The Complexity of Learning SUBSEQ(A)

Stephen Fenner[1] and William Gasarch[2]

[1] University of South Carolina***
[2] University of Maryland at College Park†

Abstract. Higman showed that if A is *any* language then SUBSEQ(A) is regular, where SUBSEQ(A) is the language of all subsequences of strings in A. We consider the following inductive inference problem: given $A(\varepsilon), A(0), A(1), A(00), \ldots$ learn, in the limit, a DFA for SUBSEQ(A). We consider this model of learning and the variants of it that are usually studied in inductive inference: anomalies, mindchanges, and teams.

1 Introduction

In Inductive Inference [2, 4, 15] the basic model of learning is as follows.

Definition 1.1. A class $\mathcal{A}$ of decidable sets of strings[1] is in EX if there is a Turing machine M (the learner) such that if M is given $A(\varepsilon)$, $A(0)$, $A(1)$, $A(00)$, $A(01)$, $A(10)$, $A(11)$, $A(000)$, ..., where $A \in \mathcal{A}$, then M will output $e_1, e_2, e_3, \ldots$ such that $\lim_s e_s = e$ and e is an index for a Turing machine that decides A.

Note that the set A must be computable and the learner learns a Turing machine index for it. There are variants [1, 11, 13] where the set need not be computable and the learner learns something about the set (e.g., 'is it infinite?', or some other question).

Our work is based on the following remarkable theorem of Higman's [16].

Definition 1.2. Let $x, y \in \Sigma^*$. We say that x is a *subsequence* of y if $x = x_1 \cdots x_n$ and $y \in \Sigma^* x_1 \Sigma^* x_2 \cdots x_{n-1} \Sigma^* x_n \Sigma^*$. We denote this by $x \preceq y$.

Notation 1.3. If A is a set of strings, then SUBSEQ(A) is the set of subsequences of strings in A.

Theorem 1.4 (Higman [16]). *If A is any language over Σ^*, then* SUBSEQ(A) *is regular. In fact, for any language A there is a unique minimum finite set S of strings such that*

$$\text{SUBSEQ}(A) = \{x \in \Sigma^* : (\forall z \in S)[z \not\preceq x]\}. \tag{1}$$

*** Department of Computer Science and Engineering, Columbia, SC 29208. fenner@cse.sc.edu. Partially supported by NSF grant CCF-05-15269.

† Department of Computer Science and Institute for Advanced Computer Studies, College Park, MD 20742. gasarch@cs.umd.edu. Partially supported by NSF grant CCR-01-05413.

[1] The basic model is usually described in terms of learning computable functions; however, virtually all of the results hold in the setting of decidable sets.

Note that A is *any language whatsoever*. Hence we can investigate the following learning problem.

Notation 1.5. We let $s_1, s_2, s_3, \ldots$ be the standard length-first lexicographic enumeration of Σ^*.

Definition 1.6. A class $\mathcal{A}$ of sets of strings in Σ^* is in SUBSEQ-EX if there is a Turing machine M (the learner) such that if M is given $A(s_1), A(s_2), A(s_3), \ldots$ where $A \in \mathcal{A}$, then M will output $e_1, e_2, e_3, \ldots$ such that $\lim_s e_s = e$ and e is an index for a DFA that recognizes SUBSEQ(A). It is easy to see that we can take e to be the least index of the minimum state DFA that recognizes SUBSEQ(A). Formally, we will refer to $A(s_1)A(s_2)A(s_3)\cdots$ being on an auxilary tape.

This problem is part of a general theme of research: given a language A, rather than try to learn the language (which may be undecidable) learn some aspect of it. In this case we learn SUBSEQ(A). Note that we learn SUBSEQ(A) in a very strong way in that we have a DFA for it.

We look at anomalies, mind-changes, and teams (standard Inductive Inference variants) in this context. We prove the following results.

1. $\mathcal{A} \in$ SUBSEQ-EX^a means that the final DFA may be wrong on $\leq a$ strings. $\mathcal{A} \in$ SUBSEQ-EX* mean that the final DFA may be wrong on a finite number of strings. The anomaly hierarchy collapses: that is SUBSEQ-EX = SUBSEQ-EX*. This constrasts sharply with the case of EX^a.
2. Let $\mathcal{A} \in$ SUBSEQ-EX$_n$ mean that the Turing machine makes at most $n+1$ conjectures (and hence changes its mind at most n times). The mind-change hiearchy separates: that is, for all n, SUBSEQ-EX$_n \subset$ SUBSEQ-EX$_{n+1}$.
3. The mind-change hierarchy also separates if you allow a transfinite number of mind-changes, up to ω_1^{CK}.
4. Let $\mathcal{A} \in [a,b]$SUBSEQ-EX mean that there are b Turing machines trying to learn the DFA, and we demand that at least a of them succeed (it may be a different a machines for different $A \in \mathcal{A}$).
   (a) If $1 \leq a \leq b$ and $q = \lfloor b/a \rfloor$, then $[a,b]$SUBSEQ-EX = $[1,q]$SUBSEQ-EX. Hence we need only look at team learning of the form $[1,n]$SUBSEQ-EX.
   (b) The team hiearchy separates. That is, for all b, $[1,b]$SUBSEQ-EX $\subset$ $[1,b+1]$SUBSEQ-EX.

Note 1.7. PEX [4, 3] is like EX except that the conjectures must be for total Turing machines. The class SUBSEQ-EX is similar in that all the machines are total (in fact, DFAs) but different in that we learn the subsequence language, and the input need not be computable. The anomaly hierarchy for SUBSEQ-EX collapses just as it does for PEX; however the team hiearchy for SUBSEQ-EX is proper, unlike for PEX.

2 Definitions

2.1 Definitions about Subsequences

We let $\mathbb{N} = \{0, 1, 2, \ldots\}$.

Notation 2.1. Given a language A, we call the unique set S satisfying (1) the *obstruction set* of A and denote it by $os(A)$. In this case, we also say that S *obstructs* A.

The following facts are obvious:

- The $\preceq$ relation is computable.
- For every string x there are finitely many $y \preceq x$, and given x one can compute a canonical index for the set of all such y.
- By various facts from automata theory, including the Myhill-Nerode minimization theorem: given a DFA, NFA, or regular expression for a language A, one can effectively compute the unique minimum state DFA recognizing A. (The minimum state DFA is given in some canonical form.)
- Given DFAs F and G, one can effectively compute DFAs for $\overline{L(F)}$, $L(F) \cup L(G)$, $L(F) \cap L(G)$, $L(F) - L(G)$, and $L(F) \triangle L(G)$ (symmetric difference). One can also effectively determine whether or not $L(F) = \emptyset$ and whether or not $L(F)$ is finite.
- For any language A, the set SUBSEQ(A) is completely determined by $os(A)$, and in fact, $os(A) = os(\text{SUBSEQ}(A))$.
- The strings in the obstruction set of a language must be pairwise $\preceq$-incomparable (i.e., the obstruction set is an $\preceq$-antichain). Conversely, any $\preceq$-antichain S obstructs some language. For any $S \subseteq \Sigma^*$ define

$$\text{obsby}(S) = \{x \in \Sigma^* : (\forall z \in S)[z \not\preceq x]\}.$$

  The term obsby(S) is an abbreviation for 'obstructed by S'. Note that $os(\text{obsby}(S)) \subseteq S$, and equality holds iff S is an $\preceq$-antichain.

Definition 2.2. We say that a language $A \subseteq \Sigma^*$ is $\preceq$*-closed* if SUBSEQ(A) = A.

Observation 2.3. A language A is $\preceq$-closed if and only if there exists a language B such that $A = \text{SUBSEQ}(B)$.

Observation 2.4. Any infinite $\preceq$-closed set contains strings of every length.

The next proposition implies that finding $os(A)$ is computationally equivalent to finding a DFA for SUBSEQ(A). We omit the easy proof.

Proposition 2.5. *The following tasks are computable:*

1. *Given a DFA F, find a DFA G such that $L(G) = \text{SUBSEQ}(L(F))$.*
2. *Given the canonical index of a finite language $D \subseteq \Sigma^*$, compute a regular expression for (and hence the minimum state DFA recognizing) the language* $\text{obsby}(D) = \{x \in \Sigma^* : (\forall z \in D)[z \not\preceq x]\}$.
3. *Given a DFA F, decide whether or not $L(F)$ is $\preceq$-closed.*
4. *Given a DFA F, compute the canonical index of $os(L(F))$.*

2.2 Classes of Languages

We define classes of languages via the types of machines that recognize them.

Notation 2.6.

1. $D_1, D_2, \ldots$ is a standard enumeration of finite languages. (e is the *canonical index* of D_e.)
2. $F_1, F_2, \ldots$ is a standard enumeration of minimized DFAs, presented in some canonical form, so that for all i and j, if $L(F_i) = L(F_j)$ then $F_i = F_j$. (We might have $i \neq j$ and $F_i = F_j$, however.) Let REG $= \{L(F_1), L(F_2), \ldots\}$.
3. $P_1, P_2, \ldots$ is a standard enumeration of $\{0,1\}$-valued polynomial-time Turing machines. Let P $= \{L(P_1), L(P_2), \ldots\}$.
4. $M_1, M_2, \ldots$ is a standard enumeration of Turing machines. We let CE $= \{L(M_1), L(M_2), \ldots\}$, where $L(M_i)$ is the set of all x such that $M_i(x)$ halts with output 1.
5. We let DEC $= \{L(N) : N$ is a total Turing machine$\}$.

For the notation that relates to computability theory, our reference is [20]. For separation results, we will often construct tally sets, i.e., subsets of 0^*.

Notation 2.7.

1. The empty string is denoted by ε.
2. For $m \in \mathbb{N}$, we define $J_m = \{0^i : i < m\}$.
3. If $A \subseteq 0^*$ is finite, we let $m(A)$ denote the least m such that $A \subseteq J_m$, and we observe that SUBSEQ$(A) = J_{m(A)}$.

If $A, B \subseteq 0^*$ and A is finite, we define a "shifted join" of A and B as follows:

$$\begin{aligned} A \cup\!\!+\, B &= \{xx0 : x \in A\} \cup \{0^{2m(A)}xx : x \in B\} \\ &= \{0^{2n+1} : 0^n \in A\} \cup \{0^{2(m(A)+n)} : 0^n \in B\}. \end{aligned}$$

In $A \cup\!\!+\, B$, all the elements from A have odd length and are shorter than the elements from B, which have even length. We define inverses to the $\cup\!\!+$ operator: For every $m \geq 0$ and language A, let

$$\begin{aligned} \xi(A) &:= \{0^n : n \geq 0 \wedge 0^{2n+1} \in A\}, \\ \pi(m; A) &:= \{0^n : n \geq 0 \wedge 0^{2(m+n)} \in A\}. \end{aligned}$$

If $A, B \subseteq 0^*$ and A is finite, then $A = \xi(A \cup\!\!+\, B)$ and $B = \pi(m(A); A \cup\!\!+\, B)$.

2.3 Variants on SUBSEQ-EX

There are several variations on the definition of SUBSEQ-EX.

Definition 2.8. Let $e_1, e_2, \ldots$ be the output sequence of some learner M on some language A. Let $n > 1$. We say that M *changes its mind at time* n if $e_n \neq e_{n-1}$. For fixed $n \geq 0$, let SUBSEQ-EX$_n$ be the same as SUBSEQ-EX except that we restrict the learner to change its mind no more than n times, for any language $A \in \mathcal{C}$.

Obviously,

$$\text{SUBSEQ-EX}_0 \subseteq \text{SUBSEQ-EX}_1 \subseteq \text{SUBSEQ-EX}_2 \subseteq \cdots \subseteq \text{SUBSEQ-EX}. \quad (2)$$

We will extend this definition into the transfinite later.

Definition 2.9. Let $a \in \mathbb{N}$, let M be a TM, and let $A \subseteq \Sigma^*$ be any language. The machine M *learns* SUBSEQ(A) *with at most a anomalies* (respectively, with *finitely many anomalies*) if it behaves as follows: when you feed $A(s_1)$, $A(s_2)$, $A(s_3)$, ... and M outputs $e_1, e_2, e_3, \ldots$, then $e = \lim_{n\to\infty} e_n$ exists, and $|L(F_e) \triangle \text{SUBSEQ}(A)| \leq a$ (respectively, $|L(F_e) \triangle \text{SUBSEQ}(A)|$ is finite). For $a \in \mathbb{N}$ let SUBSEQ-EX^a be the same as SUBSEQ-EX except that we allow the learner to learn SUBSEQ(A) with at most a anomalies. Let SUBSEQ-EX* be the same as SUBSEQ-EX except that we allow the learner to learn SUBSEQ(A) with finitely many anomalies. Note that in the latter case, the number of anomalies may vary with the language being learned.

Clearly,

$$\text{SUBSEQ-EX} = \text{SUBSEQ-EX}^0 \subseteq \text{SUBSEQ-EX}^1 \subseteq \cdots \subseteq \text{SUBSEQ-EX}^*. \quad (3)$$

Definition 2.10. For integers $1 \leq a \leq b$, we say that a class $\mathcal{C}$ of languages is in $[a,b]$SUBSEQ-EX iff there are learners $M_1, \ldots, M_b$ such that for any $A \in \mathcal{C}$, at least a of the learners learn SUBSEQ(A).

Evidently, if $a \geq c$ and $b \leq d$, then $[a,b]$SUBSEQ-EX $\subseteq [c,d]$SUBSEQ-EX.

Definition 2.11. If $X \subseteq \mathbb{N}$, then SUBSEQ-EX^X is the same as SUBSEQ-EX except that we allow the learner to be an oracle Turing machine using oracle X.

We may combine these variants in a large variety of ways.

3 Main Results

3.1 Standard Learning

It was essentially shown in [6] that DEC $\notin$ SUBSEQ-EX. The proof there can be tweaked to show the stronger result that P $\notin$ SUBSEQ-EX. We omit the proof; however, it will appear in the full version.

Theorem 3.1 ([6]). *There is a computable function g such that for all e, setting $A = L(P_{g(e)})$, we have $A \subseteq 0^*$ and* SUBSEQ(A) *is not learned by M_e.*

Corollary 3.2. P $\notin$ SUBSEQ-EX. *In fact,* P $\cap\, \mathcal{P}(0^*) \notin$ SUBSEQ-EX.

We now show some classes that are in SUBSEQ-EX.

Definition 3.3. Let $\mathcal{F} = \{A \subseteq \Sigma^* : A \text{ is finite}\}$.

Proposition 3.4. $\mathcal{F} \in$ SUBSEQ-EX.

Proof. Let M be a learner that, when $A \in \mathcal{F}$ is on the tape, outputs $k_1, k_2, \ldots$, where each k_i is the least index of a DFA recognizing SUBSEQ$(A \cap \Sigma^{\leq i})$. Clearly, M learns SUBSEQ(A). □

More generally, we have

Proposition 3.5. REG $\in$ SUBSEQ-EX.

Proof. When A is on the tape, $n = 0, 1, 2, \ldots$, the learner M

1. finds the least k such that $A \cap \Sigma^{<n} = L(F_k) \cap \Sigma^{<n}$, then
2. outputs the least ℓ such that $L(F_\ell) =$ SUBSEQ$(L(F_k))$.

If A is regular, then clearly M will converge to the least k such that $A = L(F_k)$, whence M will converge to the least ℓ such that $L(F_\ell) =$ SUBSEQ(A). □

3.2 Anomalies

The next theorem shows that the hierarchy of (3) collapses completely.

Theorem 3.6. SUBSEQ-EX = SUBSEQ-EX*. *In fact, there is a computable h such that for all e and languages A, if M_e learns* SUBSEQ(A) *with finitely many anomalies, then $M_{h(e)}$ learns* SUBSEQ(A) *(with zero anomalies).*

Proof. Given e, let $M = M_{h(e)}$ be the following learner:
When a language A is on the tape:

1. Run M_e with A. Wait for M_e to output something.
2. Whenever M_e outputs some index k do the following:
   (a) Let n be the number of outputs of M_e thus far.
   (b) Build a finite set E of anomalies as follows:
      i. Initially, $E := \emptyset$.
      ii. For each $w \in \Sigma^{<n}$, define $S(w) = \{z \in \Sigma^* : w \preceq z\}$.
         – If F_k rejects w but $S(w) \cap \Sigma^{\leq n} \cap A \neq \emptyset$, then put w into E. (w is a "false negative.")
         – If F_k accepts w but $S(w) \cap \Sigma^{\leq n} \cap A = \emptyset$ and $S(w) \cap \Sigma^{=n} \cap L(F_k) = \emptyset$, then put w into E. (w is a "potentially false positive.")
   (c) Output the least index for a DFA G such that $L(G) = L(F_k) \triangle E$.

If M_e learns SUBSEQ(A) with finite anomalies, then there is a DFA F such that for all large enough n the nth output of M_e is an index for F, and moreover, SUBSEQ$(A) \triangle L(F) \subseteq \Sigma^{<n}$ (all anomalies are of length less than n). We claim that for all large enough n the anomaly set E built in step 2b is exactly SUBSEQ$(A) \triangle L(F)$, and hence SUBSEQ$(A) = L(G)$, where G is output by M in step 2c. The theorem follows once we prove the claim.

Let n be large enough as above, and let w be any string in $\Sigma^{<n}$. There are four cases to consider:

$\boldsymbol{w \notin \text{SUBSEQ}(A) \cup L(F)}$**.** Then $F(w)$ rejects and $S(w) \cap A = \emptyset$, so we don't put w into E. (w is a "true negative.")

$\boldsymbol{w \in \text{SUBSEQ}(A) - L(F)}$**.** Then $F(w)$ rejects, but there is some $z \in S(w) \cap A$. So as long as $n \geq |z|$, i.e., $z \in \Sigma^{\leq n}$, we will put w into E.

$\boldsymbol{w \in L(F) - \text{SUBSEQ}(A)}$**.** Then $F(w)$ accepts, and $S(w) \cap A = \emptyset$. Furthermore, $S(w) \cap \Sigma^{=n} \cap L(F) = \emptyset$ because there are no anomalies of length n. Thus we put w into E.

$\boldsymbol{w \in \text{SUBSEQ}(A) \cap L(F)}$**.** Then $F(w)$ accepts. Since $w \in \text{SUBSEQ}(A)$, there is a $z \in S(w) \cap A$. If $|z| \leq n$, then $S(w) \cap \Sigma^{\leq n} \cap A \neq \emptyset$, and we would not put w into E. If $|z| > n$, then there is some $y \in \Sigma^{=n}$ with $w \preceq y \preceq z$. Thus we have $y \in \text{SUBSEQ}(A)$, and since there are no anomalies of length n, we also have $y \in L(F)$. Therefore, $y \in S(w) \cap \Sigma^{=n} \cap L(F) \neq \emptyset$, and so we don't put w into E.

Thus $E = (\text{SUBSEQ}(A) \triangle L(F)) \cap \Sigma^{<n} = \text{SUBSEQ}(A) \triangle L(F)$ for all large enough n. The claim follows. □

3.3 Mind Changes

The next theorems show that the hierarchy (2) separates.

Definition 3.7. For every $i > 0$, define the class

$$\mathcal{C}_i = \{A \subseteq \Sigma^* : |A| \leq i\}.$$

Theorem 3.8. *$\mathcal{C}_i \in \text{SUBSEQ-EX}_i$ for all $i \in \mathbb{N}$. In fact, there is a single learner M that for each i learns $\text{SUBSEQ}(A)$ for every $A \in \mathcal{C}_i$ with at most i mind-changes.*

Proof. Let M be as in the proof of Proposition 3.4. Clearly, M learns any $A \in \mathcal{C}_i$ with at most $|A|$ mind-changes. □

Theorem 3.9. *For each $i > 0$, $\mathcal{C}_i \cap \mathcal{P}(0^*) \notin \text{SUBSEQ-EX}_{i-1}$. In fact, there is a computable function ℓ such that, for each e and $i > 0$, $M_{\ell(e,i)}$ is total and decides a unary language $A_{e,i} = L(M_{\ell(e,i)}) \subseteq 0^*$ such that $|A_{e,i}| \leq i$ and M_e does not learn $\text{SUBSEQ}(A_{e,i})$ with fewer than i mind-changes.*

Proof. Given e and $i > 0$ we use the Recursion Theorem with Parameters to construct a machine $N = M_{\ell(e,i)}$ that implements the following recursive algorithm to compute $A_{e,i}$:

Given input x,

1. If $x \notin 0^*$, then reject. (This ensures that $A_{e,i} \subseteq 0^*$.) Otherwise, let $x = 0^n$.
2. Recursively compute $R_n = A_{e,i} \cap J_n$.
3. Simulate M_e for $n - 1$ steps with R_n on the tape. (Note that M_e does not have time to read any of the tape corresponding to inputs $0^{n'}$ for $n' \geq n$.) If M_e does not output anything within this time, then reject.

4. Let k be the most recent output of M_e in the previous step, and let c be the number of mind-changes that M_e has made up to this point. If $c < i$ and $L(F_k) = \text{SUBSEQ}(R_n)$, then accept; else reject.

In step 3 of the algorithm, M_e behaves the same with R_n on its tape as it would with $A_{e,i}$ on its tape, given the limit on its running time.

Let $A_{e,i} = \{0^{z_0}, 0^{z_1}, \ldots\}$, where $z_0 < z_1 < \cdots$ are natural numbers.

Claim 3.10. For $0 \leq j$, if z_j exists, then M_e (with $A_{e,i}$ on its tape) must output a DFA for $\text{SUBSEQ}(R_{z_j})$ within $z_j - 1$ steps, having changed its mind at least j times when this occurs.

Proof (of the claim). We proceed by induction on j: For $j = 0$, the string 0^{z_0} is accepted by N only if within $z_0 - 1$ steps M_e outputs a k where $L(F_k) = \emptyset = \text{SUBSEQ}(R_{z_0})$; no mind-changes are required. Now assume that $j \geq 0$ and z_{j+1} exists, and also (for the inductive hypothesis) that within $z_j - 1$ steps M_e outputs a DFA for $\text{SUBSEQ}(R_{z_j})$ after at least j mind-changes. We have $R_{z_j} \subseteq J_{z_j}$ but $0^{z_j} \in R_{z_{j+1}}$, and so $\text{SUBSEQ}(R_{z_j}) \neq \text{SUBSEQ}(R_{z_{j+1}})$. Since N accepts $0^{z_{j+1}}$, it must be because M_e has just output a DFA for $\text{SUBSEQ}(R_{z_{j+1}})$ within $z_{j+1} - 1$ steps, thus having changed its mind at least once since the z_jth step of its computation, making at least $j + 1$ mind-changes in all. So the claim holds for $j + 1$. This ends the proof of the claim. □

First we show that $A_{e,i} \in \mathcal{C}_i$. Indeed, by Claim 3.10, z_i cannot exist, because the algorithm would explicitly reject such a string 0^{z_i} if M_e made at least i mind-changes in the first $z_i - 1$ steps. Thus we have $|A_{e,i}| \leq i$, and so $A_{e,i} \in \mathcal{C}_i$.

Next we show that M_e cannot learn $A_{e,i}$ with fewer than i mind-changes. Suppose that with $A_{e,i}$ on its tape, M_e makes fewer than i mind-changes. Suppose also that there is a DFA F such that cofinitely many of M_e's outputs are indices for F. Let t be least such that $t \geq m(A_{e,i})$ and M_e outputs an index for F within $t - 1$ steps. Then $L(F) \neq \text{SUBSEQ}(A_{e,i})$, for otherwise the algorithm would accept 0^t and so $0^t \in A_{e,i}$, contradicting the choice of t. It follows that M_e cannot learn $A_{e,i}$ with fewer than i mind-changes. □

Transfinite Mind Changes and Procrastination. We extend the results of this section into the transfinite. Freivalds & Smith defined EX_α for all constructive ordinals α [8]. When $\alpha < \omega$, the definition is the same as the finite mind-change case above. If $\alpha \geq \omega$, then the learner may revise its bound on the number of mind changes during the computation. The learner may be able to revise more than once, or even compute a bound on the number of future revisions, and this bound itself could be revised, etc., depending on the size of α. We define SUBSEQ-EX_α for all constructive α, then show that this transfinite hierarchy separates. Our definition is slightly different from, but equivalent to, the definition in [8]. For general background on constructive ordinals, see [18, 19].

Definition 3.11. A *procrastinating learner* is a learner M equipped with an additional *ordinal tape*, whose contents is always a constructive ordinal. Given a language on its input tape, M runs forever, producing infinitely many outputs as usual, except that just before M changes its mind, if α is currently on its ordinal tape, M is required to compute some ordinal $\beta < \alpha$ and replace the contents of the ordinal tape with β before proceeding to change its mind. (So if $\alpha = 0$, no mind-change may take place.) M may alter its ordinal tape at any other time, but the only allowed change is replacement with a lesser ordinal.

Thus a procrastinating learner must decrease its ordinal tape before each mind-change. We abuse notation and let $M_1, M_2, \ldots$ be a standard enumeration of procrastinating learners. Such an effective enumeration can be shown to exist.

Definition 3.12. Let M be a procrastinating learner, α a constructive ordinal, and A a language. We say that M *learns* SUBSEQ(A) *with* α *mind-changes* if M learns SUBSEQ(A) with α initially on its ordinal tape.

If $\mathcal{C}$ is a class of languages, we say that $\mathcal{C} \in$ SUBSEQ-EX$_\alpha$ if there is a procrastinating learner that learns every language in $\mathcal{C}$ with α mind-changes.

The following is straightfoward and given without proof.

Proposition 3.13. *If* $\alpha < \omega$*, then* SUBSEQ-EX$_\alpha$ *is equal to the corresponding class in Definition 2.8.*

Proposition 3.14. *For all* $\alpha < \beta < \omega_1^{\mathrm{CK}}$*,*

$$\text{SUBSEQ-EX}_\alpha \subseteq \text{SUBSEQ-EX}_\beta \subseteq \text{SUBSEQ-EX}.$$

Proof. The first containment follows from the fact that any procrastinating learner allowed α mind-changes can be simulated by a procrastinating learner, allowed β mind-changes, that first decreases its ordinal tape from β to α before the simulation. (α is hard-coded into the simulator.)

The second containment is trivial; any procrastinating learner is also a regular learner. □

In [8], Freivalds and Smith and showed that the hierarchy separates using classes of languages constructed by diagonalization. We take a different approach and define "natural" (using the term loosely) classes of languages that separate the SUBSEQ-EX$_\alpha$ hierarchy.

Definition 3.15. For every $\alpha < \omega_1^{\mathrm{CK}}$, we define the class $\mathcal{F}_\alpha$ inductively as follows: Let n and λ uniquely satisfy $n < \omega$, λ is not a successor, and $\lambda + n = \alpha$.

- If $\lambda = 0$, let

$$\mathcal{F}_\alpha = \mathcal{F}_n = \{A \cup\!\!+ \emptyset : (A \subseteq 0^*) \wedge (|A| \leq n)\}.$$

- If $\lambda > 0$, then $\lambda = 3 \cdot 5^e$ for some e. Let

$$\mathcal{F}_\alpha = \{A \cup\!\!+ B : (A, B \subseteq 0^*) \wedge (|A| \leq n+1) \wedge (B \in \mathcal{F}_{M_e(m(A))})\}.$$

It is evident by induction on α that $\mathcal{F}_\alpha$ consists only of finite unary languages, and that $\emptyset \in \mathcal{F}_\alpha$. Note that in the case of finite α we have the condition that $|A| \leq n$, but in the case of $\alpha \geq \omega$ we have the condition that $|A| \leq n+1$. This is not a mistake.

The next two theorems have proofs that are similar to the finite mind-change case in some ways, but very different in others. Unfortunatly we have to omit the proofs for this version.

Theorem 3.16. *For every constructive α, $\mathcal{F}_\alpha \in \text{SUBSEQ-EX}_\alpha$. In fact, there is a single procrastinating learner N such that for every α, N learns every language in $\mathcal{F}_\alpha$ with α mind-changes.*

Theorem 3.17. *For all $\beta < \alpha < \omega_1^{\text{CK}}$, $\mathcal{F}_\alpha \notin \text{SUBSEQ-EX}_\beta$. In fact, there is a computable function r such that, for each e and $\beta < \alpha < \omega_1^{\text{CK}}$, $M_{r(e,\alpha,\beta)}$ is total and decides a language $A_{e,\alpha,\beta} = L(M_{r(e,\alpha,\beta)}) \in \mathcal{F}_\alpha$ such that M_e does not learn $\text{SUBSEQ}(A_{e,\alpha,\beta})$ with β mind-changes.*

We end with an easy observation.

Corollary 3.18.

$$\text{SUBSEQ-EX} \not\subseteq \bigcup_{\alpha<\omega_1^{\text{CK}}} \text{SUBSEQ-EX}_\alpha.$$

Proof. Let $\mathcal{F} \in \text{SUBSEQ-EX}$ be the class of Definition 3.3. For all $\alpha < \omega_1^{\text{CK}}$, we clearly have $\mathcal{F}_{\alpha+1} \subseteq \mathcal{F}$, and so $\mathcal{F} \notin \text{SUBSEQ-EX}_\alpha$ by Theorem 3.17. □

3.4 Teams

In this section, we show that $[a,b]$SUBSEQ-EX depends only on $\lfloor b/a \rfloor$. Recall that $b \leq c$ implies $[a,b]\text{SUBSEQ-EX} \subseteq [a,c]\text{SUBSEQ-EX}$.

Lemma 3.19. *For all $1 \leq a \leq b$, $[a,b]\text{SUBSEQ-EX} = [1,\lfloor b/a \rfloor]\text{SUBSEQ-EX}$.*

Proof. Let $q = \lfloor b/a \rfloor$. To show that $[1,q]\text{SUBSEQ-EX} \subseteq [a,b]\text{SUBSEQ-EX}$, let $\mathcal{C} \in [1,q]\text{SUBSEQ-EX}$. Then there are learners $Q_1, \ldots, Q_q$ such that for all $A \in \mathcal{C}$ there is some Q_i that learns $\text{SUBSEQ}(A)$. For all $1 \leq i \leq q$ and $1 \leq j \leq a$, let $N_{i,j} = Q_i$. Then clearly, $\mathcal{C} \in [a,qa]\text{SUBSEQ-EX}$ as witnessed by the $N_{i,j}$. Thus, $\mathcal{C} \in [a,b]\text{SUBSEQ-EX}$, since $b \geq qa$.

To show the reverse containment, suppose that $\mathcal{D} \in [a,b]\text{SUBSEQ-EX}$. Let $Q_1, \ldots, Q_b$ be learners such that for each $A \in \mathcal{D}$, at least a of the Q_i's learn $\text{SUBSEQ}(A)$. We define learners $N_1, \ldots, N_q$ to behave as follows.

Each N_j runs all of $Q_1, \ldots, Q_b$. At any time t, let $k_1(t), \ldots, k_b(t)$ be the most recent outputs of $Q_1, \ldots, Q_b$, respectively, after running for t steps (if some machine Q_i has not yet output anything in t steps, let $k_i(t) = 0$).

Define a *consensus value at time t* to be a value that shows up at least a times in the list $k_1(t), \ldots, k_b(t)$. There can be at most q many different consensus values at any given time. The idea is that the machines N_j output consensus values. If

k_{correct} is the least index of a DFA recognizing SUBSEQ(A), then k_{correct} will be a consensus value at all sufficiently large times t, and so we hope that k_{correct} will eventually always be output by some N_j. We could simply assign each consensus value at time t to be output by one of the machines $N_1, \ldots, N_q$ to guarantee that k_{correct} is eventually always output by one or another of the N_j, but this does not suffice, because it may be output by different N_j at different times. The tricky part is to ensure that k_{correct} is eventually output not only by some N_j, but also by the *same* N_j each time. To make sure of this, we hold a popularity contest among the consensus! values.

For $1 \leq j \leq q$ and $t = 1, 2, 3, \ldots$, each machine N_j computes $k_1(t'), \ldots, k_b(t')$ and all the consensus values at time t' for all $t' \leq t$. For each $v \in \mathbb{N}$, let $p_v(t)$ be the number of times $\leq t$ at which v is a consensus value. We call $p_v(t)$ the *popularity* of v at time t. We rank all the consensus values found so far (at all times $t' \leq t$) in order of decreasing popularity; if there is a tie, i.e., some $u \neq v$ such that $p_u(t) = p_v(t)$, then we consider the smaller value to be more popular. As its t'th output, N_j outputs the j'th most popular consensus value at time t.

This ends the description of the machines $N_1, \ldots, N_q$.

We'll be done if we can show that there is a $1 \leq j \leq q$ such that N_j outputs k_{correct} cofinitely often.

Let t_0 be least such that k_{correct} is a consensus value at time t for all $t \geq t_0$. We claim that

- from t_0 on, k_{correct} will never lose ground in the popularity rankings, and
- eventually k_{correct} will be one of the q most popular consensus values.

For all $t \geq t_0$, let $P(t)$ be the set of all values that are at least as popular as k_{correct} at time t. That is,

$$P(t) = \{v \in \mathbb{N} : \text{either } p_v(t) > p_{k_{\text{correct}}}(t), \text{ or } p_v(t) = p_{k_{\text{correct}}}(t) \text{ and } v \leq k_{\text{correct}}\}.$$

We claim that

$$P(t_0) \supseteq P(t_0 + 1) \supseteq P(t_0 + 2) \supseteq \cdots.$$

This holds because the only way for a value v to go from being less popular than k_{correct} to being more popular than k_{correct} is for there to be a time $t \geq t_0$ when v is a consensus value but k_{correct} is not, but this never happens.

Since the $P(t)$ are clearly all finite, there is a $t_1 \geq t_0$ such that $P(t_1) = P(t_1 + 1) = P(t_1 + 2) = \cdots = \bigcap_t P(t)$. Set $P = P(t_1)$, and let $r = |P|$. It suffices to show that $r \leq q$, for then N_r outputs k_{correct} for all $t \geq t_1$ and so N_r learns SUBSEQ(A).

Suppose $r > q$. Let $v_1, \ldots, v_{r-1} \in P$ be the values in P other than k_{correct}. Then $v_1, \ldots, v_{r-1} \in P(t)$ for all $t \geq t_0$. For each such $t \geq t_0$, there can be at most q consensus values at time t, and one of these is k_{correct}, so at least one of $v_1, \ldots, v_{r-1}$ does not appear as a consensus value at time t. By the pigeon hole principle, there is some v_i that does not appear as a consensus value at time t for infinitely many $t \geq t_0$. For every $t \geq t_0$ we have $p_{k_{\text{correct}}}(t + 1) = p_{k_{\text{correct}}}(t) + 1$, and

$$p_{v_i}(t + 1) = \begin{cases} p_{v_i}(t) + 1 & \text{if } v_i \text{ is a consensus value at time } t + 1, \\ p_{v_i}(t) & \text{otherwise,} \end{cases}$$

and the second case occurs infinitely often. Thus there is some $t_2 \geq t_0$ such that $p_{v_i}(t_2) < p_{k_{\text{correct}}}(t_2)$, making v_i less popular than k_{correct} at time t_2. Thus $v_i \notin P(t_2)$, which is a contradiction. Hence, $r \leq q$, and we are done. □

To prove a separation, we describe classes $\mathcal{A}_1, \mathcal{A}_2, \ldots$ and prove that for any $n > 1$,

$$\mathcal{A}_n \in [1,n]\text{SUBSEQ-EX} - [1,n-1]\text{SUBSEQ-EX}.$$

Notation 3.20. For languages $A, B \subseteq \Sigma^*$, we write $A \subseteq^* B$ to mean that $A - B$ is finite.

Definition 3.21. For $i \geq 1$, let R_i be the language $(0^*1^*)^i$, and define

$$\mathcal{Q}_i = \{A \subseteq \{0,1\}^* : R_i \subseteq \text{SUBSEQ}(A) \subseteq^* R_i\}.$$

For all $n \geq 1$, define

$$\mathcal{A}_n = \mathcal{Q}_1 \cup \mathcal{Q}_2 \cup \cdots \cup \mathcal{Q}_n.$$

Note that $R_1 \subseteq R_2 \subseteq R_3 \subseteq \cdots$, but $R_{i+1} \not\subseteq^* R_i$ for any $i \geq 1$. This means that the $\mathcal{Q}_i$ are all pairwise disjoint. Also note that $\text{SUBSEQ}(R_i) = R_i$ for all $i \geq 1$.

Lemma 3.22. *For all $n > 1$, $\mathcal{A}_n \in [1,n]$SUBSEQ-EX and $\mathcal{A}_n \cap \text{DEC} \notin [1,n-1]$SUBSEQ-EX. In fact, there is a computable function $d(s)$ such that for all $n > 1$ and all $e_1, \ldots, e_{n-1}$, the machine $M_{d([e_1,\ldots,e_{n-1}])}$ decides a language $A_{[e_1,\ldots,e_{n-1}]} \in \mathcal{A}_n$ that is not learned by any of $M_{e_1}, \ldots, M_{e_{n-1}}$.*[2]

Proof. To see that $\mathcal{A}_n \in [1,n]$SUBSEQ-EX, let $Q_1, \ldots, Q_n$ be learners that behave as follows given a language A on their tapes: For $1 \leq i \leq n$, Q_i outputs $k_{i,1}, k_{i,2}, k_{i,3}, \ldots$, where $k_{i,j}$ is the least index of a DFA recognizing $R_i \cup \text{SUBSEQ}(A \cap \Sigma^{\leq j})$. Suppose $A \in \mathcal{Q}_i$ for some $1 \leq i \leq n$. We claim that Q_i learns $\text{SUBSEQ}(A)$. Since $A \in \mathcal{Q}_i$, there is a finite set D such that $\text{SUBSEQ}(A) = R_i \cup D$. For every $x \in D$, there is a $y \in A$ with $x \preceq y$. Because D is finite, this implies that $D \subseteq \text{SUBSEQ}(A \cap \Sigma^{\leq j})$ for all large enough j. Then for all such j,

$$\begin{aligned}&\text{SUBSEQ}(A) = R_i \cup D \subseteq \\ &\quad R_i \cup \text{SUBSEQ}(A \cap \Sigma^{\leq j}) \subseteq \text{SUBSEQ}(A) \cup \text{SUBSEQ}(A) = \text{SUBSEQ}(A),\end{aligned}$$

and thus $\text{SUBSEQ}(A) = R_i \cup \text{SUBSEQ}(A \cap \Sigma^{\leq j})$ for all large enough j. This proves the claim, and shows that $\mathcal{A}_n \in [1,n]$SUBSEQ-EX.

To show that $\mathcal{A}_n \notin [1,n-1]$SUBSEQ-EX effectively, we use the Recursion Theorem with Parameters to define a computable function $d(s)$ such that for all $n > 1$ and $e_1, \ldots, e_{n-1}$, the machine $M_{d([e_1,\ldots,e_{n-1}])}$ is total and decides a language $A = A_{[e_1,\ldots,e_{n-1}]} \in \mathcal{A}_n$, and $\text{SUBSEQ}(A)$ is not learned by any of $M_{e_1}, \ldots, M_{e_{n-1}}$. The machine $M_{d([e_1,\ldots,e_{n-1}])}$ has some input alphabet Σ such that $0, 1 \in \Sigma$, and it decides A via the following recursive algorithm:

On input $x \in \Sigma^*$:

[2] $[e_1, e_2, \ldots, e_{n-1}]$ is a natural number encoding the finite sequence $e_1, e_2, \ldots, e_{n-1}$.

1. If x is not of the form $(0^t1^t)^i$, where $t \geq 1$ and $1 \leq i \leq n$, then reject. (This ensures that $A \subseteq \{(0^t1^t)^i : (t \geq 1) \wedge (1 \leq i \leq n)\}$.) Otherwise, let t and i be such that $x = (0^t1^t)^i$.
2. Recursively compute $R_t = A \cap \{(0^s1^s)^\ell : (1 \leq s < t) \wedge (1 \leq \ell \leq n)\}$.
3. Compute $k_1(t), \ldots, k_{n-1}(t)$, the most recent outputs of $M_{e_1}, \ldots, M_{e_{n-1}}$, respectively, after running for t steps with R_t on their tapes. If some M_{e_j} has not yet output anything within t steps, then set $k_j(t) = 0$. (None of these machines has time to scan any tape cells corresponding to strings of the form $(0^u1^u)^\ell$ where $\ell \geq 1$ and $u \geq t$, so the machines' behaviors with R_t on their tapes are the same as with A on their tapes.)
4. Let $1 \leq i_t \leq n$ be least such that there is no $1 \leq j \leq n-1$ such that $L(F_{k_j(t)}) \in \mathcal{Q}_{i_t}$. (Such an i_t exists by the disjointness of the $\mathcal{Q}_i$ and by the pigeon hole principle, and we can compute such an i_t.)
5. If $i = i_t$, then accept; else reject.

By the pigeon hole principle, there is some largest $i_{\max}$ that is found in step 4 for infinitely many values of t. That is, $i_t = i_{\max}$ for infinitely many t, and $i_t > i_{\max}$ for only finitely many t.

We first claim that $A \in \mathcal{Q}_{i_{\max}}$, and hence $A \in \mathcal{A}_n$. Since A contains strings of the form $(0^t1^t)^{i_{\max}}$ for arbitrarily large t, it is clear that $R_{i_{\max}} \subseteq \text{SUBSEQ}(A)$. By the choice of $i_{\max}$, there is a t_0 such that A contains no strings of the form $(0^t1^t)^i$ where $i > i_{\max}$ and $t > t_0$. Therefore the set $D = A - R_{i_{\max}}$ is finite, and we also have

$$\text{SUBSEQ}(A) = R_{i_{\max}} \cup \text{SUBSEQ}(D).$$

Thus $\text{SUBSEQ}(A) \subseteq^* R_{i_{\max}}$, and so we have $A \in \mathcal{Q}_{i_{\max}}$.

We next claim that no M_{e_j} learns $\text{SUBSEQ}(A)$ for any $1 \leq j \leq n-1$. This is immediate by the choice of $i_{\max}$: For infinitely many t, none of the $k_j(t)$ satisfies $L(F_{k_j(t)}) \in \mathcal{Q}_{i_{\max}}$, and so none of the M_{e_j} can learn $\text{SUBSEQ}(A)$. □

Lemmas 3.19 and 3.22 combine to show the following general theorem, which completely characterizes the containment relationships between the various classes $[a, b]$SUBSEQ-EX.

Theorem 3.23. *For every $1 \leq a \leq b$ and $1 \leq c \leq d$, $[a, b]$SUBSEQ-EX $\subseteq$ $[c, d]$SUBSEQ-EX if and only if $\lfloor b/a \rfloor \leq \lfloor d/c \rfloor$.*

Proof. Let $p = \lfloor b/a \rfloor$ and let $q = \lfloor d/c \rfloor$. By Lemma 3.19, $[a, b]$SUBSEQ-EX $= [1, p]$SUBSEQ-EX and $[c, d]$SUBSEQ-EX $= [1, q]$SUBSEQ-EX. By Lemma 3.22, $[1, p]$SUBSEQ-EX $\subseteq [1, q]$SUBSEQ-EX if and only if $p \leq q$. □

4 Rich Classes

Are there classes in SUBSEQ-EX containing languages of arbitrary complexity? Yes, trivially.

Proposition 4.1. *There is a $\mathcal{C} \in \text{SUBSEQ-EX}_0$ such that for all $A \subseteq \mathbb{N}$, there is a $B \in \mathcal{C}$ with $B \equiv_T A$.*

Proof. Let

$$\mathcal{C} = \{A \subseteq \Sigma^* : |A| = \infty \wedge (\forall x, y \in \Sigma^*)[x \in A \wedge |x| = |y| \rightarrow y \in A]\}.$$

That is, $\mathcal{C}$ is the class of all infinite languages, membership in whom depends only on a string's length.

For any $A \subseteq \mathbb{N}$, define

$$L_A = \begin{cases} \Sigma^* & \text{if } A \text{ is finite,} \\ \bigcup_{n \in A} \Sigma^{=n} & \text{otherwise.} \end{cases}$$

Clearly, $L_A \in \mathcal{C}$ and $A \equiv_{\mathrm{T}} L_A$. Furthermore, $\mathrm{SUBSEQ}(L_A) = \Sigma^*$, and so $\mathcal{C} \in \text{SUBSEQ-EX}_0$ witnessed by a learner that always outputs a DFA for Σ^*. □

In Proposition 3.5 we showed that REG ∈ SUBSEQ-EX. Note that the $A \in$ REG are trivial in terms of computability, but the languages in SUBSEQ(REG) can be rather complex (large obstruction sets, arbitrary $\preceq$-closed sets). By contrast, in Proposition 4.1, we show that there can be $\mathcal{A} \in$ SUBSEQ-EX of arbitrarily high Turing degree but $\mathrm{SUBSEQ}(\mathcal{A})$ is trivial. Can we obtain classes $\mathcal{A} \in$ SUBSEQ-EX where $A \in \mathcal{A}$ has arbitrary Turing degree and $\mathrm{SUBSEQ}(\mathcal{A})$ has arbitrary $\preceq$-closed sets independently? Yes, subject to an obvious restriction.

Definition 4.2. A class $\mathcal{C}$ of languages is *rich* if for every $A \subseteq \mathbb{N}$ and $\preceq$-closed $S \subseteq \Sigma^*$, there is a $B \in \mathcal{C}$ such that $\mathrm{SUBSEQ}(B) = S$ and, provided S is infinite, $B \equiv_{\mathrm{T}} A$.

Definition 4.3. Let $\mathcal{G}$ be the class of all languages $A \subseteq \Sigma^*$ for which there exists a length $c = c(A) \in \mathbb{N}$ (necessarily unique) such that

1. $A \cap \Sigma^{=c} = \emptyset$,
2. $A \cap \Sigma^{=n} \neq \emptyset$ for all $n < c$, and
3. $os(A) = os(A \cap \Sigma^{\leq c+1}) \cap \Sigma^{\leq c}$.

In the full paper, we show the following:

Proposition 4.4. $\mathcal{G} \in \text{SUBSEQ-EX}_0$ *and* $\mathcal{G}$ *is rich.*

5 Open Questions

We can combine teams, mindchanges, and anomalies in different ways. For example, for which a, b, c, d, e, f, g, n is $[a, b]\text{SUBSEQ-EX}_c^d \subseteq [e, f]\text{SUBSEQ-EX}_g^h$? This problem has been difficult in the standard case of EX though there have been some very intersesting results [9, 5]. The setting of SUBSEQ-EX may be easier since all the machines that are output are total.

We can also combine the two notions of queries with SUBSEQ-EX and its variants. The two notions are allowing queries *about the set* [14, 12, 10] and allowing queries *to an undecidable set* [7, 17]. In the full paper, we show that $\text{CE} \in \text{SUBSEQ-EX}^{\emptyset'}$, where $\emptyset'$ is the halting problem.

References

1. G. Baliga and J. Case. Learning with higher order additional information. In *Proc. 5th Int. Workshop on Algorithmic Learning Theory*, pages 64–75. Springer-Verlag, 1994.
2. L. Blum and M. Blum. Towards a mathematical theory of inductive inference. *Information and Computation*, 28:125–155, 1975.
3. J. Case, S. Jain, and S. N. Manguelle. Refinements of inductive inference by Popperian and reliable machines. *Kybernetika*, 30–1:23–52, 1994.
4. J. Case and C. H. Smith. Comparison of identification criteria for machine inductive inference. *Theoretical Computer Science*, 25:193–220, 1983.
5. R. Daley, B. Kalyanasundaram, and M. Velauthapillai. Breaking the probability 1/2 barrier in FIN-type learning. *Journal of Computer and System Sciences*, 50:574–599, 1995.
6. S. Fenner, W. Gasarch, and B. Postow. The complexity of finding SUBSEQ(L), 2006. Unpublished manuscript.
7. L. Fortnow, S. Jain, W. Gasarch, E. Kinber, M. Kummer, S. Kurtz, M. Pleszkoch, T. Slaman, F. Stephan, and R. Solovay. Extremes in the degrees of inferability. *Annals of pure and applied logic*, 66:21–276, 1994.
8. R. Freivalds and C. H. Smith. On the role of procrastination for machine learning. *Information and Computation*, 107(2):237–271, 1993.
9. R. Freivalds, C. H. Smith, and M. Velauthapillai. Trade-off among parameters affecting inductive inference. *Information and Computation*, 82(3):323–349, Sept. 1989.
10. W. Gasarch, E. Kinber, M. Pleszkoch, C. H. Smith, and T. Zeugmann. Learning via queries, teams, and anomalies. *Fundamenta Informaticae*, 23:67–89, 1995. Prior version in *Computational Learning Theory (COLT)*, 1990.
11. W. Gasarch and A. Lee. Inferring answers to queries. In *Proceedings of* 10^{th} *Annual ACM Conference on Computational Learning Theory*, pages 275–284, 1997. Long version on Gasarch's home page, in progress, much expanded.
12. W. Gasarch, M. Pleszkoch, and R. Solovay. Learning via queries to $[+, <]$. *Journal of Symbolic Logic*, 57(1):53–81, Mar. 1992.
13. W. Gasarch, M. Pleszkoch, F. Stephan, and M. Velauthapillai. Classification using information. *Annals of Math and AI*, pages 147–168, 1998. Earlier version in *Proc. 5th Int. Workshop on Algorithmic Learning Theory*, 1994, 290–300.
14. W. Gasarch and C. H. Smith. Learning via queries. *Journal of the ACM*, 39(3):649–675, July 1992. Prior version in *IEEE Sym. on Found. of Comp. Sci. (FOCS)*, 1988.
15. E. M. Gold. Language identification in the limit. *Information and Computation*, 10(10):447–474, 1967.
16. A. G. Higman. Ordering by divisibility in abstract algebra. *Proc. of the London Math Society*, 3:326–336, 1952.
17. M. Kummer and F. Stephan. On the structure of the degrees of inferability. *Journal of Computer and System Sciences*, 52(2):214–238, 1996. Prior version in *Sixth Annual Conference on Computational Learning Theory (COLT)*, 1993.
18. H. Rogers. *Theory of Recursive Functions and Effective Computability.* McGraw-Hill, 1967. Reprinted by MIT Press, 1987.
19. G. E. Sacks. *Higher Recursion Theory.* Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1990.
20. R. Soare. *Recursively Enumerable Sets and Degrees.* Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987.