathbf{M}'(T[n])$ would be σ.

It follows from the above case analysis that $\mathbf{M}'$ **TxtEx**-identifies each language **TxtEx**-identified by **M**. ∎

4 Conclusion

We solved an open problem of Kinber and Stephan in which they asked if each collection of languages identifiable by a strong monotonic learner can also be identified by a set-driven learner. We showed that if we require the set-driven learner to be total, then there are collections of languages that can be identified by total strong monotonic machines that cannot be identified by any set-driven machines. On the other hand, if the set-driven learner is allowed to diverge on certain evidential states, then the story is different. We showed that each collection of languages that can be identified by a strong monotonic machine can also be identified by a set-driven machine that is not required to be total. These two results uncover a finer gradation in the notion of set-drivenness that was not known earlier.

5 Acknowledgements

I would like to thank Arun Sharma, Efim Kinber and Frank Stephan for helpful comments and discussion.

References

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

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

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

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

[Ful90] M. Fulk. Prudence and other conditions on formal language learning. *Information and Computation*, 85:1–11, 1990.

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

[Jan90] K. P. Jantke. Monotonic and nonmonotonic inductive inference of functions and patterns. In *Nonmonotonic and Inductive Logic, 1st International Workshop, Karlsruhe, Germany*, pages 161–177. Springer Verlag, 1990. Lecture Notes in Computer Science 543.

[JS94] S. Jain and A. Sharma. On monotonic strategies for learning r.e. languages. In S.Arikawa and K. P. Jantke, editors, *Proceedings of the Fifth International Workshop on Algorithmic Learning Theory, Reinhardsbrunn Castle, Germany*, pages 349–364, October 1994.

[KS95] E. Kinber and F. Stephan. Language learning from texts: Mind changes, limited memory and monotonicity. Preliminary version to appear in Computational Learning Theory, 1995., 1995.

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

[OSW82] D. Osherson, M. Stob, and S. Weinstein. Learning strategies. *Information and Control*, 53:32–51, 1982.

[OSW86] 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.

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

[SR84] G. Schäfer-Richter. *Uber Eingabeabhangigkeit und Komplexitat von Inferenzstrategien.* PhD thesis, RWTH Aachen, 1984.