Journal of Computer and System Sciences **64**, 153–159 (2002)
doi:10.1006/jcss.2000.1702, available online at http://www.idealibrary.com on IDEAL®

Inductive Inference with Additional Information[1]

Mark Fulk

Received November 28, 1988; revised July 13, 1998

We consider the problem of inductively inferring a grammar for a language, given (positive) examples of the language and putative (possibly faulty) grammars for the complement of the language. The criterion of success is identification in the limit, defined by E. M. Gold (1967, *Inform. and Control* **10**, 447–474). Additional information is *useful* insofar as it allows the identification of language classes that would not be identified with positive examples alone. An infinite sequence of grammars past some finite position are correct for the complement of the input language, is not as useful a form of additional information as a single correct grammar for the complement. Grammars that are almost correct for the complement (that is, that make finitely many errors) are not as useful as correct grammars, and the usefulness of a grammar decreases with increasing numbers of errors.

1. INTRODUCTION

Consider the communications of a programmer and his or her employer. If the employer can only give examples of the behavior of a desired program, the programmer is in a situation very much like that of the inductive inference machines of Gold [Gol67]. However, this is not normally the case. Any employer is likely to provide further information about the problem, information which extends the provision of examples. We will consider situations in which the programmer is to produce a grammar for some language and the employer supplies examples of the language and information (perhaps a grammar) about the complement of the language.

Freivalds and Wiehagen [FW79] first considered this problem. Some of our definitions are variants of theirs; the motivation from the interaction of a programmer and his or her employer is also theirs. More recently, Jain and Sharma [JS91] have considered the case when the learning machine is given a grammar enumerating a large

[1] Mark Fulk unexpectedly passed away at the age of 45 in May 1997. Revisions to the original manuscript were carried out by Sanjay Jain and Arun Sharma. Correspondence should be addressed to Professor Arun Sharma, School of Computer Science and Engineering, The University of New South Wales, Sydney, Australia.

0022-0000/02 $35.00

enough subset of the language and/or its complement. Jain and Sharma [JS93] have also considered the case when the learner is given a bound on the minimal size grammar of the language.

2. MATHEMATICAL PRELIMINARIES

The recursion theoretic notions are from the books of Odifreddi [Odi89] and Soare [Soa87]. $N = \{0, 1, 2, ...\}$ is the set of all natural numbers, and this paper considers r.e. subsets L of N. $N^+ = \{1, 2, 3, ...\}$, the set of all positive integers. N_l denotes the set $\{x \mid x < l\}$. All conventions regarding range of variables apply, with or without decorations[2], unless otherwise specified. We let g, i, j, k, l, m, n, s, t, x, y, and z, range over N. $\varnothing$, $\in$, $\subseteq$, $\supseteq$, $\subset$, and $\supset$ denote empty set, member of, subset, superset, proper subset, and proper superset, respectively. max( ), min( ), and card( ) denote the maximum, minimum, and cardinality of a set, respectively, where by convention $\max(\varnothing) = 0$ and $\min(\varnothing) = \infty$. $\text{card}(S) \leqslant *$ means cardinality of set S is finite. a, b range over $N \cup \{*\}$. $\langle \cdot, \cdot \rangle$ stands for an arbitrary but fixed, one-to-one computable encoding of all pairs of natural numbers onto N. $\langle \cdot, \cdot, \cdot \rangle$ and similarly denotes a computable, 1–1 encoding of all triples of natural numbers onto N. $\bar{L}$ denotes the complement of set L. χ_L denotes the characteristic function of set L. $L_1 \Delta L_2$ denotes the symmetric difference of L_1 and L_2, i.e., $L_1 \Delta L_2 = (L_1 - L_2) \cup (L_2 - L_1)$. $L_1 =^a L_2$ means that $\text{card}(L_1 \Delta L_2) \leqslant a$. If $L_1 =^a L_2$, then we refer to L_1 as a-variant of L_2 and a grammar for L_1 as a grammar for a-variant of L_2. Quantifiers $\forall^\infty$, $\exists^\infty$, and $\exists!$ denote for all but finitely many, there exist infinitely many, and there exists a unique, respectively.

$\mathscr{R}$ denotes the set of total recursive functions from N to N. $\mathscr{R}_{0,1}$ denotes the set of 0–1 valued recursive functions. f, h range over total recursive functions. $\mathscr{C}$ ranges over subsets of $\mathscr{R}$. $\mathscr{E}$ denotes the set of all recursively enumerable sets. REC denotes the set of all recursive sets. L ranges over $\mathscr{E}$. $\mathscr{L}$, $\mathscr{S}$ range over subsets of $\mathscr{E}$. φ denotes a standard acceptable programming system (acceptable numbering). φ_i denotes the function computed by the ith program in the programming system φ. We also call i a program or index for φ_i. For a (partial) function η, domain(η) and range(η) respectively denote its domain and range. We often write $\eta(x)\downarrow$ ($\eta(x)\uparrow$) to denote that $\eta(x)$ is defined (undefined). W_i denotes the domain of φ_i. W_i is considered as the language enumerated by the ith program in φ system, and we say that i is a grammar or index for W_i. Φ denotes a standard Blum complexity measure [Blu67] for the programming system φ. $W_{i,s} = \{x < s \mid \Phi_i(x) < s\}$.

A *text* is a mapping from N to $N \cup \{\#\}$. Intuitively, #'s denote the pauses in the presentation of data. We let T range over texts. content(T) is defined to be set of natural numbers in the range of T (i.e., $\text{content}(T) = \text{range}(T) - \{\#\}$). T is a *text for* L iff $\text{content}(T) = L$. That means a text for L is an infinite sequence whose range, except for a possible #, is just L. Note that the only text for $\varnothing$ is an infinite sequence of #'s.

An *information sequence or informant* is a mapping from N to $(N \times N) \cup \{\#\}$. We let I range over informants. content(I) is defined to be the set of pairs in the

[2] Decorations are subscripts, superscripts, primes, etc.

range of I (i.e., content(I) = range$(I) - \{\#\}$). An *informant for* L is an infinite sequence I such that content$(I) = \{(x, b) \mid \chi_L(x) = b\}$. It is useful to consider a canonical information sequence for L. I is a canonical information sequence for L iff $I(x) = (x, \chi_L(x))$.

σ and τ range over finite initial segments of texts or information sequences, where the context determines which is meant. We denote the set of finite initial segments of texts by SEG and the set of finite initial segments of information sequences by SEQ. We use $\sigma \preccurlyeq T$ (respectively, $\sigma \preccurlyeq I$, $\sigma \preccurlyeq \tau$) to denote that σ is an initial segment of T (respectively, I, τ). If $\sigma \in$ SEQ is an initial segment of some canonical information sequence, then we refer to σ as canonical. $|\sigma|$ denotes the length of σ. $T[n]$ denotes the initial segment of T of length n. Similarly, $I[n]$ denotes the initial segment of I of length n.

A *learning machine* **M** is a mapping from initial segments of texts (information sequences) to N. We say that **M** converges on T to i (written: $\mathbf{M}(T)\downarrow = i$) iff, for all but finitely many n, $\mathbf{M}(T[n]) = i$. Convergence on information sequences is defined similarly.

DEFINITION 1 [Gol67]. **M TxtEx-***identifies* L (written $L \in \mathbf{TxtEx}(\mathbf{M})$), iff for all texts T for L, $\mathbf{M}(T)\downarrow$ and $W_{\mathbf{M}(T)} = L$.

M TxtEx-*identifies* $\mathscr{L}$ iff **M TxtEx**-identifies each $L \in \mathscr{L}$.

$\mathbf{TxtEx} = \{\mathscr{L} \mid (\exists \mathbf{M})[\mathbf{M}\ \mathbf{TxtEx}\text{-identifies } \mathscr{L}]\}$.

DEFINITION 2 [CL82]. **M TxtBc-***identifies* L (written $L \in \mathbf{TxtBc}(\mathbf{M})$), iff for all texts T for L, $(\forall^\infty n)[W_{\mathbf{M}(T[n])}] = L$

M TxtBc-*identifies* $\mathscr{L}$ iff **M TxtBc**-identifies each $L \in \mathscr{L}$.

$\mathbf{TxtBc}\{\mathscr{L} \mid (\exists \mathbf{M})[\mathbf{M}\ \mathbf{TxtBc}\text{-identities } \mathscr{L}]\}$.

DEFINITION 3 [Gol67]. **M InfEx-***identifies* L (written $L \in \mathbf{TxtEx}(\mathbf{M})$), iff for all information sequences I for L, $\mathbf{M}(I)\downarrow$ and $W_{\mathbf{M}(I)} = L$.

M InfEx-*identifies* $\mathscr{L}$ iff **M InfEx**-identifies each $L \in \mathscr{L}$.

$\mathbf{InfEx} = \{\mathscr{L} \mid (\exists \mathbf{M})[\mathbf{M}\ \mathbf{InfEx}\text{-identifies } \mathscr{L}]\}$.

DEFINITION 4 [CL82]. **M InfBc-***identifies* L (written: $L \in \mathbf{InfBc}(\mathbf{M})$), iff for all information sequences I for L, $(\forall^\infty n)[W_{\mathbf{M}(I[n])} = L]$.

M InfBc-*identifies* $\mathscr{L}$ iff **M InfBc**-identifies each $L \in \mathscr{L}$.

$\mathbf{InfBc} = \{\mathscr{L} \mid (\exists \mathbf{M})[\mathbf{M}\ \mathbf{InfBc}\text{-identifies } \mathscr{L}]\}$.

The restriction of **InfEx** to recursive languages is denoted $\mathbf{ExGen}_{0,1}$ in the literature. That is, $\mathbf{ExGen}_{0,1} = \mathbf{InfEx} \cap \{\mathscr{L} \mid \mathscr{L} \subseteq \mathrm{REC}\}$.

2.1. Learning with Additional Information

In the previous section, in learning from informants, we have already considered a situation in which negative information is available to the learner. In this section we consider the situation when the learner is given a sequence of grammars, all or all but finitely many of which approximately or perfectly generate the complement of the language.

For this, let G denote a mapping from N to N. Then we can consider G as a sequence of grammars and carry over the notation for texts to sequences of grammars (except that the grammar sequences do not contain #'s).

For learning with additional information, we consider a machine as a function from SEG × SEG to N. Intuitively, **M** gets both a text and a sequence of grammars.

We say that $\mathbf{M}(T, G)\downarrow = i$, iff for all but finitely many m, for all but finitely many n, $\mathbf{M}(T[m], G[n]) = i$. Otherwise, we say that $\mathbf{M}(T, G)$ *diverges*.

The following definition formalizes the situation in which an employer provides the machine, in addition to a text of the language L being learned, with a sequence of grammars that converges to a grammar for the complement of L.

DEFINITION 5. **M CTxtEx-***identifies* L, iff for all texts T for L, for all grammar sequences G such that $\lim_{i\to\infty} G(i)\downarrow$ to a grammar for $\bar{L}$, $\mathbf{M}(T, G)\downarrow$, and $W_{\mathbf{M}(T, G)} = L$.

Freivalds and Wiehagen [FW79] considered a variant of the above criterion in which the learning machine initially gets a grammar for the complement of L and then a text for L. It is easy to show that Freivalds and Wiehagen's variation is equivalent to the above. We used the above version in order to contrast it with Definition 7 below.

DEFINITION 6. **M C^{n}TxtEx-***identifies* L iff for all texts T for L, for all grammar sequences G such that $\lim_{i\to\infty} G(i)\downarrow$ to a grammar for n variant of $\bar{L}$, $\mathbf{M}(T, G)\downarrow$, and $W_{\mathbf{M}(T, G)} = L$.

One might work for an employer who frequently provides a new grammar, ostensibly for $\bar{L}$. For some time, the employer might provide incorrect grammars; after a while, however, he or she would start providing only correct grammars. Nonetheless, the employer would keep providing new grammars. Unfortunately, one has no way of knowing which grammars are correct for $\bar{L}$ and no way of knowing when two grammars are for the same set. The following definition models such a situation.

DEFINITION 7. **M CSTxtEx-***identifies* L iff for all texts T for L, for all grammar sequences G such that $(\forall^\infty i)[W_{G(i)} = \bar{L}]$, $\mathbf{M}(T, G)\downarrow$, and $W_{\mathbf{M}(T, G)} = L$.

One might want to consider another variant in which the employer, instead of providing a sequence of grammars, gives an algorithm to generate a sequence of grammars, all but finitely many of which are grammars for the complement. Using a trick similar to the one used in Theorem 2 below, it is easy to cancel out all grammars, in the sequence generated by the algorithm, which output an element of L. Thus, one can easily show that this variation leads to a class equivalent to **CtxtEx**.

3. RESULTS

THEOREM 1. **CSTxtEx** $\subset$ **CTxtEx**.

Proof. Suppose G is a sequence of grammars which converges to a grammar for $\bar{L}$. Then clearly, $(\forall^\infty n)[G(n)$ is a grammar for $\bar{L}]$. Thus **CSTxtEx** $\subseteq$ **CTxtEx**. We will now exhibit $\mathscr{L} \in$ **CTxtEx** $-$ **CSTxtEx**.

Let partner be a recursive function such that, for all x, partner$(2x) = 2x+1$ and partner$(2x+1) = 2x$. Let $\mathscr{L} = \{L \mid (\forall x)[x \in L \Leftrightarrow \text{partner}(x) \notin L]\}$. That is to say $L \in \mathscr{L}$ iff for every x, exactly one of x and partner(x) is in L.

Let h be a recursive function such that, for all i, $W_{h(i)} = \{x \mid \text{partner}(x) \in W_i\}$. Note that if $W_i \in \mathscr{L}$ then $W_{h(i)} = \bar{L}$.

CLAIM 1. $\mathscr{L} \in \mathbf{CTxtEx}$.

Proof. Define $\mathbf{M}$ as follows: for $m, n > 0$, $\mathbf{M}(T[m], G[n]) = h(G(n-1))$. Suppose $L \in \mathscr{L}$, T is a text for L, G is a sequence of grammars such that $\lim_{n\to\infty} G(n)\downarrow = i$, and $W_i = \bar{L}$. It is then easy to verify that $\mathbf{M}$ $\mathbf{CTxtEx}$-identifies L. ▮

CLAIM 2. $(\exists \mathscr{L}' \subseteq \mathscr{L})[\mathscr{L}' \in \mathbf{TxtBc} - \mathbf{TxtEx}]$.

Proof. For any $f \in \mathscr{R}$, let L_f^1 denote the language $\{\langle x, y\rangle \mid f(x) = y\}$ and L_f^2 denote the language $\{2\langle x, y\rangle \mid f(x) = y\} \cup \{2\langle x, y\rangle + 1 \mid f(x) \neq y\}$.

For any $\mathscr{C} \subseteq \mathscr{R}_{0,1}$, let $\mathscr{S}_{\mathscr{C}}^1 = \{L_f^1 \mid f \in \mathscr{C}\}$ and $\mathscr{S}_{\mathscr{C}}^2 = \{L_f^2 \mid f \in \mathscr{C}\}$.

One can easily verify that a grammar for L_f^1 (L_f^2) can be effectively converted into a grammar for L_f^2 (L_f^1). Moreover, a text for L_f^1 (L_f^2) can be effectively converted into a text for L_f^2 (L_f^1). Thus $\mathscr{S}_{\mathscr{C}}^1 \in \mathbf{TxtEx} \Leftrightarrow \mathscr{S}_{\mathscr{C}}^2 \in \mathbf{TxtEx}$ and $\mathscr{S}_{\mathscr{C}}^1 \in \mathbf{TxtBc} \Leftrightarrow \mathscr{S}_{\mathscr{C}}^2 \in \mathbf{TxtBc}$.

Essentially based on the techniques of [CS83] one can construct a class of functions $\mathscr{C}$ such that $\mathscr{S}_{\mathscr{C}}^1 \in \mathbf{TxtBc} - \mathbf{TxtEx}$. It follows that $\mathscr{S}_{\mathscr{C}}^2 \in \mathbf{TxtBc} - \mathbf{TxtEx}$. Taking $\mathscr{L}' = \mathscr{S}_{\mathscr{C}}^2$ proves the claim. ▮

CLAIM 3. *Let $\mathscr{L}'$ be as in Claim* 2. *Then* $\mathscr{L}' \notin \mathbf{CSTxtEx}$. *Thus* $\mathscr{L} \notin \mathbf{CSTxtEx}$.

Proof. Suppose by way of contradiction that $\mathscr{L}' \in \mathbf{CSTxtEx}$. We will then show that $\mathscr{L}' \in \mathbf{TxtEx}$, contradicting Claim 2. Let $\mathbf{M}'$ be a machine which $\mathbf{CSTxtEx}$-identifies $\mathscr{L}'$. Let $\mathbf{M}''$ be a machine which $\mathbf{TxtBc}$-identifies $\mathscr{L}'$. We define a machine $\mathbf{M}$ as follows. Suppose T is a text. Define $G(i) = h(\mathbf{M}''(T[i]))$. Define $\mathbf{M}(T[i]) = \mathbf{M}'(T[i], G[i])$.

Now suppose $L \in \mathscr{L}$, and T is a text for L. Now, since $\mathbf{M}''$ $\mathbf{TxtBc}$-identifies L, for all but finitely many i, $\mathbf{M}''(T[i])$ is a grammar for L. Thus, for all but finitely many i, $G(i)$ is a grammar for $\bar{L}$. Thus, since $\mathbf{M}'$ $\mathbf{CSTxtEx}$-identifies L, $\lim_{i\to\infty} \mathbf{M}(T[i]) = \lim_{i\to\infty} \mathbf{M}'(T[i], G[i])$ converges to a grammar for L. Thus, $\mathbf{M}$ $\mathbf{TxtEx}$-identifies L. Since L was an arbitrary member of $\mathscr{L}'$ it follows that $\mathscr{L}' \in \mathbf{TxtEx}$. A contradiction to Claim 2. Thus, $\mathscr{L}' \notin \mathbf{CSTxtEx}$.

The theorem follows from the above claims. ▮

The proof of the above theorem is modeled after a simplification of the proof of Theorem 5 in [FW79] (Corollary 1 below). However, the construction in that paper would not have sufficed for the above theorem.

COROLLARY 1 [FW79]. $\mathbf{ExGen}_{0,1} \subset \mathbf{CTxtEx}$.

The above corollary follows from Theorems 1 and 2.

THEOREM 2. $\mathbf{ExGen}_{0,1} \subseteq \mathbf{CSTxtEx}$.

Proof. Suppose $\mathbf{M}$ $\mathbf{ExGen}_{0,1}$-identifies $\mathscr{L}$. Suppose T is a text for L and G is grammar sequence such that $(\forall^{\infty} i)[W_{G(i)} = \bar{L}]$. Using such a T and G we first show how to construct an information sequence for L. This will then allow us to show that $\mathscr{L} \in \mathbf{CSTxtEx}$.

Fix a text T and a grammar sequence G. For any n we define Sq_n as follows (Sq_n depends on $T[n]$ and $G[n]$; we have omitted parameters T and G for ease of notation).

Let $X_n = \{G(i) \mid i < n \wedge W_{G(i),n} \cap \mathrm{content}(T[n]) = \varnothing\}$.

Intuitively, X_n is obtained by removing the "bad" grammars for $G[n]$.

Let $\mathrm{Comp}_n = \bigcup_{j \in X_n} W_{j,n}$.

Let l_n be the largest integer such that $N_{l_n} \subseteq \mathrm{content}(T[n]) \cup \mathrm{Comp}_n$.

Let Sq_n be a canonical information segment such that $\mathrm{Pos}(\mathrm{Sq}_n) = \mathrm{content}(T[n]) \cap N_{l_n}$ and $\mathrm{Neg}(\mathrm{Sq}_n) = \mathrm{Comp}_n \cap N_{l_n}$.

Suppose L is any recursive language. Suppose I is the canonical information sequence for L. Suppose T is a text for L and G is a sequence of grammars such that $(\forall^{\infty} i)[W_{G(i)} = \bar{L}]$. Then, for all but finitely many n, $\mathrm{Comp}_n \subseteq \overline{\mathrm{content}(T)}$. Moreover, for each $x \in \overline{\mathrm{content}(T)}$, for all but finitely many n, $x \in \mathrm{Comp}_n$. Thus, $(\forall^{\infty} n)[\mathrm{Sq}_n \preccurlyeq I]$ and $\lim_{n \to \infty} |\mathrm{Sq}_n| = \infty$.

Now define $\mathbf{M}'(T[n], G[m]) = \mathbf{M}(\mathrm{Sq}_{\min(\{m,n\})})$.

It is easy to verify that if $\mathbf{M}$ $\mathbf{ExGen}_{0,1}$-identifies L, then $\mathbf{M}'$ $\mathbf{CSTxtEX}$-identifies L. ∎

It is open at present whether $\mathbf{ExGen}_{0,1}$ is a proper subset of $\mathbf{CSTxtEX}$.

We now examine the effects of slightly erroneous grammar as additional information.

THEOREM 3. $(\forall n)[\mathbf{C}^n\mathbf{TxtEx} - \mathbf{C}^{n+1}\mathbf{TxtEx} \neq \varnothing]$.

Proof. Fix n. Let $L_n^i = N - \{x \mid (n+1) \cdot i \leqslant x < (n+1) \cdot (i+1)\}$. Note that L_n^i leaves out exactly $n+1$ elements of N.

Let $\mathscr{L}_n = \{N\} \cup \{L_n^i \mid i \in N\}$.

CLAIM 4. $\mathscr{L}_n \notin \mathbf{C}^{n+1}\mathbf{TxtEx}$.

Proof. Let $g_{\varnothing}$ be a grammar for $\varnothing$. Let $G(i) = g_{\varnothing}$. It is easy to verify that, for all $L \in \mathscr{L}_n$, $g_{\varnothing}$ is a grammar for $n+1$ variant of $\bar{L}$. Thus, $\mathscr{L}_n \in \mathbf{C}^{n+1}\mathbf{TxtEx}$ iff $\mathscr{L}_n \in \mathbf{TxtEx}$. Also, note that one can effectively convert a grammar for L_n^i to a grammar for L_0^i (and vice versa). Furthermore one can effectively convert a text for L_0^i to a text for L_n^i (and vice versa). Thus, $\mathscr{L}_n \in \mathbf{TxtEx}$ iff $\mathscr{L}_0 \in \mathbf{TxtEx}$. It was shown in [CL82] that $\mathscr{L}_0 \notin \mathbf{TxtEx}$. Thus, $\mathscr{L}_n \notin \mathbf{TxtEx}$ and $\mathscr{L}_n \notin \mathbf{C}^{n+1}\mathbf{TxtEx}$. ∎

CLAIM 5. $\mathscr{L}_n \in \mathbf{C}^n\mathbf{TxtEx}$.

Proof. Let g_N denote a grammar for N. Let g_i denote a grammar for L_n^i. We define $\mathbf{M}$ as follows:

$$\mathbf{M}(T[m], G[j]) = \begin{cases} g_N, & \text{if } W_{G(j),j} \subseteq \mathrm{content}(T[m]); \\ g_i & \text{otherwise, where } i = \left\lfloor \dfrac{\min(\overline{\mathrm{content}(T[M])})}{n+1} \right\rfloor. \end{cases}$$

It is easy to verify that $\mathbf{M}$ $\mathbf{C}^n\mathbf{TxtEx}$-identifies $\mathscr{L}$. ∎

Theorem follows from the above claims.

4. CONCLUSIONS

In this paper we considered the case when the learning machine is given additional information about the language L being learned in the form of sequence of grammars converging to a grammar for the complement of L. We extended this criteria allowing (a) the converged to grammar being for a variant of the complement of L, and (b) the sequence of grammars converging semantically rather than syntactically to grammar for the complement of L. We compared the above criteria with each other showing (a) a hierarchy based on the quality of grammar converged to (Theorem 3), and (b) semantic convergence (in additional information) is more restrictive than syntactic convergence (Theorem 1).

ACKNOWLEDGMENT

We thank John Case for suggesting the topic of this paper.

REFERENCES

[Blu67] M. Blum, A machine-independent theory of the complexity of recursive functions, *J. Assoc. Comput. Mach.* **14** (1967), 322–336.

[CL82] J. Case and C. Lynes, Machine inductive inference and language identification, *in* "Proceedings of the 9th International Colloquium on Automata, Languages and Programming" (M. Nielsen and E. M. Schmidt, Eds.), Lecture Notes in Computer Science, Vol. 140, pp. 107–115, Springer-Verlag, Berlin/New York, 1982.

[CS83] J. Case and C. Smith, Comparison of identification criteria for machine inductive inference, *Theoret. Comput. Sci.* **25** (1983), 193–220.

[FW79] R. Freivalds and R. Wiehagen, Inductive inference with additional information, *Electron. Inform. Kybernetik* **15** (1979), 179–195.

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

[JS91] S. Jain and A. Sharma, Learning in the presence of partial explanations, *Inform. and Comput.* **95** (1991), 162–191.

[JS93] S. Jain and A. Sharma, Learning with the knowledge of an upper bound on program size, *Inform. and Comput.* **102** (1993), 118–166.

[Odi89] P. Odifreddi, "Classical Recursion Theory," North-Holland, Amsterdam, 189.

[Soa87] R. Soare, "Recursively Enumerable Sets and Degrees," Springer-Verlag, Berlin/New York, 1987.