Extremes in the Degrees of Inferability

Lance Fortnow* — Univ. of Chicago

William Gasarch† — Univ. of Maryland

Sanjay Jain‡ — Nat. U. of Sing.

Efim Kinber§ — Univ. of Latvia

Martin Kummer¶ — Univ. Karlsruhe

Stuart Kurtz‖ — Univ. of Chicago

Mark Pleszkoch** — IBM Corporation

Theodore Slaman†† — Univ. of Chicago

Robert Solovay‡‡ — Univ. of California

Frank Stephan *** — Univ. Karlsruhe

*Department of Computer Science, University of Chicago, Chicago, ILL 60637. Supported in part by NSF grant CCR 90-09936 (fortnow@cs.uchicago.edu).

†Dept. of C.S. and Inst. for Adv. Stud., Univ. of MD., College Park, MD 20742. Supp. in part by NSF grants CCR-8803641 and CCR-9020079 (gasarch@cs.umd.edu).

‡Institute of Systems Science, National University of Singapore, Heng Mui Keng Terrace, Kent Ridge, Singapore 0511, Republic of Singapore, (sanjay@iss.nus.sg).

§Inst. of Math and C.S., Univ. of Latvia, Raiņa bulvāris 29 (kinber@mii.lu.lv).

¶Institut für Logik, Komplexität und Deduktionssysteme, Postfach 6980, D-7500 Karlsruhe 1, Germany, (kummer@ira.uka.de).

‖University of Chicago, Department of Computer Science, Chicago, ILL 60637. (stuart@cs.uchicago.edu)

**IBM Corp. 100 Lakeforest Blvd., Gaithersburg, MD., 20877. (markp@vnet.ibm.com)

††Department of Mathematics, University of Chicago, Chicago, ILL 60637. Supported in part by NSF grant DMS 88-902437 (slaman@math.uchicago.edu).

‡‡Department of Mathematics, University of California at Berkeley, Berkeley, CA Supported in part by NSF grant DMS 88-902437 (solovay@math.berkeley.edu)

***Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, Postfach 6980, D-7500 Karlsruhe 1, Germany, (fstephan@ira.uka.de). Supported by the Deutsche Forschungsgemeinschaft (DFG) grant Me 672/4-1

1 Introduction

Most theories of learning (e.g., [Gol67, Val84]) have dealt with learning a function f by observing the behavior of f. This roughly models learning from data. In the last few years theories have been developed that allow the learner to ask questions about the function (e.g., [Ang88, GS92]). This roughly models learning from a helpful teacher.

In this paper we consider the scenario where the learner can ask a fellow student questions. Note that the other student does not know anything more about the function f than the learner; however, she might still be helpful. Less whimsically, we wish to investigate how information, not necessarily related to the function being learned, may still help in the learning of that function.

We will consider several recursion-theoretic models of learning. In these models the learner, while trying to infer a recursive function f, is able to both observe increasing portions of the graph of f and query an oracle A. The power of the learner, measured as its ability to learn collections of functions, depends on both the learning model and the oracle A.

In this paper we examine extremes: Which oracles add no power to the learner? Which oracles allow the learner to infer the set of all recursive functions? Kummer and Stephan [KS93b] have investigated structural issues. For several models they show exactly when oracle A is weaker (in terms of learning sets of recursive functions) than oracle B.

Some of the results in this paper were announced in [GP89] and [CDF+92].

2 Definitions and Notation

2.1 Standard Definitions

Notation 2.1 We denote the natural numbers by $\mathbb{N} = \{0, 1, 2, \ldots\}$. We denote subsets of $\mathbb{N}$ by capital letters (usually A or B) and elements of $\mathbb{N}$ by small letters (usually n, m).

Notation 2.2 Throughout this paper $M_0, M_1, \ldots$ is a standard list of all Turing machines, $M_0^{()}, M_1^{()}, \ldots$ is a standard list of all oracle Turing machines, $\varphi_0, \varphi_1, \ldots$ is the acceptable programming system, obtained by letting φ_e be the partial recursive function computed by M_e. The domain of φ_e is denoted W_e.

Notation 2.3 Let $M_{e,s}$ be the machine that, on input x, runs $M_e(x)$ for s steps, outputs $M_e(x)$ if the computation has halted within s steps, and diverges otherwise. Let $\varphi_{e,s}$ be the partial function computed by $M_{e,s}$. Let $W_{e,s} = \{0, \ldots, s\} \cap dom(\varphi_{e,s})$. Let $\Phi_0, \Phi_1, \ldots$ be the Blum complexity measure defined by the number of steps the Turing machine takes. (Using Turing machine steps as a measure of complexity is not crucial. We could have used any acceptable programming system and any Blum measure. We use Turing machines and runtimes so we can speak of 'running a machine for s steps'.)

Our definition of high and low differ slightly from that in [Soa87], so we state our definition.

Notation 2.4 A' is the halting problem relative to A, that is, $\{e : M_e^A(e) \text{ halts }\}$. A is *high* if $\emptyset'' \leq_{\mathrm{T}} A'$. A is *high*$_2$ if $\emptyset''' \leq_{\mathrm{T}} A''$. A is *low* if $A' \leq_{\mathrm{T}} K$.

Notation 2.5 Let σ, τ be strings over an alphabet Σ. $|\sigma|$ denotes the length of σ. $\sigma \preceq \tau$ means that σ is a prefix of τ. We think of σ as being a map from $\{0, 1, \ldots, |\sigma| - 1\}$ to Σ. If $a \in \mathbb{N}$ then we use σa^ω to denote the total function whose characteristic string has initial segment σ and then consists of all a's.

Notation 2.6 Let $\sigma \in \{0, 1\}^*$ and $M^{()}$ be an oracle Turing machine. M^σ is the Turing machine that attempts to simulate $M^{()}$ by answering questions as though σ were an initial segment of the oracle. If ever a query is made that is bigger than $|\sigma| - 1$, then the computation diverges. Any divergent computation that results from running $M^\sigma(x)$ is denoted by $M^\sigma(x) \uparrow$.

Notation 2.7 $A \upharpoonright x$ denotes $A \cap \{0, 1, \ldots, x\}$. If A is r.e. then A_s denote the first s elements in some fixed recursive enumeration.

Notation 2.8 $(\stackrel{\infty}{\exists} x)$ means 'for an infinite number of x.' $(\stackrel{\infty}{\forall} x)$ means 'for all but a finite number of x;' equivalently, 'for almost all x.'

Notation 2.9 If f and g are two functions then $f =^* g$ means that $(\stackrel{\infty}{\forall} x)[f(x) = g(x)]$. If a is a constant then $\lambda x[a]$ denotes the constant function that always outputs a. The expression $f =^* \lambda x[a]$ means that $(\stackrel{\infty}{\forall} x)[f(x) = a]$.

Remaining recursion-theoretic notation is from [Soa87].

2.2 Definitions from Inductive Inference

We consider the learnability of collections of recursive functions. We are especially interested in situations that render the entire collection of recursive functions learnable by a single machine.

Notation 2.10 We denote the set of all recursive functions by REC.

The following definitions are from [CS83].

Definition 2.11 An *inductive inference machine* (IIM) M is a total Turing machine. We interpret M to be trying to learn a recursive function f by viewing M as taking as input the values $f(0), f(1), \ldots$ (one value at a time) and producing output (from time to time) in the form of a program intended to compute f. If almost all the programs are the same, and compute f, then we say that M *EX-identifies* f. If almost all the programs compute f, but are not necessarily the same, then M *BC-identifies* f. Formally, M computes a total function from $\mathbb{N}^*$ (finite sequences of natural numbers) to $\mathbb{N}$. The input is an initial segment of the function to be inferred, and the output is the current guess as to the index of that function. The indices output by IIMs are relative to the acceptable programming system $\{\varphi_i\}_{i=0}^{\infty}$ specified in Notation 2.2.

Convention 2.12 If the output of an IIM is 0 then we interpret this as meaning 'no guess at this time'. Formally an IIM M takes as input elements of the form $\langle\sigma_1, \ldots, \sigma_n\rangle$ or $\langle f(0), \ldots, f(n)\rangle$, but we denote $M(\langle\sigma_1, \ldots, \sigma_n\rangle)$ by $M(\sigma_1, \ldots, \sigma_n)$, and $M(\langle f(0), \ldots, f(n)\rangle)$ by $M(f(0), \ldots, f(n))$.

Definition 2.13 Let $S \subseteq REC$. Then $S \in EX$ (BC) if there exists an IIM M such that for all $f \in S$, M EX-identifies f (BC-identifies f).

Note 2.14 If we did not require IIMs to be total, then the class EX would not change. If M is a (not necessarily total) Turing machine that we want to use as an IIM then we define M' to simulate M as follows. To compute $M'(f(0), \ldots, f(n))$ find the largest $i \leq n$ (if it exists) such that $M(f(0), \ldots, f(i))$ halts within n^2 steps, and output the answer produced; if no such i exists then output 0. Clearly M' EX-identifies all the functions that M did. These remarks also apply to BC. (The choice of time bound is not important.)

Example 2.15 Let $S_1 = \{f : \varphi_{f(17)} = f\}$ and $S_2 = \{f : f =^* \lambda x[0]\}$. Note that $S_1 \in EX$ and $S_2 \in EX$. Techniques of the Blums [BB75], or Case and Smith [CS83], can be used to show that $S_1 \cup S_2 \notin EX$. Hence, $REC \notin EX$.

We consider using (categorical) oracle Turing machines instead of (total) Turing machines.

Definition 2.16 An oracle Turing machine $M^{()}$ is *categorically total* if, for every X, M^X is total. Note that if $M^{()}$ is categorical then, given σ and x, one can test if $M^\sigma(x) \uparrow$ recursively.

Definition 2.17 An *oracle inductive inference machine* (OIIM) $M^{()}$ is a categorically total Turing machine. We interpret M^A to be trying to learn a recursive function f similar to our interpretation of an IIM M trying to learn a recursive function f. We define M^A $EX[A]$*-identifies* f ($BC[A]$*-identifies* f) similar to our definition of M EX-identifies f (BC-identifies f).

Definition 2.18 Let S be a set of recursive functions. $S \in EX[A]$ $(BC[A])$ is defined similar to $S \in EX$ (BC).

Note 2.19 If we did not require OIIMs to be categorically total, then the classes $EX[A]$ would not change. If $M^{()}$ is a (not necessarily categorical) Turing machine that we want to use as an OIIM then we define M'^A to simulate M^A as follows. To compute $M'^A(f(0), \ldots, f(n))$, find the largest $i \leq n$ (if it exists) such that $M^A(f(0), \ldots, f(i))$ halts within n^2 steps, and output the answer produced; if no such i exists then output 0. Clearly M'^A $EX[A]$-identifies all the functions that M^A did. These remarks also apply to $BC[A]$. (The choice of time bound is not important.)

Note that in the definition of $EX[A]$ (and the other classes) we are inferring indices for recursive functions, *not* indices for recursive-in-A functions.

Example 2.20 $REC \in EX[K]$ via the following well-known inference procedure (introduced in [Gol67]). Upon seeing initial segment σ, output the least e such that for every $x \in dom(\sigma)[\varphi_e(x)\downarrow= \sigma(x)]$. Note that even after the correct index is found infinitely many queries to K are made to keep verifying that the index is correct.

Note 2.21 Using $REC \in EX[K]$ we can give an easy proof of a theorem of Harrington. Define BC^* as follows: $S \in BC^*$ if there exists an IIM M such that, for all $f \in S$, when f is fed into M almost all the programs output compute functions that are equal to f almost everywhere. Harrington showed (see [CS83]) that $REC \in BC^*$. We give an alternate proof. Let $M^{()}$ be the OIIM such that M^K infers REC. Let N be the IIM that operates as follows: $\varphi_{N(\sigma)}(s) = \varphi_{M^{K_s}(\sigma)}(s)$. It is easy to show that N BC^*-infers REC.

Example 2.22 Let

$$S = \{f : f(0) \in A' \wedge \varphi_{f(1)} = f\} \cup \{f : f(0) \notin A' \wedge f =^* \lambda x[0]\}.$$

Note that $S \in EX[A]$ by using an A-approximation to A', which exists by the Limit Lemma (see [Soa87, p. 57]).

Definition 2.23 Let S be a set of recursive functions. $S \in EX[A*]$ if there exists an OIIM $M^{()}$ such that (1) S is $EX[A]$-identified by M^A, and (2) for every $f \in S$, during the inference of f by M^A, only finitely many queries to A are made. $BC[A*]$ is defined similarly.

Example 2.24 Note that $REC \in EX[TOT*]$ by asking if a machine is total before considering it. Also note that the proof of $REC \in EX[K]$ uses an OIIM M^K that makes infinitely many queries to K; we later show (Theorem 5.7) that $REC \notin EX[K*]$.

Definition 2.25 Let S be a set of recursive functions. $S \in EX[A[m]]$ if there exists an OIIM $M^{()}$ such that (1) S is $EX[A]$-identified by M^A, and (2) for every $f \in S$, during the inference of f by M^A, at most m queries to A are made. $BC[A[m]]$ is defined similarly.

Example 2.26 Let

$$T_1 = \{f : (\exists i \le 63)[(f(0), \ldots, f(i) \in A) \wedge (f(i+1), \ldots, f(63) \notin A) \wedge (\varphi_{f(i)} = f)\}.$$

It is easy to see that $T_1 \in EX[A[64]]$. Using binary search one can obtain $T_1 \in EX[A[6]]$. (Actually $T_1 \in EX$ since $|T_1|$ is finite.)

Let

$$T_2 = \{f : (\exists i)[(f(0), f(1), \ldots, f(i) \in A) \wedge (f(i+1) \notin A) \wedge (\varphi_{f(i)} = f)\}.$$

It is easy to see that $T_2 \in EX[A*]$.

Let

$$T_3 = \{f : f(0) \in A \wedge \varphi_{f(1)} = f\} \cup \{f : f(0) \notin A \wedge f =^* \lambda x[0]\}.$$

It is easy to see that $T_3 \in EX[A[1]]$.

Definition 2.27 $A \le_{\mathrm{i}} B$ if $EX[A] \subseteq EX[B]$ (the 'i' stands for 'inference'). $A \equiv_{\mathrm{i}} B$ if $EX[A] = EX[B]$. An *EX-degree* is an equivalence class under $\equiv_{\mathrm{i}}$. $A \le_{\mathrm{i}}^* B$ if $EX[A*] \subseteq EX[B*]$. $A \equiv_{\mathrm{i}}^* B$ if $EX[A*] = EX[B*]$. An *EX∗-degree* is an equivalence class under $\equiv_{\mathrm{i}}^*$. The BC-degrees and $BC*$-degrees are defined similarly. More generally, these definitions would make sense for any of the classes usually studied in inductive inference. These degree structures are spoken of informally as the *degrees of inferability.*

Definition 2.28 An EX-degree is *trivial* if for every A in that degree, $EX[A] = EX$. An EX-degree is *omniscient* if for every A in that degree $REC \in EX[A]$. The notions trivial and omniscient can be defined for other types of degrees of inferability.

Notation 2.29 Let $\sigma \in \mathbb{N}^*$, $A \subseteq \mathbb{N}$, f be a function, and $M^{()}$ be an OIIM. $\sigma \preceq f$ means that σ is an initial segment of f. $M^A(\sigma)$ is a guess for an index for f; note that $M^A(\sigma)$ does not have access to $f(|\sigma|)$. We will often try to diagonalize by looking at $\varphi_{M^A(\sigma)}(|\sigma|)$.

We will need the notion of team inference while investigating EX and BC degrees of inferability. We will study degrees of inferability relative to team inference itself in Section 6.1

Definition 2.30 Let a, b be such that $1 \le a \le b$. A set of recursive functions S is in $[a, b]EX$ (concept from [Smi82], notation from [PS88]) if there exist b IIMs $M_1, M_2, \ldots, M_b$ such that, for every $f \in S$, there exist $i_1, \ldots, i_a$, $1 \le i_1 < \cdots < i_a \le b$, such that $M_{i_1}, \ldots, M_{i_a}$ all EX-infer f. If $a = 1$ then in the literature this is referred to as inferring S by a *team of b IIMs*. $[a, b]BC$ is defined similarly.

Convention 2.31 In this paper when the notation $[a, b]EX$ is used it is assumed that $1 \le a \le b$.

We will need the following construct.

Definition 2.32 If I is a finite set of indices for Turing machines then the partial recursive function $AM(I)$ is computed as follows: on input x run, for every $e \in I$, $\varphi_e(x)$ (dovetail over all $e \in I$). Whichever one halts first (if any), output its answer. The 'AM' stands for *amalgamation*. We write $AM(i_1, \ldots, i_n)$ instead of the (formally correct) $AM(\{i_1, \ldots, i_n\})$.

3 Technical Summary

We examine when a degree of inferability can be trivial, and when it can be omniscient. We then extend these questions to other notions of inferability. This paper, together with [AB91, JS, KS93b], describes all that is known for these questions. All results listed are in this paper unless otherwise noted. A more comprehensive summary is in Section 7.

Notation 3.1 $\mathcal{G}(A)$ stands for the condition that either A is recursive, or $A \leq_{\mathrm{T}} K$ and is in a 1-generic Turing degree.

1) When are EX-degrees (and variations) trivial?

   a) If $EX[A] = EX$ then A is low.

   b) $EX[A] = EX$ iff $\mathcal{G}(A)$. (The backwards direction is in this paper. The forward direction was first shown in [SS91] and is quite difficult. An easier proof appears in [KS93b].) This supersedes item *a*), but *a*) has an easy proof.

   c) $EX[A*] = EX$ iff $A \leq_{\mathrm{T}} K$.

   d) $(\forall m)[EX[A[m]] = EX$ iff $A \leq_{\mathrm{T}} K]$.

2) When are BC-degrees (and variations) trivial?

   a) If $BC[A] = BC$ then A is low.

   b) $BC[A] = BC$ iff $\mathcal{G}(A)$. (The backwards direction is in this paper. The forward direction can be obtained by a modification of the proof of a similar result for EX which is in [SS91], or directly in [KS93b].) This supersedes item *a*), but *a*) has an easy proof.

   c) $BC[A*] = BC$ iff $A \leq_{\mathrm{T}} K$.

   d) $(\forall m)[BC[A[m]] = BC$ iff $A \leq_{\mathrm{T}} K]$.

3) When are EX-degrees (and variations) omniscient?

   a) $REC \in EX[A]$ iff A is high (proven in [AB91]). Also see [KS93b]).

   b) $REC \in EX[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

c) $(\forall m)(\forall A)[REC \notin EX[A[m]]]$.

4) When are BC-degrees (and variations) omniscient?

   a) If A is r.e. then $REC \in BC[A]$ iff A is high.

   b) There exists a low set A such that $REC \in BC[A]$.

   c) For all X there exists A such that $X \leq_{\mathrm{T}} A''$ and $REC \notin BC[A]$.

   d) $REC \in BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

   e) $(\forall m)(\forall A)[REC \notin BC[A[m]]]$.

5) Other notions of inference. For most other degrees of inference, those that are variations on $EX[A]$ ($EX[A*]$, $BC[A]$, $BC[A*]$) act very much like $EX[A]$ ($EX[A*]$, $BC[A]$, $BC[A*]$). There are two notable exceptions. (1) PEX is the set of all $S \subseteq REC$ such that S can be inferred by a machine that outputs only indices for total functions. The PEX-degrees and $PEX*$-degrees seem to behave in a manner quite different from the EX-degrees and $EX*$-degrees. (2) If the number of mindchanges that an OIIM can make is bounded, the resulting inference-degrees behave exactly like the Turing-degrees, which is not at all similar to the EX-degrees. See Section 6 for definitions of other notions of inference, and the table in Section 7 for a summary of results.

It is open to determine when $REC \in BC[A]$. Our results suggest that there is no nice characterization of such A.

The sections of this paper are organized as follows.

1. Introduction
2. Definitions and Notation
   2.1. Standard Definitions
   2.2. Definitions from Inductive Inference
3. Technical Summary
4. When are Degrees Trivial?
   4.1 $EX[A[m]]$, $BC[A[m]]$, $EX[A*]$, and $BC[A*]$
   4.2 $EX[A]$ and $BC[A]$
5. When are Degrees Omniscient?
   5.1 $EX[A[m]]$ and $BC[A[m]]$
   5.2 $EX[A*]$ and $BC[A*]$

5.3 $EX[A]$ and $BC[A]$
5.3.1 The r.e. case
5.3.2 A Low A such that $REC \in BC[A]$
5.3.3 Arbitrary High Double Jump is Not Enough
6. Other Notions of Inference
6.1 $[a,b]EX$– Teams of Machines
6.1.1 $*$-Triviality
6.1.2 Triviality
6.1.3 $*$-Omniscience
6.1.4 Omniscience for $[a,b]EX$
6.1.5 Omniscience for $[a,b]BC$
6.2 EX^n, EX^*, BC^n, BC^*– Allowing Errors
6.3 EX_n– Bounding Mind Changes
6.4 PEX– Guesses are total
6.5 Combinations
7. Conclusions and Open Problems
8. Acknowledgments

4 When are Degrees Trivial?

Slaman and Solovay proved the following theorem (see [KS93b] for an easier proof). We state it here so we can refer to it.

Theorem 4.1 ([SS91]) *If $EX = EX[A]$ then $\mathcal{G}(A)$.*

We will need the following concepts from the theory of bounded queries. They were introduced in [BGGO93]. We use them in Sections 4.1 and 6.3.

Definition 4.2 F_k^A is the function with domain $\mathbb{N}^k$ and range $\{0,1\}^k$ defined by

$$F_k^A(x_1, \ldots, x_k) = \chi_A(x_1) \cdots \chi_A(x_k).$$

We will need to code the output of F_k^A as a natural number.

Definition 4.3 Let NUM denote the function from $\{0,1\}^*$ to $\mathbb{N}$ that maps τ to the number it represents in binary (formally τ maps to $\sum_{\tau(i)=1} 2^{|\tau|-i-1}$). Note that $NUM(\tau 1) = 2NUM(\tau) + 1$ and $NUM(\tau 0) = 2NUM(\tau)$.

Clearly F_k^A can be computed with k queries to A. In [BGGO93] it was shown that for all k, A, X, Y the following holds: if $F_{2^k}^A$ can be computed with k queries to X and arbitrary queries to Y then $A \leq_{\rm T} Y$. The following proposition is equivalent to this statement. (The proof of the proposition, and the equivalence, are in [BGGO93].)

Convention 4.4 Throughout this section we take $W_0, W_1, \ldots$ to be an enumeration of all r.e. subsets of $\{0,1\}^*$, instead of r.e. subsets of $\mathbb{N}$.

Definition 4.5 A function f is *m-enumerable-in-Y* if there exists a recursive function h such that for all x, $|W_{h(x)}| \leq m$ and $f(x) \in W_{h(x)}$.

Proposition 4.6 *If $F_m^A(x_1, \ldots, x_m)$ is m-enumerable-in-Y then $A \leq_{\rm T} Y$.*

Note 4.7 Much more is known about bounded queries. We state two results, one of which we will refer to in later notes.

i. Kummer[Kum92] showed that Proposition 4.6 hold if $F_m^A(x_1, \ldots, x_m)$ is replaced by the function $\#_m^A(x_1, \ldots, x_m) = |\{i : x_i \in A\}|$.

ii. Kummer and Stephan [KS93a] have shown that for any nonrecursive set A there is a function $TREE_k^A$ that can be computed with k queries to A that cannot be computed with $k-1$ queries to *any* set. This will be used in two places (Notes 4.16 and 6.35) to slightly improve our results.

4.1 $EX[A[m]]$, $BC[A[m]]$, $EX[A*]$, and $BC[A*]$

Kinber [Kin90] showed that there exists a set A such that, for all i, $EX[A[i]] \subset EX[A[i+1]]$. It was later shown (see [GJPS91]) that, for all i, $EX[FIN[i]] \subset EX[FIN[i+1]]$.

We show that $EX[A*] = EX$ iff $A \leq_{\rm T} K$. Hence when $A \leq_{\rm T} K$ we have $EX = EX[A[1]] = EX[A[2]] = \cdots = EX[A*]$. The question arises as to when and how query hierarchies can collapse. The following theorem answers virtually all questions that could be asked.

Theorem 4.8 *The following are equivalent.*

i. $A \leq_{\mathrm{T}} K$.

ii. $(\exists n)[EX[A[n]] = EX[A[n+1]]]$.

iii. $(\forall n)[EX[A[n]] = EX[A[n+1]]]$.

iv. $EX = EX[A*]$.

v. $(\exists n)[BC[A[n]] = BC[A[n+1]]]$.

vi. $(\forall n)[BC[A[n]] = BC[A[n+1]]]$.

vii. $BC = BC[A*]$.

We prove this after establishing several lemmas.

Lemma 4.9 *If $A \leq_{\mathrm{T}} K$, then $EX[A*] = EX$ and $BC[A*] = BC$.*

Proof:
Let $A \leq_{\mathrm{T}} K$ and let S be $EX[A*]$-identified via M^A. We show $S \in EX$.

Since $A \leq_{\mathrm{T}} K$, by the Limit Lemma (see [Soa87, p. 57]), there exists a recursive function $h(x,s)$ such that $A(x) = \lim_{s\to\infty} h(x,s)$. Let $A_s = \{0,\ldots,s\} \cap \{x : h(x,s) = 1\}$.

We define an IIM M' that infers S. On input σ, M' outputs $M^{A_{|\sigma|}}(\sigma)$.

We show that if $f \in S$ then M' infers f. Let $f \in S$. Let $x_0, \ldots, x_m$ be the set of all queries that M^A makes while inferring f. Let n be the least number such that $(\forall i \leq m)(\forall s \geq n)[x_i \in A \text{ iff } x_i \in A_s]$. It is easy to see that $(\forall \sigma \preceq f)[|\sigma| \geq n \Rightarrow M'(\sigma) = M^A(\sigma)]$. Hence M' infers f.

The proof for BC is similar. ∎

Note 4.10 The proof of Lemma 4.9 can be easily modified to show that, for all B, $EX[(B \oplus K)*] = EX[B*]$ and $BC[(B \oplus K)*] = BC[B*]$. We will use this later in Lemma 5.6 to help prove Theorem 5.8. It can be further modified to show that $[a,b]EX[(B \oplus K)*] = [a,b]EX[B*]$ and $[a,b]BC[(B \oplus K)*] = [a,b]BC[B*]$. We will use this later in Lemma 6.5 to help prove Theorem 6.6

The next lemma we prove about $EX[A[m]]$ (and $BC[A[m]]$) gives another way of dealing with these classes. It will be useful in proving Lemma 4.13. We will also use it later to prove Theorem 5.2.

Lemma 4.11 *For all m, A, $EX[A[m]] \subseteq [1, 2^m]EX$ and $BC[A[m]] \subseteq [1, 2^m]BC$.*

Proof:

We show $EX[A[m]] \subseteq [1, 2^m]EX$. Let $S \in EX[A[m]]$ via M^A. We define 2^m IIMs as follows: for every string $\sigma \in \{0,1\}^m$, let $M(-;\sigma)$ be the IIM that simulates M^A by answering the i^{th} query with the i^{th} bit of σ. If more than m queries are made, then $M(-;\sigma)$ outputs 0 thereafter. For every $f \in S$, there exists a $\tau \in \{0,1\}^*$, $|\tau| \leq m$, such that τ contains all the correct answers to queries asked by M^A while inferring f. Let σ be such that $|\sigma| = m$ and τ is a prefix of σ. It is easy to see that f is inferred by $M(-;\sigma)$.

The proof for $BC[A[m]] \subseteq [1, 2^m]BC$ is similar. ∎

Definition 4.12 Let S_m^A be the set of recursive functions f such that the following holds.

i. There exist $a, b, x_1, \ldots, x_m, d$ such that $f(0) = \langle a, b, x_1, \ldots, x_m \rangle$ and $d = a + (b \times NUM(F_m^A(x_1, \ldots, x_m)))$.

ii. There exists e such that

  (a) if d is as in part i then for almost all k, $f(\langle d, k \rangle) = e$, and

  (b) for all $x \geq 1$, $f(x) = \varphi_e(x)$.

If $f \in S_m^A$ then f codes an index for a function that is identical to f except at 0. However, knowledge of A is needed to know *where* in f to look for this index.

Lemma 4.13 *The following are true for all A, m, n.*

i. $S_m^A \in EX[A[m]]$.

ii. $S_m^A \in EX[A[n]] \Rightarrow S_{m+1}^A \in EX[A[n+1]]$.

iii. $(\exists m')[S_{m'}^A \notin EX[A[m]]] \Rightarrow (EX[A[m]] \subset EX[A[m+1]])$.

iv. $EX[A[m]] = EX[A[m+1]] \Rightarrow (\forall m')[S^A_{m'} \in EX[A[m]] \subseteq [1, 2^m]EX]$.

v. $BC[A[m]] = BC[A[m+1]] \Rightarrow (\forall m')[S^A_{m'} \in BC[A[m]] \subseteq [1, 2^m]BC]$.

Proof:

i. We infer S^A_m as follows. Upon seeing $f(0) = \langle a, b, x_1, \ldots, x_m \rangle$ we ask the m queries '$x_i \in A$?' ($1 \le i \le m$) and compute $d = a + (b \times NUM(F^A_m(x_1, \ldots, x_m)))$. Henceforth, whenever $f(\langle d, k \rangle) = e$ is observed output $s(e)$ where s is the total recursive function defined by

$$\varphi_{s(e)}(x) = \begin{cases} f(0) & \text{if } x = 0; \\ \varphi_e(x) & \text{otherwise.} \end{cases}$$

By the definition of S^A_m almost all guesses will be the same and will be indices for f.

ii. Assume $S^A_m \in EX[A[n]]$ via M^A. We infer S^A_{m+1} with $n+1$ queries to A as follows. Upon seeing $f(0) = \langle a, b, x_1, \ldots, x_{m+1} \rangle$ query '$x_{m+1} \in A$?' If YES then feed the graph of the following function into M^A.

$$h(x) = \begin{cases} \langle a+b, 2b, x_1, \ldots, x_m \rangle & \text{if } x = 0; \\ f(x) & \text{otherwise.} \end{cases}$$

Note that $h \in S^A_m$ (this uses part ii.b of the definition of S^A_m, and $NUM(\tau 1) = 2NUM(\tau) + 1$). Hence M^A will correctly infer h and make only n queries. Whenever M^A outputs index e we output $s(e)$ where s is the total recursive function defined by

$$\varphi_{s(e)}(x) = \begin{cases} f(0) & \text{if } x = 0; \\ \varphi_e(x) & \text{otherwise.} \end{cases}$$

Since M^A correctly infers h our process correctly infers f. Since the only queries made are '$x_{m+1} \in A$?' and the $\le n$ queries in the inference of h by M^A, a total of $\le n+1$ queries to A are made.

If the answer to '$x_{m+1} \in A$?' had been NO then use the function

$$h(x) = \begin{cases} \langle a, 2b, x_1, \ldots, x_m \rangle & \text{if } x = 0; \\ f(x) & \text{otherwise} \end{cases}$$

and proceed as above.

iii. Assume there exists m' such that $S^A_{m'} \notin EX[A[m]]$. By *i*, $S^A_m \in EX[A[m]]$. Hence there exists m'' such that $S^A_{m''} \in EX[A[m]]$ but $S^A_{m''+1} \notin EX[A[m]]$. By *ii*, $S^A_{m''+1} \in EX[A[m+1]]$. Hence $S^A_{m''+1} \in EX[A[m+1]] - EX[A[m]]$ so $EX[A[m]] \subset EX[A[m+1]]$.

iv. If $EX[A[m]] = EX[A[m+1]]$ then, by *iii*, for all m', $S^A_{m'} \in EX[A[m]]$. By Lemma 4.11, $EX[A[m]] \subseteq [1, 2^m]EX$. Hence, for all m', $S^A_{m'} \in [1, 2^m]EX$.

v. The proofs of *i*, *ii*, *iii*, *iv* hold for BC by replacing EX by BC. ▮

Lemma 4.14 *If $S^A_m \in [1, m]BC$ then $A \leq_{\mathrm{T}} K$.*

Proof:

Throughout this proof 'infer' means 'BC-infer'.

Assume $S^A_m \in [1, m]BC$ via $M_1, \ldots, M_m$. We use $M_1, \ldots, M_m$ to show that F^A_m is m-enumerable-in-K. By Proposition 4.6 this will imply $A \leq_{\mathrm{T}} K$.

On input $x_1, \ldots, x_m$ we try to construct recursive functions $\{f_\tau\}_{\tau \in \{0,1\}^m}$ (which depend on $x_1, \ldots, x_m$) such that, for all τ, f_τ is not inferred by any of $M_1, \ldots, M_m$. We will fail—one of the f_τ will be partial. This failure will yield information about A. Using oracle K we will find τ such that f_τ is partial. We will then enumerate τ as a possibility for $F^A_m(x_1, \ldots, x_m)$. Using information about f_τ we will construct a new set of functions that may yield another possibility for $F^A_m(x_1, \ldots, x_m)$. This process may be repeated; however, at most m possibilities will be enumerated.

ENUMERATION

i. $P := \{0,1\}^m$, $J := \{1, \ldots, m\}$, and $g := \{(0, \langle 0, 1, x_1, \ldots, x_m\rangle)\}$. ($P$ stands for possibilities for $F^A_m(x_1, \ldots, x_m)$.) We construct $\{f_\tau\}_{\tau \in P}$ to diagonalize against all IIMs in $\{M_j\}_{j \in J}$. The function g will be a subfunction of every f_τ. In the future this will guarantee that f_τ is not inferred by any IIM in $\{M_j\}_{j \in \{1,\ldots,m\} - J}$.)

ii. Compute indices for the functions in $\{f_\tau\}_{\tau \in P}$ described below.

   CONSTRUCTION OF f_τ

   (a) *Stage 0:* Let e be an index for f_τ (obtained via the recursion theorem). Let

$$f^0_\tau = g \cup \{(\langle NUM(\tau), k\rangle, e) : \langle NUM(\tau), k\rangle \notin dom(g)\}.$$

(This step needs an index for $dom(g)$. During the first execution of step ii this is trivial. In later executions the index will come from the last execution of step iii.c.)

(b) *Stage* $s+1$*:* If there is no $j \in J$ such that $j \equiv s \pmod{m}$ then go to stage $s+2$. If there is such a j then look for $\sigma \in \mathbb{N}^*, t \in \mathbb{N}, b \in \{0,1\}$ such that σ is consistent with f_τ^s, $|\sigma| \notin dom(f_\tau^s)$, and $\varphi_{M_j(\sigma)}(|\sigma|) \downarrow \neq b$. If such are found then set

$$f_\tau^{s+1} := f_\tau^s \cup \{(x, \sigma(x)) : x \in dom(\sigma)\} \cup \{(|\sigma|, b)\}.$$

END OF CONSTRUCTION

iii. Using oracle K search for $\tau \in P$, $j \in J$, such that f_τ is partial and during the stage where f_τ could not be extended the construction was working with machine M_j. (This search will terminate the first time step iii is executed but need not terminate in a later execution.) If such τ, j are found then do the following.

(a) Enumerate τ (τ is a possibility for $F_m^A(x_1, \ldots, x_m)$).

(b) $P := P - \{\tau\}$, $J := J - \{j\}$, $g := f_\tau$. (Note that all total extensions of g are not inferred by M_j. Inductively, all total extensions of g are not inferred by any IIM in $\{M_j\}_{j \in \{1,\ldots,m\}-J}$, and g is undefined on almost all elements in $\{\langle NUM(\tau), k\rangle : \tau \in P, k \in \mathbb{N}\}$.

(c) Using oracle K find the index for a machine that decides $dom(g)$. This is easy since g is the union of a finite function with the current f_τ. (This index is needed in the next execution of step ii.)

(d) If $J = \emptyset$ then halt, else go to step ii.

END OF ENUMERATION

Initially $|J| = m$. Whenever a possibility is enumerated, an element is taken from J; hence at most m possibilities are enumerated. We show that $F_m^A(x_1, \ldots, x_m)$ is one of them.

Assume, by way of contradiction, that $\tau_0 = F_m^A(x_1, \ldots, x_m)$ is never enumerated. Hence throughout the enumeration $\tau_0 \in P$. There are two cases.

Case 1: Fewer than m possibilities are enumerated. Hence there is some iteration where step iii begins but never terminates. Let $P, J, g, \{f_\tau\}_{\tau \in P}$

denote the values of these variables during this iteration. Since step iii never terminates the functions $\{f_\tau\}_{\tau\in P}$ are total. Since $\tau_0 \in P$, f_{τ_0} is constructed and is total. Since $F_m^A(x_1,\ldots,x_m) = \tau_0$, by stage 0 of the construction $f_{\tau_0} \in S_m^A$. Since g is a subfunction of f_{τ_0} none of the IIMs in $\{M_j\}_{j\in\{1,\ldots,m\}-J}$ infers f_{τ_0}. By the construction none of the IIMs in $\{M_j\}_{j\in J}$ infers f_{τ_0}. Hence none of $M_1,\ldots,M_m$ infers f_{τ_0}. This contradicts that $S_m^A \in [1,m]BC$ via $M_1,\ldots,M_m$.

Case 2: Exactly m possibilities are enumerated. Hence the enumeration halts with $\tau_0 \in P$. The value of g at the end of the enumeration is such that any extension of g is not inferred by any of $M_1,\ldots,M_m$. Since $\tau_0 \in P$, g is undefined on almost all elements of $\{\langle NUM(\tau_0),k\rangle : k \in \mathbb{N}\}$. By the recursion theorem there exists e such that the following function has index e.

$$g'(x) = \begin{cases} g(x) & \text{if } x \in dom(g); \\ e & \text{if } x \notin dom(g) \text{ and } x = \langle NUM(\tau_0),k\rangle; \\ 0 & \text{otherwise.} \end{cases}$$

Since $g'(0) = \langle 0,1,x_1,\ldots,x_m\rangle$, $F_m^A(x_1,\ldots,x_m) = NUM(\tau_0)$, and for almost all k, $g'(\langle NUM(\tau_0),k\rangle) = e$, which is an index for g', $g' \in S_m^A$. Since g' is an extension of g, g' is not inferred by any of $M_1,\ldots,M_m$. This contradicts that $S_m^A \in [1,m]BC$ via $M_1,\ldots,M_m$. ∎

Proof: (of Theorem 4.8.)

By Lemma 4.9 $i \Rightarrow iv$ and $i \Rightarrow vii$. Clearly $iv \Rightarrow iii \Rightarrow ii$, and $vii \Rightarrow vi \Rightarrow v$. We need only show $ii \Rightarrow i$ and $v \Rightarrow i$.

$ii \Rightarrow i$: Assume $(\exists n)[EX[A[n]] = EX[A[n+1]]$. By Lemma 4.13.$iv$, for all m', $S_{m'}^A \in [1,2^n]EX$. In particular $S_{2^n}^A \in [1,2^n]EX \subseteq [1,2^n]BC$. By Lemma 4.14 with $m = 2^n$ we obtain $A \leq_{\mathrm{T}} K$.

The proof for $v \Rightarrow i$ is similar but uses Lemma 4.13.v. ∎

From Lemma 4.14 we can obtain a result of Smith [Smi82]. This is not a new proof since we used his techniques.

Corollary 4.15 *$REC \notin [1,m]BC$.*

Proof:

Let A be such that $A \not\leq_{\mathrm{T}} K$. By Lemma 4.14 $S_1^A \notin [1,m]BC$. Hence $REC \notin [1,m]BC$. ∎

Note 4.16 Part of Theorem 4.8 can be restated as $A \not\leq_{\mathrm{T}} K \Rightarrow EX[A[n]] \subset EX[A[n+1]]$. Using Note 4.7.$ii$, and using $TREE_n^A$ instead of F_n^A, the proof of Theorem 4.8 can be modified to obtain $A \not\leq_{\mathrm{T}} K \Rightarrow (\forall B)[EX[A[n+1]] \not\subseteq EX[B[n]]]$.

4.2 $EX[A]$ and $BC[A]$

We show (1) if $EX[A] = EX$ or $BC[A] = BC$ then A is low, and (2) if $A \leq_{\mathrm{T}} K$ and $A \equiv_{\mathrm{T}} G$ where G is 1-generic then $EX[A] = EX$ and $BC[A] = BC$. Slaman and Solovay [SS91] (see also [KS93b]) have shown (3) $EX[A] = EX \Rightarrow \mathcal{G}(A)$ (see Notation 3.1 for what $\mathcal{G}(A)$ means). A modification of their proof (or a direct proof from [KS93b]) yields (4) $BC[A] = BC \Rightarrow \mathcal{G}(A)$. Although (1) is superseded by (3),(4) we include the proof of (1) since it is much simpler than the proof of (3),(4).

Theorem 4.17 *If $EX = EX[A]$ or $BC = BC[A]$, then A is low.*

Proof:

Assume $EX = EX[A]$. Consider the set $S_1^{A'}$ from Definition 4.12. Note that $S_1^{A'} \in EX[A]$ (use the Limit Lemma [Soa87, p. 57] to approximate A'). Since $EX[A] = EX$ we have $S_1^{A'} \in EX \subseteq BC$. By Lemma 4.14 $A' \leq_{\mathrm{T}} K$.

The proof for $BC = BC[A]$ is similar. ∎

We now show that there exist nonrecursive sets A such that $EX[A] = EX$. By Lemma 4.9 if $A \leq_{\mathrm{T}} K$ then $EX[A*] = EX$ and $BC[A*] = BC$. Hence, we seek a set $A \leq_{\mathrm{T}} K$ such that $EX[A] = EX[A*]$. It turns out that 1-generic sets suffice. This is not surprising since 1-generic sets force statements to be true with only finite information.

We include a definition of genericity for completeness. For more information on genericity see [Joc80].

Definition 4.18 A set G is *i-generic* if for every Σ_i set W (of elements of $\{0,1\}^*$) either
$(\exists \sigma \preceq G)[\sigma \in W]$, in which case we say that G *meets* W, or
$(\exists \sigma \preceq G)[(\forall \tau \succeq \sigma)[\tau \notin W]]$, in which case we say G *strongly avoids* W.

Lemma 4.19 *If $A \equiv_{\mathrm{T}} G$ where G is 1-generic, then $EX[A*] = EX[A]$.*

Proof:

Let $S \in EX[A]$. Since $A \equiv_{\mathrm{T}} G$, $S \in EX[G]$. Assume $S \in EX[G]$ via M^G. We describe an $EX[G*]$ inference procedure for S. We describe it as a machine that *requests* values of f (rather than receiving them) and outputs an infinite stream of guesses for indices. This is clearly equivalent to the usual definition of an OIIM.

INFERENCE-ALGORITHM

To infer $f \in S$, we do the following. Initialize i to 0.

1) Compute $e = M^G(f(0), \ldots, f(i))$. Let σ denote the shortest initial segment of G of length greater than i which contains all the bits used in the computation.

2) Dovetail the following two procedures.

   a) Search for $j \geq i$ and $\tau \in \{0,1\}^*$ such that $M^{\sigma\tau}(f(0), \ldots, f(j)) \downarrow \neq e$. If such a j, τ are found then set i to $i+1$ and go to step 1

   b) Continue to output e as the guess for an index for f.

END of INFERENCE-ALGORITHM

Each pass through steps 1,2 with a new value of i is called an *iteration.*

We first show that if an iteration never terminates then the index e output in step $2.b$ is an index for f. Since step $2.a$ never finds a j, τ, and every τ is tried (including those that are initial segments of G), we must have $(\forall j \geq i)[M^G(f(0), \ldots, f(j)) = e]$. Hence, e must be an index for f.

We now show that there is an iteration that never terminates. Let $e_0, i_0 \in \mathbb{N}$, $\sigma \in \{0,1\}^*$, and $W \subseteq \{0,1\}^*$ be such that the following hold.

- $(\forall j \geq i_0)[M^G(f(0), \ldots, f(j)) = e_0]$.
- σ is the initial segment of G used in the computation of $M^G(f(0), \ldots, f(i_0))$.
- $W = \{\sigma\tau \in \{0,1\}^* : (\exists j \geq i_0)[M^{\sigma\tau}(f(0), \ldots, f(j)) \neq e_0]\}$.

Since M^G infers f, the values of e_0, i_0, and σ exist. Since f is recursive, W is r.e. Assume, by way of contradiction, that every iteration of the inference procedure above terminates. Then every initial segment of G can be extended to meet W (use that the length of σ is chosen greater than i during iteration i). Hence, G cannot strongly avoid W. Since G is 1-generic, G must meet W; hence, there is an initial segment of G in W. This contradicts the definition of e_0, i_0.

Hence we have that $S \in EX[G*]$. Since $A \equiv_{\mathrm{T}} G$, $S \in EX[A*]$. ∎

We now prove a similar lemma for BC.

Lemma 4.20 *If $A \equiv_{\mathrm{T}} G$ where G is a 1-generic set, then $BC[A] = BC[A*]$.*

Proof:

Let $S \in BC[A]$. Since $A \equiv_{\mathrm{T}} G$, $S \in BC[G]$. Assume $S \in BC[G]$ via M^G. We describe a $BC[G*]$ inference procedure for S. We use the same convention for describing OIIMs as in Lemma 4.19.

INFERENCE-ALGORITHM

To infer $f \in S$, we do the following. Initialize i to 0.

1) Compute $e = M^G(f(0), \ldots, f(i))$. Let $\sigma \preceq G$ be the shortest initial segment of G of length greater than i that contains answers to all queries made. Let I be the singleton set just containing e.

2) Dovetail the following two procedures.

   a) Search for $j \geq i$, $\tau \in \{0,1\}^*$, $e', s, x \in \mathbb{N}$ such that we have $e' = M^{\sigma\tau}(f(0), f(1), \ldots, f(j))$ and $\varphi_{e',s}(x) \downarrow \neq f(x)$. If such a j, τ, e', s, x are found then set i to $i+1$ and go to step 1.

   b) Let $\tau_0, \tau_1, \tau_2, \ldots$ be the set of all strings that extend σ. Let $m_0, m_1, m_2, \ldots$ be the set of all numbers $\geq i$.

   For $k := 0$ to ∞ let $k = \langle i, j \rangle$ and do the following:

      i) Compute $e_{i,j} = M^{\tau_i}(f(0), \ldots, f(m_j))$ (if a number that is not in $dom(\tau_i)$ is queried then set $e_{i,j}$ to an index for the empty function).

      ii) Let I be $I \cup \{e_{i,j}\}$. Output $AM(I)$ as a conjecture for what f does. (See Definition 2.32 for the definition of AM.)

END of INFERENCE-ALGORITHM

Every pass through steps 1,2 with a new value of i is called an *iteration.*

We first show that if an iteration never terminates then the algorithm BC-infers f. Assume that some iteration never terminates. Since M^G infers f, sometime during the non-terminating iteration, an $e_{i,j}$ is produced that is the correct index for f. We claim that once that index enters I, $AM(I)$ will always compute f correctly. If it does not then some other index in I is converging and disagreeing with f. But if this happens then the iteration will terminate by part $2a$.

We now show that there is an iteration that never terminates. Let $i_0 \in \mathbb{N}$, $\sigma \in \{0,1\}^*$ be such that the following hold:

- $(\forall j \geq i_0)[\varphi_{M^G(f(0),\ldots,f(j))} = f]$.
- σ is the initial segment of G used in the computation of $M^G(f(0),\ldots,f(i_0))$.
- $W = \{\sigma\tau \in \{0,1\}^* : (\exists j \geq i_0)(\exists s,x)[\varphi_{M^{\sigma\tau}(f(0),\ldots,f(j)),s}(x) \downarrow \neq f(x)]\}$.

From this point on the proof is similar to that of Lemma 4.19. ▮

Theorem 4.21 *If $\mathcal{G}(A)$ then $EX[A] = EX$ and $BC[A] = BC$.*

Proof:
If A is recursive then clearly $EX[A] = EX$ and $BC[A] = BC$. Otherwise, by $\mathcal{G}(A)$, there exists a 1-generic set G such that $A \equiv_{\mathrm{T}} G \leq_{\mathrm{T}} K$.

By Lemmas 4.19 and 4.20, $EX[A*] = EX[A]$ and $BC[A*] = BC[A]$. Since $A \leq_{\mathrm{T}} K$, by Lemma 4.9, $EX[A*] = EX$ and $BC[A*] = BC$. Hence, $EX[A] = EX[A*] = EX$ and $BC[A] = BC[A*] = BC$. ▮

5 When are Degrees Omniscient?

Adleman and Blum completely characterized the EX-omniscient sets (see also [KS93b]). We state their result so we can refer to it later.

Theorem 5.1 (Theorem 7 of [AB91]) *$REC \in EX[A]$ iff A is high.*

5.1 $EX[A[m]]$ and $BC[A[m]]$

Theorem 5.2 *For all m, A, $REC \notin BC[A[m]]$ and $REC \notin EX[A[m]]$.*

Proof:

By Lemma 4.11, for any oracle A, $BC[A[m]] \subseteq [1, 2^m]BC$. By Corollary 4.15 (or [Smi82]) $REC \notin [1, 2^m]BC$. Hence, $REC \notin BC[A[m]]$. Since $EX[A[m]] \subseteq BC[A[m]]$, $REC \notin EX[A[m]]$. ∎

5.2 $EX[A*]$ and $BC[A*]$

In this subsection we solve the problem of exactly when $REC \in EX[A*]$ and exactly when $REC \in BC[A*]$.

We use the following lemmas. The first was proven by Jockusch, the second is the (relativized) Friedberg Completeness Criterion, and the third is an easy relativization of Lemma 4.9.

Lemma 5.3 ([Joc72], Theorem 9) *For any A the following are equivalent.*

i. There exists $h \leq_{\mathrm{T}} A$ such that $REC = \{\varphi_{h(i)}\}_{i \in \mathbb{N}}$.

ii. $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

Lemma 5.4 (See [Soa87], p. 97) *The following are true.*

i. If $K \leq_{\mathrm{T}} Z$ then there exists G such that $G' \equiv_{\mathrm{T}} Z$. Moreover, G can be taken to be 1-generic (and in particular $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} Z$).

ii. Let $j \geq 1$. If $\emptyset^{(j)} \leq_{\mathrm{T}} Z$ then there exists G such that $\emptyset^{(j-1)} \leq_{\mathrm{T}} G$ and $G' \equiv_{\mathrm{T}} Z$. Moreover, G can be taken to be j-generic (and in particular $G' \equiv_{\mathrm{T}} G \oplus \emptyset^{(j)} \equiv_{\mathrm{T}} Z$).

Lemma 5.5 *If $A \leq_{\mathrm{T}} B'$, then $EX[A*] \subseteq EX[B]$ and $BC[A*] \subseteq BC[B]$.*

Lemma 5.6 *For any A there is a 1-generic G such that $BC[G*] = BC[G] = BC[A*]$ and $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$.*

Proof:

By applying Lemma 5.4.i to $Z = A \oplus K$ we obtain a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$. Since G is 1-generic, by Lemma 4.20, $BC[G*] = BC[G]$. Since $A \leq_{\mathrm{T}} G'$, by Lemma 5.5, $BC[A*] \subseteq BC[G] = BC[G*]$. Since $G \leq_{\mathrm{T}} A \oplus K$, $BC[G*] \subseteq BC[(A \oplus K)*] \subseteq BC[A*]$ (the last inclusion is obtained by Note 4.10). Hence $BC[G*] = BC[G] = BC[A*]$ as desired. ∎

Theorem 5.7 *$REC \in EX[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

Proof:

($\Rightarrow$): Assume $REC \in EX[A*]$. By Lemma 5.4, applied to $A \oplus K$, there is a set B such that $B' \equiv_{\mathrm{T}} A \oplus K$. Since $A \leq_{\mathrm{T}} B'$, by Lemma 5.5 $EX[A*] \subseteq EX[B]$, so $REC \in EX[B]$. By Theorem 5.1 B is high, hence $\emptyset'' \leq_{\mathrm{T}} B' \equiv_{\mathrm{T}} A \oplus K$.

($\Leftarrow$): Assume $\emptyset'' \leq_{\mathrm{T}} A \oplus K$. By Lemma 5.3 there exists $h \leq_{\mathrm{T}} A$ such that $REC = \{\varphi_{h(i)}\}_{i \in \mathbb{N}}$. We define an OIIM (that uses oracle A) as follows: Upon receiving the first s input values of f, output $h(i)$ for the smallest i such that $\varphi_{h(i)}$ agrees with f on the given values. Clearly, for any input $f \in REC$ the machine computes h only on finitely many arguments, which requires only finitely many queries to A. Thus $REC \in EX[A*]$. ∎

Theorem 5.8 *$REC \in BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

Proof:

By Theorem 5.7 $\emptyset'' \leq_{\mathrm{T}} A \oplus K$ implies $REC \in EX[A*] \subseteq BC[A*]$. Hence we need only prove the other direction.

Assume $REC \in BC[A*]$. Then by Lemma 5.6, there is a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$ and $BC[G*] = BC[A*]$. Since $REC \in BC[A*]$, $REC \in BC[G*]$. Let $REC \in BC[G*]$ via M^G.

We will show that G is high by constructing a G-recursive function g that dominates all recursive functions. After we show that G is high we will have $\emptyset'' \leq_{\mathrm{T}} G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$, as desired. (We will not be using the fact that G is 1-generic.)

First we define two functions independent of G. Second we use these functions to construct a G-recursive function g. Third we prove that g dominates all recursive functions. We split it up this way so that we can use the functions constructed in the first part in both the second and third parts.

First part: We define recursive functions ψ and T where ψ maps $\Sigma^* \times \mathbb{N}$ to Σ^*, and T maps $\Sigma^* \times \mathbb{N}$ to $\mathcal{P}^{fin}(\mathbb{N})$ (the set of finite subsets of $\mathbb{N}$). We will have

$$\begin{aligned} \sigma = &\ \psi(\sigma,0) \preceq \psi(\sigma,1) \preceq \cdots \\ \{D : D \subseteq \{0,\ldots,|\sigma|\}\} = &\ T(\sigma,0) \supseteq T(\sigma,1) \supseteq \cdots \end{aligned}$$

The idea is that we are (at first) looking for an extension τ of σ and a subset D of $\{0,1,\ldots,|\sigma|\}$ such that $\varphi_{M^D(\tau)}$ is wrong on $|\tau|$ (or asks a question $> |\sigma|$). The second parameter bounds how long we can search for such an extension. ψ will be larger and larger extensions of σ; T will be a shrinking collection of possible finite oracles. We try to make more and more of the oracles in T yield incorrect guesses.

Let $\cdot$ denote the concatenation of strings. We define ψ and T inductively. For the base case we define

$$\begin{aligned} \psi(\sigma,0) = &\ \sigma, \\ T(\sigma,0) = &\ \{D : D \subseteq \{0,\ldots,|\sigma|\}\}. \end{aligned}$$

Assume $\psi(\sigma,t)$ and $T(\sigma,t)$ have been defined. If there exists some $D \in T(\sigma,t)$, some string τ with $|\tau| < t$ and some $b \in \{0,1\}$ such that

$$\begin{gathered} \varphi_{M^D(\psi(\sigma,t)\cdot\tau)}(|\psi(\sigma,t)\cdot\tau|) \downarrow \neq b \text{ within } t \text{ steps} \\ \text{or } M^D(\psi(\sigma,t)\cdot\tau) \text{ queries some } x > |\sigma| \end{gathered}$$

then take the first such (D,τ,b) and let

$$\begin{aligned} \psi(\sigma,t+1) = &\ \psi(\sigma,t)\cdot\tau\cdot b, \\ T(\sigma,t+1) = &\ T(\sigma,t) - \{D\} \end{aligned}$$

else ψ, T remain unchanged ($\psi(\sigma,t+1) = \psi(\sigma,t)$, $T(\sigma,t+1) = T(\sigma,t)$).

Note 5.9 There are at most $2^{|\sigma|+1}+1$ distinct values of $\psi(\sigma,t)$. This is because every time $\psi(\sigma,t)$ changes a set is removed from $T(\sigma,t)$, and there are $2^{|\sigma|+1}$ elements in $T(\sigma,0)$.

Second part: There is a G-recursive function g given by:

$$\begin{aligned} \hat{g}(\sigma) &= (\mu t > |\sigma|)\ [\ G \upharpoonright |\sigma| \notin T(\sigma,t)] \\ g(n) &= \max_{|\sigma|\leq n}\{\hat{g}(\sigma)\} \end{aligned}$$

Note 5.10 The intuition behind $\hat{g}(\sigma)$ is that, given σ, and noting that $\psi(\sigma,0) \preceq \psi(\sigma,1)\cdots$, we are looking for the least t such that the extension $\psi(\sigma,t)$ might fool M^G. In particular note that either $M^G(\psi(\sigma,\hat{g}(\sigma))$ makes a query larger than $|\sigma|$ or $\varphi_{M^G(\psi(\sigma,\hat{g}(\sigma)))}$ is a function that does not have initial segment $\psi(\sigma,\hat{g}(\sigma))$.

We show that $\hat{g}$ is total (given this, clearly g is total and $g \leq_{\mathrm{T}} G$). Let t_1 be the least number $> |\sigma|$ such that

$$(\forall s \geq t_1)[\psi(\sigma,t_1) = \psi(\sigma,s) \text{ and } T(\sigma,t_1) = T(\sigma,s)].$$

We claim $G \restriction |\sigma| \notin T(\sigma,t_1)$, so $\hat{g}(\sigma) \leq t_1$ and exists. Assume, by way of contradiction, that $G \restriction |\sigma| \in T(\sigma,t_1)$. Let t_2 be the minimum number such that $t_1 \leq t_2$ and there exists $t_3 < t_2$ such that one of the following occurs.

i. The $M^G(\psi(\sigma,t_2)\cdot 0^{t_3})$ computation never makes a query $> |\sigma|$ and $\varphi_{M^G(\psi(\sigma,t_2)\cdot 0^{t_3})}(|\psi(\sigma,t_2)\cdot 0^{t_3}|) \downarrow \neq 1$ within t_2 steps.

ii. $M^G(\psi(\sigma,t_2)\cdot 0^{t_3})$ queries some $x > |\sigma|$.

Such t_2 exists since M^G infers $\psi(\sigma,t_2)\cdot 0^{\omega}$. The triple $(G \restriction |\sigma|, 0^{t_3}, 1)$ will be noted as a candidate for (D,τ,b) when defining $\psi(\sigma,t_2+1)$. Hence $\psi(\sigma,t_2+1) \neq \psi(\sigma,t_1)$. But this contradicts the choice of t_1.

Third part: Assume, by way of contradiction, that g does not dominate all recursive functions. Then there exists a recursive increasing h such that $(\overset{\infty}{\exists} n)[g(n) < h(n)]$. For given σ, let

$$h'(\sigma) = (\mu t \geq |\sigma|)\,[\psi(\sigma,t) = \psi(\sigma,h(t)) \wedge t > |\psi(\sigma,t)|\,].$$

h' is total by Note 5.9. Clearly h' is recursive. We define a sequence of strings inductively. Let $\sigma_0 = 0$, and let $\sigma_{n+1} = \psi(\sigma_n, h'(\sigma_n))\cdot 0$. Let f be defined as the limit of the sequences. Note that f is recursive.

Note 5.11 The intuition behind f is as follows. f is the limit of the sequence $\sigma_0 \preceq \sigma_1 \cdots$. We are hoping to fool M^G each time we extend σ_i to σ_{i+1}. That is, we are hoping that, for each i, either $M^G(\sigma_i)$ makes queries larger than $|\sigma_i|$ or $\varphi_{M^G(\sigma_i)}$ is a function that does not have initial segment σ_i. If we could use g instead of h this hope would be a reality; however, by using h, we are at least guaranteed to fool M^G infinitely often.

Since f is recursive, f is $BC[G*]$-inferred by M^G. Therefore there is a number m such that for all σ, $\sigma_m \preceq \sigma \preceq f$:

i. $\varphi_{M^G(\sigma)} = f$.

ii. $M^G(\sigma)$ queries no element greater than $|\sigma_m|$.

Since $(\overset{\infty}{\exists} n)[g(n) < h(n)]$, there is some $n > |\sigma_{m+1}|$ such that $g(n) < h(n)$. Let k be the greatest number such that $|\sigma_k| \leq n$. We have

$$|\sigma_m| < |\sigma_k| \leq n < |\sigma_{k+1}|.$$

Let $t = h'(\sigma_k)$. By the definition of h' we have $\psi(\sigma_k, t) = \psi(\sigma_k, h(t))$ and $t > |\psi(\sigma_k, t)|$. By the definition of σ_{k+1} we have $\sigma_{k+1} = \psi(\sigma_k, t) \cdot 0$, hence $n < |\sigma_{k+1}| = |\psi(\sigma_k, t)| + 1 \leq t$. Since h is increasing we know that $h(n) \leq h(t)$. We now have

$$\psi(\sigma_k, \hat{g}(\sigma_k)) \preceq \psi(\sigma_k, g(n)) \preceq \psi(\sigma_k, h(n)) \preceq \psi(\sigma_k, h(t)) = \psi(\sigma_k, t) \prec \sigma_{k+1}.$$

By the construction of $\hat{g}$, $G \upharpoonright |\sigma_k| \notin T(\sigma_k, \hat{g}(\sigma_k))$. Thus there is some string σ, $\sigma_k \preceq \sigma \prec \sigma_{k+1}$, such that one of the following occurs.

- The $M^G(\sigma)$ computation makes no query $> |\sigma_k|$ and $\varphi_{M^G(\sigma)}(|\sigma|) \downarrow \neq f(|\sigma|)$, which contradicts condition i.
- $M^G(\sigma)$ queries some $x > |\sigma_k| > |\sigma_m|$, which contradicts condition ii.

∎

5.3 $EX[A]$ and $BC[A]$

Adleman and Blum [AB91] showed that $REC \in EX[A]$ iff A is high. It is an open question as to when $REC \in BC[A]$. The rest of this subsection will provide evidence that the question of when $REC \in BC[A]$ does not have a nice answer. We show (1) for A r.e., $REC \in BC[A]$ iff A is high, (2) there exists a low set A such that $REC \in BC[A]$, and (3) the statement 'if A is high$_2$ then $REC \in BC[A]$' is false in a strong way: there are sets A of arbitrarily high double jumps such that $REC \notin BC[A]$.

5.3.1 The r.e. case

Notation 5.12 $u(A; e, x, s)$ is the maximum element of the oracle that $M_{e,s}^A(x)$ queries. u is referred to as the *use function*. We will be dealing with a fixed categorically total oracle Turing machine $M^{()}$, so we leave out the index e and the time s. The resulting notation is $u(A; \sigma)$.

Note 5.13 If A is r.e. and $A_s \upharpoonright x = A \upharpoonright x$ then for all $t \geq s$ $A_t \upharpoonright x = A \upharpoonright x$. In particular if $A_s \upharpoonright u(A; \sigma) = A \upharpoonright u(A; \sigma)$ then for all $t \geq s$ the computation $M^{A_t}(\sigma)$ is identical to that of $M^A(\sigma)$.

Definition 5.14 Let $\sigma, \tau \in \{0,1\}^*$. Then $\sigma < \tau$ means that either $|\sigma| < |\tau|$, or $|\sigma| = |\tau|$ and σ is lexicographically less than τ.

Theorem 5.15 *Let A be r.e. $REC \in BC[A]$ iff A is high.*

Proof:

The reverse direction is easy: A high $\Rightarrow REC \in EX[A]$ (by Theorem 5.1) and $EX[A] \subseteq BC[A]$ trivially.

For the forward direction, assume, by way of contradiction, that A is not high (i.e., $\emptyset'' \not\leq_{\mathrm{T}} A'$) and $REC \in BC[A]$ via M^A. Since $\emptyset'' \not\leq_{\mathrm{T}} A'$, for all $g \leq_{\mathrm{T}} A$, there exists a recursive h such that $\overset{\infty}{\exists} x[g(x) < h(x)]$ (see [Soa87, p. 208]). We will define a particular $g \leq_{\mathrm{T}} A$ and use the corresponding h to build an $f \in REC$ such that M^A does not BC-infer f.

We try to imitate the standard construction of a recursive function that is not BC-identified by a (non oracle) IIM. Since we have an OIIM, a direct imitation is impossible. We can use an approximation to A, but we need that the approximation is valid infinitely often. To help achieve this we define an auxiliary function $\hat{g} \leq_{\mathrm{T}} A$ from $\{0,1\}^*$ to $\mathbb{N}$. Intuitively $\hat{g}(\sigma)$ tells us how good an approximation to A we need to look at to find an extension $\sigma' \succeq \sigma$ such that $\varphi_{M^A(\sigma')}(|\sigma'|)\downarrow$. This is useful in trying to build a function f not inferred by M^A since, as in the standard construction of a recursive function not BC-identified, a convergence is a chance to diagonalize.

ALGORITHM for $\hat{g}$

1) Input(σ).

2) Look for a σ', s' such that $\sigma \preceq \sigma'$ and $\varphi_{M^A(\sigma'),s'}(|\sigma'|)\downarrow$.

3) Let $s'' = \mu s[(\forall \sigma'' \leq \sigma')[A_s \upharpoonright u(A;\sigma'') = A \upharpoonright u(A;\sigma'')]]$. (By Note 5.13 $(\forall s \geq s'')[A_s \upharpoonright u(A;\sigma'') = A \upharpoonright u(A;\sigma'')]$.)

4) Output $\max\{s', s'', |\sigma'|\}$.

END OF ALGORITHM

We show that $\hat{g}$ is total by showing that, during step 2 of the execution of the algorithm on input σ, an appropriate σ' and s' are found. Since M^A BC-identifies REC, M^A BC-identifies the function $\sigma 0^\omega$. Let $\sigma^\triangleleft$ be such that $\sigma \preceq \sigma^\triangleleft \preceq \sigma 0^\omega$ and $M^A(\sigma^\triangleleft)$ is an index for the function $\sigma 0^\omega$. Since $\varphi_{M^A(\sigma^\triangleleft)}(|\sigma^\triangleleft|)\downarrow$, there exists $s^\triangleleft$ such that $\varphi_{M^A(\sigma^\triangleleft),s^\triangleleft}(|\sigma^\triangleleft|)\downarrow$. Since $\sigma^\triangleleft, s^\triangleleft$ satisfy the conditions for σ', s', some σ', s' will always be found.

Let g be defined by $g(n) = \max_{|\sigma|=n} \hat{g}(\sigma)$. It is easy to see that $g \leq_{\mathrm{T}} A$ and g is total. Let h be a recursive function such that $(\overset{\infty}{\exists} n)[g(n) < h(n)]$. We can take h to be increasing.

At the end of stage n of the construction we will have σ_n, an initial segment of the function f. During stage $n+1$ we will want to extend σ_n to some σ^+, in the hope of making $\varphi_{M^A(\sigma^+)}(|\sigma^+|) \neq f(|\sigma^+|)$. In the search for such an extension we can only use an approximation to A. If we are considering using σ^+ for our extension, we will use the approximation $A_{h(|\sigma^+|)}$.

In the construction below the term 'least' when applied to strings is relative to the ordering of Definition 5.14.

CONSTRUCTION

Stage 0: $\sigma_0 = \lambda$ (the empty string).

Stage n+1: Look for the least σ^+ such that there exist b, t with $\sigma_n \preceq \sigma^+$, $|\sigma^+| \leq t$, $b \in \{0,1\}$, and $\varphi_{M^{A_t}(\sigma^+),t}(|\sigma^+|) \downarrow\neq b$, where $t = h(|\sigma^+|)$. Let $\sigma_{n+1} = \sigma^+ b$.

END OF CONSTRUCTION

We show that every stage terminates, hence for all n, σ_n exists. Thus the function $f = \bigcup_n \sigma_n$ is total recursive. We then show that M^A does not BC-infer f.

1) Every stage terminates. We need to show that the search in stage $n+1$ terminates. Let l be minimal such that $l \geq |\sigma_n|$ and $g(l) < h(l)$.

Let $\sigma = \sigma_n 0^{l-|\sigma_n|}$. By the definition of g, there are σ', s' such that $\sigma \preceq \sigma'$, $\varphi_{M^A(\sigma'),s'}(|\sigma'|)\downarrow$, and $A_{g(l)} \upharpoonright u(A;\sigma') = A \upharpoonright u(A;\sigma')$. As $h(|\sigma'|) \geq h(l) > g(l)$ it follows that $\sigma^+ := \sigma'$ and $b := 1 \dot{-} \varphi_{M^A(\sigma'),s'}(|\sigma'|)$ satisfy the condition in stage $n+1$. By Note 5.13, $A_{h(|\sigma'|)} \upharpoonright u(A;\sigma') = A \upharpoonright u(A;\sigma')$.

2) M^A does not infer f: Suppose that l, n and σ^+ satisfy the following conditions: $|\sigma_n| \leq l < |\sigma_{n+1}|$, $\sigma_{n+1} = \sigma^+ b$, $g(l) < h(l)$. Then $\varphi_{M^A(\sigma^+)}(|\sigma^+|) \neq f(|\sigma^+|)$, as we will now show: Let σ denote the initial segment of σ^+ of length l, and let σ' be the string found in step 2 of the computation of $\hat{g}(\sigma)$. Note that $\hat{g}(\sigma) \leq g(l) < h(l) \leq h(|\sigma^+|)$. Since σ^+ is chosen to be the least extension that works, $\sigma^+ \leq \sigma'$. By Note 5.13, $(\forall t \geq \hat{g}(\sigma))[A_t \upharpoonright u(A;\sigma^+) = A \upharpoonright u(A;\sigma^+)]$. It follows that $\varphi_{M^A(\sigma^+)}(|\sigma^+|) \neq b = f(|\sigma^+|)$.

As there are infinitely many l as above, we get that M^A does not BC-infer f. ∎

Corollary 5.16 *The following statement is false: if A is* high$_2$ *then $REC \in BC[A]$.*

Proof:

Let A be r.e. and high$_2$ but not high (such an A exists— see [Soa87, p. 140]). By Theorem 5.15 $REC \notin BC[A]$. ∎

5.3.2 A Low A such that $REC \in BC[A]$

In this section we show that there is a low set A such that $REC \in BC[A]$.

Definition 5.17 Let f and g be partial functions. f is *compatible with* g iff $(\forall x \in \mathrm{dom}(f) \cap \mathrm{dom}(g))[f(x) = g(x)]$. f is *incompatible with* g otherwise. f *extends* g ($g \subseteq f$), iff f is compatible with g and $\mathrm{dom}(g) \subseteq \mathrm{dom}(f)$.

Theorem 5.18 *There is a low set A such that $REC \in BC[A]$.*

Proof:

We will obtain a low set A and present an A-recursive algorithm that $BC[A]$-infers REC. The idea behind the algorithm is that, at stage s, we will amalgamate some subset of (modified versions of) the functions $\{\varphi_{c(s)}, \varphi_{c(s)+1}, \ldots, \varphi_s\}$. We control $c(s)$. We think of $\varphi_{c(s)}$ as being correct (or at least not in contradiction to f) and hence we only use partial recursive functions that *seem* to be compatible with it. We may also change our value of $c(s)$ if either a guess made a while back based on this value is seen to contradict f, or if some function φ_j that is seen to be extended by f *seems* to not be extended by $\varphi_{c(s)}$. Both of these conditions are seen as evidence that the current value of $c(s)$ leads to guesses that are incorrect. In either case we increment $c(s)$ by one.

The problem is how to get information about what *seems* to be an extension. In order to make it possible to *know* that two functions are compatible we deal with functions that are modified. Let $u(j,a)$ be the total recursive function such that

$$\varphi_{u(j,a)}(x) = \begin{cases} \varphi_j(x) & \text{if } (\exists s)[\varphi_{j,s}(x)\downarrow \text{ and } a \notin K_{s+x}]; \\ \uparrow & \text{otherwise.} \end{cases}$$

If $a \notin K$ then $\varphi_{u(j,a)} = \varphi_j$. If $a \in K$ and s is the least number such that $a \in K_s$ then $dom(\varphi_{u(j,a)}) \subseteq \{0,\ldots,s-1\}$, $\varphi_{u(j,a)} \subseteq \varphi_{j,s}$, and $\varphi_{u(j,a)}$ can be completely determined; hence in this case testing whether (say) φ_i is an extension of $\varphi_{u(j,a)}$ is an r.e. procedure.

Definition 5.19 Let $a,i,j \in \mathbb{N}$. If there exist x,s such that $\varphi_{i,s}(x)\downarrow \neq \varphi_{u(j,a),s}(x)\downarrow$ then φ_i *is seen to be incompatible with* $\varphi_{u(j,a)}$. If there exist s,t such that $a \in K_s$ and $\varphi_{i,t}$ is an extension of $\varphi_{u(j,a)}$ (which can be completely determined) then φ_i *is seen to be an extension of* $\varphi_{u(j,a)}$. Note that it is not possible for both of these to occur, though it is possible for neither to occur.

We now present a partial recursive function whose intention (not always achieved) is to tell whether φ_i is an extension of $\varphi_{u(j,a)}$ or if φ_i is incompatible with $\varphi_{u(j,a)}$.

$$\gamma(i,j,a) = \begin{cases} EXT & \text{if } \varphi_i \text{ is seen to be an extension of } \varphi_{u(j,a)}; \\ INCOMP & \text{if } \varphi_i \text{ is seen to be incompatible with } \varphi_{u(j,a)}; \\ \uparrow & \text{otherwise.} \end{cases}$$

Note that the following hold.

i. $\gamma(i,j,a) = INCOMP$ iff φ_i is incompatible with $\varphi_{u(j,a)}$.

ii. $\gamma(i,j,a) = EXT$ implies φ_i is compatible with $\varphi_{u(j,a)}$.

iii. If $a \in K$ and $\gamma(i,j,a) \neq EXT$ then φ_i does not extend $\varphi_{u(j,a)}$.

By the Low Basis Theorem (see [Soa87, p. 109]) there is a low set A such that γ can be extended to an A-recursive total function g with range $\{EXT, INCOMP\}$. We relabel the outputs so that they are now $\{COMP, INCOMP\}$ (Fact 5.20 will make clear why we relabel it as such). This is the desired set A. We will use A as an oracle to compute g, which will yield information about functions being compatible.

We will use the following easily verified facts about g. They all clarify in what ways we can use g to test whether φ_i is compatible with $\varphi_{u(j,a)}$.

Fact 5.20 *The following hold.*

i. If $g(i,j,a) = INCOMP$ then $\gamma(i,j,a) \neq EXT$, which does not yield any information. If in addition $a \in K$ then φ_i does not extend $\varphi_{u(j,a)}$.

ii. If $g(i,j,a) = COMP$ then $\gamma(i,j,a) \neq INCOMP$ hence $\varphi_{u(j,a)}$ and φ_i are compatible.

iii. Let $i, j_1, \ldots, j_n, a_1, \ldots, a_n \in \mathbb{N}$. Assume that $g(i,j_1,a_1) = g(i,j_2,a_2) = \cdots = g(i,j_n,a_n) = COMP$. Then $\varphi_{AM(u(j_1,a_1),\ldots,u(j_n,a_n))}$ is compatible with φ_i. In particular, if φ_i is total, then $\varphi_{AM(u(j_1,a_1),\ldots,u(j_n,a_n))} \subseteq \varphi_i$.

iv. Let $i, j_1, \ldots, j_n, a, a_1, \ldots, a_n \in \mathbb{N}$. Assume φ_i is total and $a \notin K$. Then $\varphi_{AM(u(i,a),u(j_1,a_1),\ldots,u(j_n,a_n))}$ is total.

INFERENCE-ALGORITHM M
To infer $f \in REC$, initialize $c(0) = 0$. For all stages s and input $f_s = (f(0), \ldots, f(s))$ do the following three steps:

1) Let $I(s)$ be

$$\{u(j,a) : c(s) \leq j \leq s \text{ and } a \leq s \text{ and } g(c(s),j,a) = COMP\}.$$

2) Output $M(f_s) := AM(I(s))$. Note that by Fact 5.20.*iii* $\varphi_{M(f_s)}$ is compatible with $\varphi_{c(s)}$.

3) If one of the following conditions holds, then let $c(s+1) = c(s)+1$ else $c(s+1) = c(s)$.

(a) $(\exists s' < s)[c(s') = c(s)$ and $\varphi_{M(f_{s'}),s}$ is incompatible with $f_s]$. Note that if $\varphi_{c(s)}$ is total then, by Fact 5.20.*iii*, $\varphi_{M(f_{s'})} \subseteq \varphi_{c(s)}$; and since $\varphi_{M(f_{s'})}$ is incompatible with f, $\varphi_{c(s)}$ is incompatible with f.

(b) $(\exists j, a, t \leq s)[a \in K_t$ and $\varphi_{u(j,a),t} \subseteq f_s \wedge g(c(s), j, a) = INCOMP]$. By Fact 5.20.*i* $\varphi_{c(s)}$ is not an extension of $\varphi_{u(j,a)}$, hence $\varphi_{c(s)}$ is not an extension of f_s.

END of INFERENCE-ALGORITHM

For a given recursive f there is a least index i such that $\varphi_i = f$.

Claim 1 $(\forall s)[c(s) \leq i]$.

Assume there is a stage s_0 such that $c(s_0) = i$. By steps 1 and 2 of the inference-algorithm, for all $s \geq s_0$ such that $c(s) = i$, $\varphi_{M(f_s)}$ is compatible with $\varphi_i = f$. We show that for all $s \geq s_0$ conditions (a) and (b) of step 3 are not satisfied at stage s. If (a) is satisfied then f is incompatible with $\varphi_i = f$, a contradiction. If (b) is satisfied then φ_i is not an extension of f_s; since $\varphi_i = f$ this is a contradiction.

Therefore, $c(s)$ converges to some limit $k \leq i$. Say, $c(s) = k$ for all $s \geq s_0$.

Claim 2 $\varphi_{M(f_s)} = f$ for almost all stages s.

Since $c(s+1) = c(s)$ for all stages $s > s_0$, conditions (a) and (b) are never satisfied. Hence for almost all stages $\varphi_{M(f_s)}$ is compatible with f. It suffices to show that $(\overset{\infty}{\forall} s)[\varphi_{M(f_s)}$ is total$]$. We achieve this by showing that $(\exists a \notin K)(\overset{\infty}{\forall} s)[u(i,a) \in I(s)]$ and use that φ_i is total, and Fact 5.20.*iv*. Assume, by way of contradiction, that no such a exists. Note this implies the following.

i. $(\forall a \notin K)[g(k, i, a) = INCOMP]$ (else $(\exists a \notin K)(\overset{\infty}{\forall} s)[u(i,a) \in I(s)]$).

ii. $(\forall a \in K)[g(k, i, a) = COMP]$ (else condition (b) would occur and $c(s)$ would change).

Together these items yield $K \leq_{\mathrm{T}} A$, which contradicts A being low. Hence such an a exists. ▮

Corollary 5.21 *There is a low ω-r.e. set A such that $REC \in BC[A]$. (For a definition of ω-r.e. see [EHK81].)*

Proof:

A careful examination of the proof of the low basis theorem reveals that the low set produced is ω-r.e. Hence the low set constructed in Theorem 5.18 is ω-r.e. ▮

Corollary 5.22 *There is a set A of hyperimmune-free degree such that $REC \in BC[A]$. (For a definition of hyperimmune-free see [Soa87].)*

Proof:

Every infinite recursive tree has a hyperimmune-free branch (see [Soa87, p. 109, 5.15]). If this is used instead of the Low Basis Theorem in the proof of Theorem 5.18 then one obtains a hyperimmune-free set A such that $REC \in BC[A]$. ▮

5.3.3 Arbitrary High Double Jump is not Enough

The following theorem was originally proven directly in [CDF+92]. We present a simpler proof based on Theorem 5.8.

Theorem 5.23 *For any X there exists a G such that $X \leq_{\mathrm{T}} G''$ and $REC \notin BC[G]$.*

Proof:

By Lemma 5.4.*ii* there exists a set Y such that $K \leq_{\mathrm{T}} Y$ and $X \oplus \emptyset'' \equiv_{\mathrm{T}} Y' \equiv_{\mathrm{T}} Y \oplus \emptyset''$. By Lemma 5.4.*i* there is a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} Y$. So $X \leq_{\mathrm{T}} X \oplus \emptyset'' \equiv_{\mathrm{T}} G''$. Assume now that $REC \in BC[G]$. Then by Lemma 4.20 $REC \in BC[G*]$. This implies, by Theorem 5.8, that $\emptyset'' \leq_{\mathrm{T}} G \oplus K$ and $Y \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} Y \oplus \emptyset'' \equiv_{\mathrm{T}} Y'$, a contradiction. Thus $REC \notin BC[G]$. ▮

6 Other Notions of inference

In this section we explore other notions of inference. In each subsection we define a notion of inference (e.g. EX_n); the corresponding notions of inference-with-an-oracle (e.g. $EX_n[A]$) are obtained in a manner similar to the definition of $EX[A]$, and hence are omitted.

6.1 $[a,b]EX$ and $[a,b]BC-$ Teams of Machines

The reader is referred to Definition 2.30 for a definition of $[a,b]EX$ and $[a,b]BC$. As an example, the set $S_1 \cup S_2$ from Example 2.15 is in $[1,2]EX$.

We show the following.

a) $[a,b]EX[A*] = [a,b]EX$ iff $[a,b]BC[A*] = [a,b]BC$ iff $A \leq_{\mathrm{T}} K$.

b) $[a,b]EX[A] = [a,b]EX$ iff $[a,b]BC[A] = [a,b]BC$ iff $\mathcal{G}(A)$.

c) $REC \in [a,b]EX[A*]$ iff $REC \in [a,b]BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

d) $REC \in [a,b]EX[A]$ iff $\emptyset'' \leq_{\mathrm{T}} A'$.

e) The question of when $REC \in [a,b]BC[A]$ seems similar to that of when $REC \in BC[A]$.

Note 6.1 Pitt and Smith [PS88] showed that $[a,b]EX = [1,\lceil \frac{b}{a} \rceil]EX$ and $[a,b]BC = [1,\lceil \frac{b}{a} \rceil]BC$. Their proofs relativize. We state and proof our results in terms of $[a,b]EX[A]$ ($[a,b]BC[A]$, etc.). The proofs in this paper would be no easier if done for $[1,c]EX[A]$ ($[1,c]BC[A]$, etc.).

6.1.1 $*$-Triviality

Theorem 6.2 *$[a,b]EX[A*] = [a,b]EX$ iff $[a,b]BC[A*] = [a,b]BC$ iff $A \leq_{\mathrm{T}} K$.*

Proof:

Assume $A \leq_{\mathrm{T}} K$. Let $S \in [a,b]EX[A*]$ ($[a,b]BC[A*]$) via $M_1^A, \ldots, M_b^A$. By Lemma 4.8, for each M_i^A there exists an IIM N_i that EX-infers (BC-infers) the same set that M_i^A did. It is easy to see that $S \in [a,b]EX$ ($S \in [a,b]BC$) via $N_1, \ldots, N_b$.

Assume $[a,b]BC[A*] = [a,b]BC$. Let S_b^A be from Definition 4.12 with $m = b$. Note that $S_b^A \in [a,b]BC[A*] = [a,b]BC \subseteq [1,b]BC$. By Lemma 4.14 $A \leq_{\mathrm{T}} K$. The same proof works for $[a,b]EX[A*]$. ∎

6.1.2 Triviality

Theorem 6.3 *$[a,b]EX[A] = [a,b]EX$ iff $[a,b]BC[A] = [a,b]BC$ iff $\mathcal{G}(A)$.*

Proof:

Assume $\mathcal{G}(A)$. Let $S \in [a,b]EX[A]$ ($[a,b]BC[A]$) via $M_1^A, \ldots, M_b^A$. By Theorem 4.21, for each M_i^A, there exists an IIM N_i that EX-infers (BC-infers) the same set of functions as M_i^A. Clearly $S \in [a,b]EX$ BC) via $N_1, \ldots, N_b$.

A modification of the proof of Theorem 4.1 (or see [KS93b]) yields

$$\begin{aligned} [a,b]EX[A] = [a,b]EX &\Rightarrow \mathcal{G}(A), \text{ and} \\ [a,b]BC[A] = [a,b]BC &\Rightarrow \mathcal{G}(A) \end{aligned}$$

▮

6.1.3 $*$-Omniscience

In this section we show that $REC \in [a,b]EX[A*]$ iff $REC \in [a,b]BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$. We will need to use results from Section 6.1.4; however those results do not depend on these. The proof for EX is an easy corollary of Corollary 6.21. The proof for BC is a modification of the proof of Theorem 5.8.

Theorem 6.4 *$REC \in [a,b]EX[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

Proof:

Assume $REC \in [a,b]EX[A*]$. By Lemma 5.4 applied to $A \oplus K$ there is a set B such that $B' \equiv_{\mathrm{T}} A \oplus K$; and by Lemma 5.5 $REC \in [a,b]EX[B]$. By Corollary 6.21 B is high and therefore $\emptyset'' \leq_{\mathrm{T}} A \oplus K$. The other direction follows from Theorem 5.7. ▮

Lemma 6.5 *For any A there is a 1-generic G such that $[a,b]BC[G*] = [a,b]BC[G] = [a,b]BC[A*]$ and $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$.*

Proof:

By applying Lemma 5.4.i to $Z = A \oplus K$ we obtain a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$. Since G is 1-generic, by an easy modification of the proof of Lemma 4.20, $[a,b]BC[G*] = [a,b]BC[G]$. Since $A \leq_{\mathrm{T}} G'$, by an easy modification of the proof of Lemma 5.5,

$$[a,b]BC[A*] \subseteq [a,b]BC[G] = [a,b]BC[G*].$$

Since $G \leq_{\mathrm{T}} A \oplus K$,

$$[a,b]BC[G*] \subseteq [a,b]BC[(A \oplus K)*] \subseteq [a,b]BC[A*]$$

(the last inclusion is obtained by Note 4.10). Hence

$$[a,b]BC[G*] = [a,b]BC[G] = [a,b]BC[A*]$$

as desired. ▮

Theorem 6.6 *$REC \in [1,n]BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

Proof:

Clearly $\emptyset'' \leq_{\mathrm{T}} A \oplus K$ implies $REC \in EX[A*] \subseteq [1,n]BC[A*]$.

We prove the converse by induction on n. For $n = 1$ this is Theorem 5.8. Assume the inductive hypothesis to be true for $n-1$.

Assume $REC \in [1,n]BC[A*]$. By Lemma 6.5, there is a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$ and $[1,n]BC[G*] = [1,n]BC[A*]$ (we will not be using the fact that it is 1-generic). Since $REC \in [1,n]BC[A*]$, $REC \in [1,n]BC[G*]$. Let $REC \in [1,n]BC[G*]$ via $M_1^G, \ldots, M_n^G$.

There are two cases. In the first one we obtain $REC \in [1,n-1]BC[G*]$; we can then use the induction hypothesis and the nature of G to obtain $\emptyset'' \leq_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$. In the second one we show that G is high by constructing a G-recursive function g that dominates all recursive functions. Once G is high we have $\emptyset'' \leq_{\mathrm{T}} G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$, as desired. This case is similar to the proof of Theorem 5.8. (This case does not use the induction hypothesis.)

Case 1: Assume there exist i, σ such that some M_i^G does not BC-converge on any recursive function f with $\sigma \preceq f$. Then $REC \in [1,n-1]BC[G*]$ by reducing this problem to the task of inferring all recursive functions which begin with σ. By the induction hypothesis and the nature of G we have $\emptyset'' \leq_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} A \oplus K$.

Case 2: Assume the negation, namely that for all i, σ there exists a recursive function f such that $\sigma \preceq f$ and M_i^G infers f. In particular, for every i, σ, there is some $\sigma' \succ \sigma$ such that $M_i^G(\sigma')$ is the index of a total function. We sketch a modification of the proof of Theorem 5.8 to show that $\emptyset'' \leq_{\mathrm{T}} G \oplus K$. We describe how to modify the first, second, and third parts of the proof of Theorem 5.8 to meet our needs.

In the first part the index of the machine is added to the definition of ψ, which is intended to diagonalize all machines. For the base case we define

$$\begin{aligned} \psi(\sigma, 0) &= \sigma, \\ T(\sigma, 0) &= \{(D, i) : D \subseteq \{0, \ldots, |\sigma|\} \ \wedge i \in \{1, \ldots, n\}\}. \end{aligned}$$

Assume $\psi(\sigma, t)$ and $T(\sigma, t)$ have been defined. If there exists some $(D, i) \in T(\sigma, t)$, some string τ with $|\tau| < t$ and some $b \in \{0, 1\}$ such that

$$\begin{gathered} \varphi_{M_i^D(\psi(\sigma,t)\cdot\tau)}(|\psi(\sigma, t) \cdot \tau|) \downarrow \neq b \text{ within } t \text{ steps} \\ \text{or } M_i^D(\psi(\sigma, t) \cdot \tau) \text{ queries some } x > |\sigma| \end{gathered}$$

then take the first such (D, i, τ, b) and let

$$\begin{aligned} \psi(\sigma, t+1) &= \psi(\sigma, t) \cdot \tau \cdot b, \\ T(\sigma, t+1) &= T(\sigma, t) - \{(D, i)\} \end{aligned}$$

else ψ, T remain unchanged.

The second part is adapted such that g is defined to wait until all machines are diagonalized:

$$\begin{aligned} \hat{g}(\sigma) &= (\mu t > |\sigma|)[\forall i \ [\ (G \upharpoonright |\sigma|, i) \notin T(\sigma, t)]] \\ g(n) &= \max_{|\sigma| \leq n}\{\hat{g}(\sigma)\} \end{aligned}$$

The proof that g is total recursive in G is similar to that in Theorem 5.8.

In the third part, we show that g dominates all recursive functions. This proof is similar to that in Theorem 5.8: we assume that g does not dominate all recursive functions and, using this, construct a recursive f that is not inferred by any M_i^G, a contradiction to $REC \in [1, n]BC[G*]$ via $M_1^G, \ldots, M_n^G$. ∎

Corollary 6.7 *$REC \in [a, b]BC[A*]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

6.1.4 Omniscience For $[a,b]EX[A]$

In this section we show that $REC \in [a,b]EX[A]$ iff $\emptyset'' \leq_{\rm T} A'$ (by Theorem 5.18 this is false for $[a,b]BC[A]$). The proof involves looking carefully at how Adleman and Blum [AB91] proved that if $REC \in EX[A]$ then A is high.

Definition 6.8 $REC_{0,1}$ denotes the recursive 0-1 valued functions. The *functions of finite support* are the functions in $REC_{0,1}$ which are almost everywhere 0. We denote the set of all such functions by FS.

Definition 6.9 Let $h \in REC$. A total recursive function f is *h-hard* if, for all φ_i that compute f, $(\stackrel{\infty}{\forall} x)[\Phi_i(x) > h(x)]$. (Any function that computes f takes more time than $h(x)$ almost always.)

Definition 6.10 Let $h \in REC$ and $g \in REC_{0,1}$. g is *h-sparse* if for all x, if $g(x) = 1$ then $g(x+1) = \cdots = g(x+1+h(x)) = 0$. The function $g_{\rm pair}$ is defined by $g_{\rm pair}(x) = (g(2x), g(2x+1))$. $g_{\rm pair}$ is h-sparse if for all x, if $g_{\rm pair}(x) \neq (0,0)$ then $g_{\rm pair}(x+1) = \cdots = g_{\rm pair}(x+h(x)) = (0,0)$.

Notation 6.11 Let $S \subseteq REC_{0,1}$. The following conditions will be referred to as $C1, C2, C3$ and $C4$.

$C1)$ $FS \subseteq S$.

$C2)$ $(\forall h \in REC)(\exists g \in S)[g$ is h-sparse and h-hard$]$.

$C3)$ $(\forall h \in REC)(\exists g \in S)[g_{\rm pair}$ is h-sparse and h-hard$]$.

$C4)$ $(\forall g \in REC_{0,1} - FS)(\exists \hat{g} \in S)[$

$$\begin{array}{ll} g(x) = 0 & \Rightarrow \hat{g}_{\rm pair}(x) = (0,0), \text{ and} \\ g(x) = 1 & \Rightarrow \hat{g}_{\rm pair}(x) \in \{(0,1),(1,0)\} \\ &]. \end{array}$$

The following facts are easily verified.

Fact 6.12 *Let h be an increasing recursive function. Let $g \in REC_{0,1}$.*

i. If g_{pair} is $h(x)$-sparse then g is $2h(\lfloor \frac{x}{2} \rfloor)$-sparse.

ii. If g_{pair} is $h(x)$-hard then, if $\varphi_i = g$, $(\overset{\infty}{\forall} x)[\Phi_i(2x) + \Phi_i(2x+1) \geq h(x)]$.

Lemma 6.13 *Let $S \subseteq REC_{0,1}$.*

i. If S satisfies $C1$ and $C3$ then ($S \in EX[A] \Rightarrow A$ is high).

ii. If S satisfies $C4$ then S satisfies $C3$.

iii. If S satisfies $C1$ and $C4$ then ($S \in EX[A] \Rightarrow A$ is high).

Proof:
i) The proof of Theorem 5.1 shows that if S satisfies $C1$ and $C2$ then ($S \in EX[A] \Rightarrow A$ is high). Our result can be obtained by modifying their proof: replace "$\Phi_i(x)$" by "$\Phi_i(2x) + \Phi_i(2x+1)$" and use Fact 6.12 (both parts).
ii) Let $h \in REC$. We can assume h is strictly increasing. By Lemma 2 of [AB91] $REC_{0,1}$ satisfies $C2$; hence there exists a function $g \in REC_{0,1}$ that is h-sparse and h-hard. We can assume $g \notin FS$. (This assumption is valid for the step-counting complexity measure but is not valid for some other complexity measures.) Use this g and $C4$ to obtain $\hat{g} \in S$. It is easy to see that $\hat{g}_{\text{pair}}$ is h-sparse and h-hard.
iii) This follows from i and ii. ∎

Definition 6.14 Given some string σ, let $TABLE(\sigma)$ be an index of the recursive function which outputs $\sigma(x)$ for $x < |\sigma|$ and 0 otherwise. The coding should be such that if $\sigma = \tau 0^m$ then $TABLE(\sigma) = TABLE(\tau)$.

Lemma 6.15 *If $M^{()}$ is an OIIM such that M^A infers S, then there is an OIIM $N^{()}$ such that N^A infers S and converges on all functions in FS (but N^A does not necessarily infer FS).*

Proof:

The algorithm of M^A is translated into that of N^A as follows:
Input σ, let τ be the first $|\sigma|-1$ bits of σ.
Calculate $M^A(\sigma)$, $M^A(\tau)$ and $N^A(\tau)$. There are four cases:

a) $N^A(\tau) = TABLE(\tau')$ for some $\tau' \preceq \tau$ and $\sigma = \tau' 0^{|\sigma|-|\tau'|}$
Then $N^A(\sigma) = TABLE(\tau')$.

b) $N^A(\tau) = TABLE(\tau')$ for some $\tau' \preceq \tau$ and $\sigma \neq \tau' 0^{|\sigma|-|\tau'|}$
Then $N^A(\sigma) = M^A(\sigma)$.

c) $N^A(\tau) \neq TABLE(\tau')$ for all $\tau' \preceq \tau$ and $M^A(\tau) = M^A(\sigma)$
Then $N^A(\sigma) = M^A(\sigma)$.

d) $N^A(\tau) \neq TABLE(\tau')$ for all $\tau' \preceq \tau$ and $M^A(\tau) \neq M^A(\sigma)$
Then $N^A(\sigma) = TABLE(\sigma)$.

The informal idea of the algorithm is as follows. Whenever M^A is about to change its mind, rather than output that new guess, we output a function of the form $TABLE(\sigma)$. As soon as N^A outputs a function of the form $TABLE(\sigma)$, in the next few stages it will continue to output this as long as the function continues to look like $\sigma 0^\omega$. When the function stops looking like this we output what M^A would output.

If f is inferred by M^A then eventually M^A stops changing its mind on f, so N^A infers f. If $f \in FS$ then N^A has at most two mindchanges after the last nonzero value of f; therefore N^A converges on any input from FS. ∎

Note 6.16 We may assume $\varphi_{N^A(\sigma)}$ does not contradict σ within $|\sigma|$ steps since in this case the planned output can be replaced by $TABLE(\sigma)$. So in the case of convergence to some index i, the function φ_i coincides with the inferred function f on its domain W_i.

Lemma 6.17 *Assume $REC_{0,1} \in [1,n]EX[A]$ via $M_1^A, \ldots, M_n^A$. Assume that there exists i such that the set of functions inferred by M_i^A does not fulfill condition $C4$. Then $REC_{0,1} \in [1,n-1]EX[A]$.*

Proof:

Assume the set of functions inferred by M_1^A does not fulfill condition $C4$ because of $g \in REC_{0,1} - FS$. Let S be the set of all recursive functions $\hat{g}$ such that

$$\begin{array}{ll} g(x) = 0 & \Rightarrow \hat{g}_{\text{pair}}(x) = (0,0), \text{ and} \\ g(x) = 1 & \Rightarrow \hat{g}_{\text{pair}}(x) \in \{(0,1),(1,0)\}. \end{array}$$

None of the functions in S are inferred by M_1^A; hence $S \in [1, n-1]EX[A]$ via $M_2^A, \ldots, M_n^A$. We show that $S \in [1, n-1]EX[A]$ implies that $REC \in [1, n-1]EX[A]$; we will then use the induction hypothesis.

Let $x_0, x_1, \ldots$ be the set of numbers where g takes value 1. Let $f \in REC_{0,1}$. We define a function f^+ such that $f^+ \in S$ and f^+ contains information about f. Let

$$f^+(x) = \begin{cases} f(m) & \text{if } x = 2x_m \\ 1 - f(m) & \text{if } x = 2x_m + 1 \\ 0 & \text{otherwise.} \end{cases}$$

One can verify that $f^+ \in S$. There is a total recursive function s satisfying $\varphi_{s(i)}(m) = \varphi_i(2x_m)$ and the operator $^+$ transforming f to f^+ is recursive. Let $N_i^A(f) = s(M_i^A(f^+))$. Clearly $REC_{0,1}$ is $[1, n-1]EX[A]$-identified via $N_2^A, \ldots, N_n^A$. ∎

The following lemma is easy, hence the proof is omitted.

Lemma 6.18 *For all A, $REC \in EX[A]$ iff $REC_{0,1} \in EX[A]$.*

Theorem 6.19 *$REC \in [1, n]EX[A]$ iff A is high.*

Proof:

If A is high then by Theorem 5.1 $REC \in EX[A]$. Hence $REC \in [1, n]EX[A]$.

We prove that if $REC_{0,1} \in [1, n]EX[A]$ then A is high, and then use Lemma 6.18. We prove this by induction on n. If $n = 1$ then this is Theorem 5.1. Assume that $n \geq 2$ and that the theorem is true for $n - 1$.

Let $REC_{0,1} \in [1, n]EX[A]$ via $M_1^A, \ldots, M_n^A$. By Lemma 6.15 we can assume that each M_i^A converges on all elements of FS. By Note 6.16 we can also assume that $\varphi_{M_i^A(\sigma)}$ does not contradict σ when run for $\leq |\sigma|$ steps. By padding we can assume that $(\forall i \neq j)[range(M_i^A) \cap range(M_j^A) = \emptyset]$.

We define new machines $N_1^{()}, \ldots, N_n^{()}$ such that $REC_{0,1} \in [1,n]EX[A]$ via $N_1^A, \ldots, N_n^A$. The idea is that if, on input $f \in REC_{0,1}$, exactly i of the $M_1^A, \ldots, M_n^A$ converge to programs that do not contradict f, then N_i^A infers f.

We need the following definition.

Definition 6.20 Let $\sigma = \sigma_1 \cdots \sigma_k$ where $\sigma_i \in \{0,1\}$. If $M^A(\sigma) = e$ then the *confidence* $M^A(\sigma)$ *has in* e is the largest number m such that

$$M^A(\sigma_1 \cdots \sigma_{k-m}) = M^A(\sigma_1 \cdots \sigma_{k-m+1}) = \cdots = M^A(\sigma_1 \cdots \sigma_k).$$

ALGORITHM FOR N_i^A.

i. Input(σ). Let $s = |\sigma|$.

ii. $(\forall j)[1 \leq j \leq n]$ compute $e_j = M_j^A(\sigma)$. Let $I = \{e_j : (\forall x < |\sigma|)[\varphi_{e_j,s}(x)\downarrow \Rightarrow \varphi_{e_j,s}(x) = \sigma(x)]\}$. If $|I| < i$ then output 0 and halt.

iii. For all $e_j \in I$ compute c_j, the confidence $M_j^A(\sigma)$ has in e_j.

iv. Find $c_{j_1}, \ldots, c_{j_i}$, the i largest values of c_j (if there is a tie then break it arbitrarily).

v. Output $AM(e_{j_1}, \ldots, e_{j_i})$. (See Definition 2.32 for the definition of AM.)

END OF ALGORITHM

It is easy to see that, for all $f \in REC_{0,1}$, if on input f exactly i of $M_1^A, \ldots, M_n^A$ converge to a program that does not contradict f, then N_i^A infers f (we need that all the machines have disjoint ranges). Hence $REC_{0,1} \in [1,n]EX[A]$ via $N_1^A, \ldots, N_i^A$. Since all the M_i^A converge on all $f \in FS$ we have that N_n^A infers FS.

Let S be the set of functions in $REC_{0,1}$ which are inferred by N_n^A. S satisfies $C1$. If S satisfies $C4$ then by Lemma 6.13.*iii*, since $S \in EX[A]$ via N_n^A, A is high. If S does not satisfy $C4$ then by Lemma 6.17 $REC_{0,1} \in [1,n-1]EX[A]$; therefore, by the induction hypothesis A is high. ∎

Corollary 6.21 *$REC \in [a,b]EX[A]$ iff A is high.*

6.1.5 Omniscience For $[a,b]BC[A]$

The question of when $REC \in [a,b]BC[A]$, much like the question of when $REC \in BC[A]$, does not appear to have a clean answer.

Theorem 6.22 *The following are true.*

i. For all high sets A, $REC \in [a,b]BC[A]$.

ii. There exists a low set A such that $REC \in [a,b]BC[A]$

iii. For any X such that $\emptyset'' \leq_{\mathrm{T}} X$ there exists a set G such that $X \leq_{\mathrm{T}} G''$ and $REC \notin [a,b]BC[G]$.

iv. If A is r.e. then $REC \in [a,b]BC[A]$ iff $\emptyset'' \leq_{\mathrm{T}} A'$.

Proof:

i) If A is high then, by Theorem 5.1, $REC \in EX[A] \subseteq [a,b]BC[A]$.

ii) Let A be the low set constructed in Theorem 5.18. For this A, $REC \in BC[A] \subseteq [a,b]BC[A]$.

iii) By Lemma 5.4.ii there is a set Y such that $K \leq_{\mathrm{T}} Y$ and $X \oplus \emptyset'' \equiv_{\mathrm{T}} Y' \equiv_{\mathrm{T}} Y \oplus \emptyset''$. By Lemma 5.4.$i$ there is a 1-generic set G such that $G' \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} Y$. So $X \leq_{\mathrm{T}} X \oplus \emptyset'' \equiv G''$. Assume now that $REC \in [a,b]BC[G]$. Then by an easy modification of the proof of Corollary 4.20 $REC \in [a,b]BC[G*]$. This implies, by Corollary 6.7, that $\emptyset'' \leq_{\mathrm{T}} G \oplus K$ and $Y \equiv_{\mathrm{T}} G \oplus K \equiv_{\mathrm{T}} Y \oplus \emptyset'' \equiv_{\mathrm{T}} Y'$, a contradiction. Thus $REC \notin [a,b]BC[G]$.

iv) This was proven by Kummer and Stephan [KS93b].

▮

6.2 EX^n, EX^*, BC^n, and BC^*– Allowing errors

Definition 6.23 $S \in EX^n$ ($S \in EX^*$) if there exists an IIM M such that for all $f \in S$, when M is run on initial segments of f, almost all the programs output are the same, and the function computed by that program differs from f on at most n numbers (on some finite set of numbers).

The definitions of BC^n and BC^* look similar to those of EX^n and EX^* but are actually quite different.

Definition 6.24 $S \in BC^n$ ($S \in BC^*$) if there exists an IIM M such that for all $f \in S$, when M is run on initial segments of f, almost all the programs output compute functions that differ from f on at most n numbers (on some finite set of numbers).

Note 6.25 Assume $S \in BC^n$ via M and $f \in S$. If f is fed into M then the programs output in the limit may compute different functions, differing from f at different sets of n numbers. For $S \in BC^*$ the situation is worse — the programs output in the limit may be computing different functions, differing from f at different finite sets, perhaps even larger and larger finite sets. In fact, BC^*-inference is so powerful that $REC \in BC^*$ (see Note 2.21 or see Harrington's proof in [CS83]).

Note 6.26 Assume $S \in EX^*$ via IIM M. We can adjust the IIM M so that if φ_e is the program output in the limit and x is a number such that $\varphi_e(x) \neq f(x)$ then $\varphi_e(x)\uparrow$. First, let u be the total recursive function defined by

$$\varphi_{u(e,\langle a_0,\ldots,a_s\rangle)}(x) = \begin{cases} a_i & \text{if } x = i \leq s; \\ \varphi_e(x) & \text{otherwise.} \end{cases}$$

We adjust M as follows. If $M(\langle f(0),\ldots,f(s)\rangle) = e$, then compute, for all $x \leq s$, the value $\varphi_{e,s}(x)$. If none of them converge and differ from f then output e. If any of them converge and differ from f then output $u(e,\langle f(0),\ldots,f(a)\rangle)$ where $a = \max\{i : i \leq s \text{ and } f(i) \neq \varphi_{e,s}(i)\downarrow\}$.

Definition 6.27 If f is a function then *the cylindrification of* f is the function $\tilde{f}(\langle x,y\rangle) = f(x)$. Let $S \subseteq REC$. The *the cylindrification of* S is $\tilde{S} = \{\tilde{f} : f \in S\}$.

Fact 6.28 *The following hold for all* n, A.

i. *If* $\tilde{S} \in BC^n[A]$ *then* $S \in BC[A]$.

ii. *If* $\tilde{S} \in BC^n[A*]$ *then* $S \in BC[A*]$.

iii. *If* $\tilde{S} \in EX^*[A]$ *then* $S \in EX[A]$.

iv. *If* $\tilde{S} \in EX^*[A*]$ *then* $S \in EX[A*]$.

v. *For* $I \in \{EX[A], EX[A*], BC[A], BC[A*]\}$ *, and for any* $S \subseteq REC$*,* $S \in I \Rightarrow \tilde{S} \in I$*. (This is obvious so its proof is omitted.)*

vi. *For any notion of inference* I *discussed in this paper, if* $REC \in I$ *then* $\widetilde{REC} \in I$*. (This follows from* $\widetilde{REC} \subseteq REC$*.)*

Proof:

i) Let M^A be a machine that $BC^n[A]$-infers $\tilde{S}$. We $BC[A]$-infer S as follows. Upon seeing initial segment σ of $g \in S$ we construct an initial segment τ of the corresponding $f \in \tilde{S}$ that is as long (and contiguous) as possible given σ. Compute $M^A(\tau) = e$. Output the index of a function that does the following: On input x compute $\varphi_e(\langle x, i\rangle)$ for $i = 0, 1, \ldots$ until $n + 1$ of them converge and agree, and then output that value.

ii) The proof of *i* also works for $BC^n[A*]$.

iii) Let M^A be a machine that $EX^*[A]$-infers $\tilde{S}$. By Note 6.26 we can assume that for all $f \in \tilde{S}$, when M^A tries to infer f it produces (in the limit) a program that only differs from f by diverging. We $EX[A]$-infer S as follows. Upon seeing initial segment σ of $g \in S$ we construct an initial segment τ of the corresponding $f \in \tilde{S}$ that is as long (and contiguous) as possible given σ. Compute $M^A(\tau) = e$. Output the index of a function that does the following: On input x compute $\varphi_e(\langle x, i\rangle)$ for $i = 0, 1, \ldots$ until one of them converges, and then output that value.

iv) The proof of *iii* also works for $EX^*[A*]$.

∎

Recall that $\mathcal{G}(A)$ means that either A is recursive or $A \leq_{\mathrm{T}} K$ and is in a 1-generic degree.

Theorem 6.29 *The following hold for all* n, A*.*

i. $EX^n[A] = EX^n$ *iff* $EX^*[A] = EX^*$ *iff* $\mathcal{G}(A)$.

ii. $EX^n[A*] = EX^n$ *iff* $EX^*[A*] = EX^*$ *iff* $BC^n[A*] = BC^n$ *iff* $A \leq_{\mathrm{T}} K$.

iii. $REC \in EX^n[A]$ *iff* $REC \in EX^*[A]$ *iff* $\emptyset'' \leq_{\mathrm{T}} A'$.

iv. $REC \in EX^n[A*]$ *iff* $REC \in EX^*[A*]$ *iff* $REC \in BC^n[A*]$ *iff* $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

v. $REC \in BC[A]$ *iff* $REC \in BC^n[A]$.

vi. *For all* A, $REC \in BC^*[A] = BC^*[A*] = BC^*$.

Proof:

The results about when $EX^n[A]$ and $EX^*[A]$ are trivial and $*$-trivial, and when $BC^n[A]$ is $*$-trivial, are obtained as follows. By a simple modification of the proof of Lemmas 4.19 and 4.9 we have the following.

$$\begin{array}{rl} \mathcal{G}(A) & \Rightarrow (EX^n[A] = EX^n \text{ and } EX^*[A] = EX^*). \\ A \leq_{\mathrm{T}} K & \Rightarrow (EX^n[A*] = EX^n \text{ and } EX^*[A*] = EX^* \text{ and } BC^n[A*] = BC^n). \end{array}$$

We show that

$$\begin{array}{rl} \neg\mathcal{G}(A) & \Rightarrow (EX^n[A] \neq EX^n \text{ and } EX^*[A] \neq EX^*). \\ A \not\leq_{\mathrm{T}} K & \Rightarrow (EX^n[A*] \neq EX^n \text{ and } EX^*[A*] \neq EX^* \text{ and } BC^n[A*] \neq BC^n). \end{array}$$

If $\mathcal{G}(A)$ does not hold then by Theorem 4.1 there exists $S \in EX[A] - EX$. By Fact 6.28 (parts *iii* and *v*) $\tilde{S} \in EX[A] - EX^*$. Hence $EX^*[A] \neq EX^*$ and $EX^n[A] \neq EX^n$. The proof for $A \not\leq_{\mathrm{T}} K$ is similar.

The results about omniscience and $*$-omniscience (except those for BC^*) are obtained as follows. By Theorems 5.1,5.7, and $EX^*[A] \subseteq BC[A] \subseteq BC^n[A]$ we have the following.

$$\begin{array}{rl} \emptyset'' \leq_{\mathrm{T}} A' & \Rightarrow REC \in EX[A] \subseteq EX^n[A] \subseteq EX^*[A]. \\ \emptyset'' \leq_{\mathrm{T}} A \oplus K & \Rightarrow REC \in EX[A*] \subseteq EX^n[A*] \subseteq EX^*[A*] \subseteq BC^n[A*]. \\ REC \in BC[A] & \Rightarrow REC \in BC^n[A]. \end{array}$$

By Fact 6.28 (parts i, ii, iii, iv, and vi) we have the following.

$$\begin{array}{l} REC \in EX^*[A*] \Rightarrow \widetilde{REC} \in EX^*[A*] \Rightarrow REC \in EX[A*] \Rightarrow \emptyset'' \leq_{\mathrm{T}} A \oplus K. \\ REC \in EX^*[A] \Rightarrow \widetilde{REC} \in EX^*[A] \Rightarrow REC \in EX[A] \Rightarrow \emptyset'' \leq_{\mathrm{T}} A'. \\ REC \in BC^n[A*] \Rightarrow \widetilde{REC} \in BC^n[A*] \Rightarrow REC \in BC[A*] \Rightarrow \emptyset'' \leq_{\mathrm{T}} A \oplus K. \\ REC \in BC^n[A] \Rightarrow \widetilde{REC} \in BC^n[A] \Rightarrow REC \in BC[A]. \end{array}$$

By Note 2.21 (or see Harrington's proof in [CS83]) $REC \in BC^*$. Hence, for all A, $REC \in BC^* = BC^*[A*] = BC^*[A]$. ▮

It is open to determine, for which A, $BC^n[A] = BC^n$. By a modification of the proof of Theorem 4.1 (or see [KS93b]) one can show that $BC^n[A] = BC^n \Rightarrow \mathcal{G}(A)$. The difficulty in establishing the converse is that the proof of Lemma 4.20 does not seem to apply to BC^n.

6.3 EX_n– Bounding Mind Changes

Definition 6.30 $S \in EX_n$ if there exists an IIM M such that for all $f \in S$, M EX-identifies f, and changes its guess about the function at most n times. (Formally, we allow an IIM to guess 0 which means 'no guess at this time' and do not count the first real guess as a mindchange.)

Note 6.31 Let T_1, T_2 be the sets from Example 2.26. Clearly $T_1 \in EX_0[A[64]]$. Using binary search $T_1 \in EX_0[A[6]]$. (Actually $T_1 \in EX_0$ since $|T_1|$ is finite.) Clearly $T_2 \in EX_0[A*]$.

We show that, for $n \in \mathbb{N}$, both the EX_n-degrees and the EX_n*-degrees are identical to the Turing degrees. As corollaries we obtain that, for all A, (1) $EX_n[A] = EX_n$ iff A is recursive, (2) $REC \notin EX_n[A]$, (3) $EX_n[A*] = EX_n$ iff A is recursive, and (4) $REC \notin EX_n[A*]$.

We prove a lemma about when a certain set of functions can be in $[a,b]EX^*_m[B]$. As a corollary we obtain information about how EX_k and $EX_m[B]$ compare. We will use the full strength of this lemma in Section 6.5.

Lemma 6.32 *Let* $S_k = \{0^*1^* \cdots (p-1)^* p^\omega : p \le k\}$.

i. *If* $a(k+1) \le b(m+1)$ *then* $S_k \in [a,b]EX_m$

ii. *If* $S_k \in [a,b]EX^*[B]$ *via* $M_1^B, \ldots, M_b^B$ *and* $(\forall i \ne j)(\forall \sigma, \tau)[M_i^B(\sigma) \ne M_j^B(\tau)]$ *(the machines can easily be modified to make this true) then there exists* $\sigma \in 0^*1^* \cdots k^*$ *such that the total number of guesses made by the team while being fed* σ *is at least* $a(k+1)$.

iii. *If* $S_k \in [a,b]EX^*_m[B]$ *then* $a(k+1) \le b(m+1)$.

Proof: We will assume throughout the proof that a divides b. The modifications needed for the case where a does not divide b are easy.

i) Assume that $a(k+1) \le b(m+1)$, so $k \le \frac{b}{a}m + \frac{b}{a} - 1$. For $1 \le i \le \frac{b}{a}$ let

$$T_i = \{0^*1^* \cdots (p-1)^* p^\omega : (i-1)m + i - 1 \le p \le im + i - 1\}.$$

Clearly each T_i is in EX_m and $S_k \subseteq \bigcup_{i=1}^{b/a} T_i$. Hence $S_k \in [a,b]EX_m$ by a team of b machines which consist of $\frac{b}{a}$ groups of a machines each where the ith group of machines are all EX_m machines for T_i.

ii) Let

$$W(\sigma) = \{i : (\exists \tau \prec \sigma)(\exists J \subseteq \{1,\ldots,b\}, |J| = a)(\forall j \in J)[\varphi_{M_j^B(\tau)} =^* \lambda x[i]]\}.$$

We construct a sequence of strings as follows. Let σ_{-1} be the empty string. Let $s \leq k-1$. Inductively assume $\sigma_s \in 0^*1^* \cdots s^*$. Let

$$\begin{aligned} i_{s+1} = &\quad \mu i[s+1 \in W(\sigma_s \cdot (s+1)^i)] \text{ (such } i \text{ exists since } \sigma_s(s+1)^\omega \in S_k) \\ \sigma_{s+1} = &\quad \sigma_s \cdot (s+1)^{i_{s+1}} \end{aligned}$$

Let $\sigma = \sigma_k$. When $M_1^B, \ldots, M_b^B$ are fed σ we obtain a different indices for functions that are $=^* \lambda x[p]$ for every p, $0 \leq p \leq k$. Hence we obtain at least $a(k+1)$ different indices.

iii) Assume $S_k \in [a,b]EX_m^*$ via $M_1^B, \ldots, M_b^B$. We may assume that, for all σ, i, when M_i^B is fed σ it makes $\leq m+1$ guesses; hence the total number of guesses made by the team while inferring any function is at most $b(m+1)$. Let σ be obtained by applying part ii to $M_1^B, \ldots, M_b^B$. The function $f = \sigma k^\omega$ is in S_k. When $M_1^B, \ldots, M_b^B$ tries to infer f there are at least $a(k+1)$ guesses generated. Hence $a(k+1) \leq b(m+1)$.

▮

Corollary 6.33 *If $EX_k \subseteq EX_m[B]$ then $k \leq m$.*

Proof:

By Lemma 6.32.i, with $a = b = 1$ and $m = k$, $S_k \in EX_k$. By the hypothesis $S_k \in EX_m[B]$. By Lemma 6.32.iii we obtain $k \leq m$. ▮

The next lemma deals with the Turing degrees. It uses the material on bounded queries discussed in Definitions 4.2, 4.3, and Propositions 4.6.

Lemma 6.34 *If $EX_0[A[n+1]] \subseteq EX_n[B]$ then $A \leq_{\mathrm{T}} B$.*

Proof:

Let S be

$$\{\langle x_1, \ldots, x_{n+1}\rangle \sigma i^\omega : i = NUM(F_{n+1}^A(x_1, \ldots, x_{n+1})) \text{ and } \sigma \in (\mathbb{N} - \{i\})^*\}.$$

Clearly $S \in EX_0[A[n+1]]$. Assume $S \in EX_n[B]$ via M^B. We show that F_{n+1}^A is $(n+1)$-enumerable-in-B. By Proposition 4.6 this shows $A \leq_{\mathrm{T}} B$.

The enumeration is achieved by trying to construct a function not inferred by M^B. While constructing it we find information about A that leads to possibilities for $F_{n+1}^A(x_1, \ldots, x_{n+1})$.

Given $x_1, \ldots, x_{n+1}$ we enumerate possibilities as follows.

ENUMERATION

Stage 0: Let σ be such that $\sigma(0) = \langle x_1, \ldots, x_{n+1} \rangle$ and $(\forall i > 0)[\sigma(i)$ is undefined]. We assume $M^B(\sigma) = 0$ (not a real guess). Set $P := \{0,1\}^{n+1}$. (P stands for Possibilities for $F_{n+1}^A(x_1, \ldots, x_{n+1})$.) Set $CG = 0$ (CG stands for Current Guess).

Stage s+1: Search for $\tau \in P$, $N \in \mathbb{N}$, $t \in \mathbb{N}$, $b \in \mathbb{N}$ such that the following hold. (In what follows $\sigma \cdot NUM(\tau)^N$ denotes the concatenation of σ and $(NUM(\tau))^N$.)

i. $b \notin range(\sigma)$ and $b > 2^{n+1}$,

ii. $M^B(\sigma \cdot NUM(\tau)^N) \neq CG$, and

iii. $\varphi_{M^B(\sigma \cdot NUM(\tau)^N),t}(|\sigma \cdot NUM(\tau)^N|) \downarrow \neq b$.

(We will later show that during stage $s = 1$ this search must terminate. For stages $s > 1$ this search need not terminate.)

If such τ, N, t, b are found then do the following.

i. Set $CG = M^B(\sigma \cdot NUM(\tau)^N)$.

ii. Set $\sigma := \sigma \cdot NUM(\tau)^N \cdot b$.

iii. Enumerate τ.

iv. $P := P - \{\tau\}$

v. If $s + 1 = n + 1$ then halt, else go to stage $s + 2$.

END OF ENUMERATION

If stage $s \geq 1$ terminates, then when it does so we enumerate the sth possibility, force the $(s-1)$th mindchange, and make the sth guess incorrect.

Since we never allow the stage number to be $n+2$, at most $n+1$ possibilities are enumerated. We show that one of them is $\tau = F_{n+1}^A(x_1, \ldots, x_{n+1})$.

Assume, by way of contradiction, that τ is not enumerated. Let $\langle x_1, \ldots, x_{n+1}\rangle\sigma$ be the finite function produced at the end of the enumeration. (Note that whether or not stage $n+1$ is reached a finite σ is constructed.) Let $f = \langle x_1, \ldots, x_{n+1}\rangle\sigma \cdot (NUM(\tau))^\omega$ and let $i = NUM(\tau)$. We show that $f \in S$. Note that $f(0) = \langle x_1, \ldots, x_{n+1}\rangle$ and $f =^* \lambda x[i]$. The elements in the range of σ are either numbers $b > 2^n$ or numbers of the form $NUM(\tau')$ where $\tau' \neq \tau$. Hence σ does not have i in its range. Therefore $f \in S$. Hence f is inferred by M^B. We obtain a contradiction by showing that M^B does not infer f. There are two cases.

Case 1: There exists a stage s that does not terminate. Let CG be as at the beginning of stage s. By the actions taken at stage $s-1$ either $CG = 0$ or $(\exists x)[\varphi_{CG}(x) \neq \sigma(x)]$. In either case CG is not the index of any function that has initial segment σ. Since stage s does not terminate and $\tau \in P$, for every $N \in \mathbb{N}$ either $M^B(\sigma \cdot NUM(\tau)^N) = CG$ or $\varphi_{M^B(\sigma \cdot NUM(\tau)^N)}$ is not total. Hence M^B does not infer f.

Case 2: Stage $n+1$ is reached and terminates. If M^B tries to infer any function that begins with σ then before M^B has seen all of σ, M^B has already made n mindchanges. Let CG be as at the end of stage $n+1$. Note that CG is the final guess that M^B makes and that $(\exists x)[\varphi_{CG}(x) \neq \sigma(x)]$. Hence M^B cannot infer any function that begins with σ. In particular M^B cannot infer f. ∎

Note 6.35 Using Note 4.7.*ii* Lemma 6.34 can be strengthened to show that if $EX_0[A[n]] \subseteq EX_{2^n-2}[B]$ then $A \leq_{\mathrm{T}} B$. The result is optimal since there exist nonrecursive sets A such that $EX_0[A[n]] \subseteq EX_{2^n-1}$ (any nonrecursive r.e. set A will suffice).

Theorem 6.36 *$EX_m[A] \subseteq EX_n[B]$ iff $EX_m[A*] \subseteq EX_n[B*]$ iff $m \leq n$ and $A \leq_{\mathrm{T}} B$.*

Proof:

If $EX_m[A] \subseteq EX_n[B]$ or $EX_m[A*] \subseteq EX_n[B*]$, then $EX_m \subseteq EX_n[B]$ and $EX_0[A[n+1]] \subseteq EX_n[B]$. By Lemma 6.33 $m \leq n$, and by Lemma 6.34 $A \leq_{\mathrm{T}} B$.

Clearly if $m \leq n$ and $A \leq_{\rm T} B$ then $EX_m[A] \subseteq EX_n[B]$ and $EX_m[A*] \subseteq EX_n[B*]$. ▮

Corollary 6.37 *For all n, A the following hold.*

i. $EX_n = EX_n[A]$ *iff A is recursive.*

ii. $EX_n = EX_n[A*]$ *iff A is recursive.*

iii. $REC \notin EX_n[A]$.

iv. $REC \notin EX_n[A*]$.

6.4 $PEX-$ Guesses are total

Definition 6.38 $S \in PEX$ if $S \in EX$ via an IIM that, on any input, outputs an index to a total function. $S \in PEX[A]$ if $S \in EX[A]$ via M^A such that, on any input, M^A outputs a total function. Note that if $S \in PEX[A]$ via M^A then $(\forall \sigma \in \mathbb{N}^*)[\varphi_{M^A(\sigma)}$ is total], but if $A \neq B$ then there could exist a σ such that $\varphi_{M^B(\sigma)}$ is not total.

Lemma 6.39 *The following hold for all A.*

i. $PEX[A] = PEX[A*]$.

ii. $S \in PEX[A]$ *iff there exists $h \leq_{\rm T} A$ such that*

$$S \subseteq \{\varphi_{h(i)} : i \in \mathbb{N}\} \subseteq REC.$$

Proof:

i) Let $S \in PEX[A]$. Adjust the machine so that it never changes its mind unless it sees a mistake (this is possible since all guesses are total). Such a machine will only need to query the oracle a finite number of times.

ii) Let $S \in PEX[A]$ via M^A. Let *code* be a recursive bijection from $\mathbb{N}$ to $\mathbb{N}^*$. Let $h(i) = M^A(code(i))$. Clearly

$$S \subseteq \{\varphi_{h(i)} : i \in \mathbb{N}\} \subseteq REC.$$

Let $S \subseteq \{\varphi_{h(i)} : i \in \mathbb{N}\} \subseteq REC$ where $h \leq_{\mathrm{T}} A$. $S \in PEX[A]$ by a machine which outputs $h(i)$ where i is the smallest number such that $\varphi_{h(i)}$ is consistent with the input. Consistency can be checked since all the $\varphi_{h(i)}$ are total recursive. ∎

Not much is known about when $PEX[A] = PEX$. The next theorem shows that there are nonrecursive sets for which this occurs.

Theorem 6.40 *If $A \leq_{\mathrm{T}} K$ or A is in a hyperimmune-free degree then $PEX[A] = PEX$.*

Proof:
We show that $PEX[K] = PEX$: use an approximation to K and incorporate the approximation being used into the index so that if an answer is ever discovered to be wrong then the function becomes a function that is almost always zero (we need to do this to make sure that the function output is total). Since $PEX[K] = PEX$ we have, for any $A \leq_{\mathrm{T}} K$, $PEX[A] = PEX$.

Let A be in a hyperimmune-free degree, let $S \in PEX[A]$, and let h be as in Lemma 6.39.*ii*. Recall that, since A is in a hyperimmune-free degree, for every $f \leq_{\mathrm{T}} A$ there is a recursive function g such that $(\forall x)[f(x) < g(x)]$.

Let f be defined by $f(n) = \max\{\Phi_{h(i)}(j) : 0 \leq i, j \leq n\}$. Since $f \leq_{\mathrm{T}} A$ there exists a recursive g such that $(\forall x)[f(x) < g(x)]$. Let h' be defined as follows:

$$\varphi_{h'(i)}(j) = \begin{cases} \varphi_i(j) & \text{if } \Phi_i(j) \leq g(i+j); \\ 0 & \text{otherwise.} \end{cases}$$

Clearly $\{\varphi_{h(i)} : i \in \mathbb{N}\} \subseteq \{\varphi_{h'(i)} : i \in \mathbb{N}\}$, and hence $S \in PEX$. ∎

Since the hyperimmune-free degrees and the degrees that are $\leq_{\mathrm{T}} K$ are very different from each other, yet both are PEX-trivial, we do not believe there is a degree theoretic characterization of when $PEX = PEX[A]$.

Theorem 6.41 *$REC \in PEX[A]$ iff $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.*

Proof:
If $REC \in PEX[A] = PEX[A*] = PEX[A*] \subseteq EX[A*]$, then by Theorem 5.7 $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.

If $\emptyset'' \leq_{\mathrm{T}} A \oplus K$ then by Lemma 5.3 there is a recursive function h such that $REC = \{\varphi_{h(i)}\}_{i \in \mathbb{N}}$. By Lemma 6.39.*ii* $REC \in PEX[A]$. ∎

6.5 Combinations

In this subsection we examine the effect of combining mindchanges, anomalies, and teams. When the number of mindchanges is bounded the classes behave like EX_m, if not then they behave like EX (or BC).

The following six-part lemma, without oracles, was proven in [FSV89]). The proofs with oracles are similar, hence they are omitted.

Lemma 6.42 *Let i be any number.*

i. $[a, b]EX_m[A] \subseteq EX_{2b(m+1)-2}[A]$.

ii. $EX_{(m+1)b}[A] \subseteq [1, b]EX_m[A]$.

iii. $[a, b]EX_m[A*] \subseteq EX_{2b(m+1)-2}[A*]$.

iv. $EX_{(m+1)b}[A*] \subseteq [1, b]EX_m[A*]$.

v. $[a, b]EX_m[A[i]] \subseteq EX_{2b(m+1)-2}[A[i]]$.

vi. $EX_{(m+1)b}[A[i]] \subseteq [1, b]EX_m[A[i]]$.

The next fact is similar to Fact 6.28 and uses the set $\tilde{S}$ from Definition 6.27. The proof is similar to the proof of Fact 6.28, hence we omit it

Fact 6.43 *The following hold for all a, b, m, n, A.*

i. If $\tilde{S} \in [a, b]BC^n[A]$ then $S \in [a, b]BC[A]$.

ii. If $\tilde{S} \in [a, b]BC^n[A*]$ then $S \in [a, b]BC[A*]$.

iii. If $\tilde{S} \in [a, b]EX^*[A]$ then $S \in [a, b]EX[A]$.

iv. If $\tilde{S} \in [a, b]EX^*[A*]$ then $S \in [a, b]EX[A*]$.

v. If $\tilde{S} \in [a, b]EX^n_m[A]$ then $S \in [a, b]EX_m[A]$. (The proof is similar to that of Fact 6.28.i.)

vi. If $\tilde{S} \in [a, b]EX^n_m[A*]$ then $S \in [a, b]EX_m[A*]$. (The proof is similar to that of Fact 6.28.ii.)

vii. *If I is any of $[a,b]EX_m[A]$, $[a,b]EX_m[A*]$, $[a,b]EX_m[A[i]]$, $[a,b]BC[A]$, $[a,b]BC[A*]$, or $[a,b]BC[A[i]]$, then for any $S \subseteq REC$, $S \in I \Rightarrow \tilde{S} \in I$.*

viii. *For any notion of inference I discussed in this paper, if $REC \in I$ then $\widetilde{REC} \in I$.*

Note 6.44 It is not the case that $\tilde{S} \in EX_m^* \Rightarrow S \in EX_m$. To see this take $S = \{f : \varphi_{f(0)} =^* \tilde{f}\}$ where $\tilde{f}$ is the cylindrification of f (i.e., $\tilde{f}(\langle x, y\rangle) = x$). Clearly, $\tilde{S} \in EX_0^*$. One can show $S \notin EX_0$ by a standard argument. Also note that the proof that $\tilde{S} \in EX^* \Rightarrow S \in EX$ needed the fact that we could assume there were no convergent errors. The technique used to get rid of convergent errors makes the number of mindchanges unbounded.

Theorem 6.45 *The following hold for all a, b, m, n, A.*

i. *$REC \notin [a,b]EX_m^*[A]$ (hence $REC \notin [a,b]EX_m^n[A]$, $REC \notin [a,b]EX_m^*[A*]$, and $REC \notin [a,b]EX_m^n[A*]$).*

ii. *$REC \in [a,b]EX^n[A]$ iff $REC \in [a,b]EX^*[A]$ iff $REC \in [a,b]EX$ iff $\emptyset'' \leq_T A'$.*

iii. *$REC \in [a,b]EX^n[A*]$ iff $REC \in [a,b]EX^*[A*]$ iff $REC \in [a,b]EX[A*]$ iff $\emptyset'' \leq_T A \oplus K$.*

iv. *$REC \in [a,b]BC^n[A]$ iff $REC \in [a,b]BC[A]$.*

v. *$REC \in [a,b]BC^n[A*]$ iff $\emptyset'' \leq_T A \oplus K$.*

Proof:

i) This follows from Lemma 6.32.

ii) By Theorem 5.1 $\emptyset'' \leq_T A' \Rightarrow REC \in EX[A] \subseteq [a,b]EX^n[A] \subseteq [a,b]EX^*[A]$.
By Fact 6.43 (parts *iii* and *viii*)

$$REC \in [a,b]EX^n[A] \subseteq [a,b]EX^*[A] \Rightarrow \widetilde{REC} \in [a,b]EX^*[A] \Rightarrow REC \in [a,b]EX[A].$$

By Theorem 6.19 we have $\emptyset'' \leq_T A'$.

iii) By Theorem 5.7 $\emptyset'' \leq_{\mathrm{T}} A \oplus K \Rightarrow REC \in EX[A*] \subseteq [a,b]EX^n[A*] \subseteq [a,b]EX^*[A*]$. By Fact 6.43 (parts *iv* and *viii*)

$$REC \in [a,b]EX^n[A*] \subseteq [a,b]EX^*[A*] \Rightarrow \widetilde{REC} \in [a,b]EX^*[A*] \Rightarrow REC \in [a,b]EX[A*].$$

By Theorem 6.4 $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.
iv) Clearly $REC \in [a,b]BC[A] \Rightarrow REC \in [a,b]BC^n[A]$. By Fact 6.43 (parts *i* and *viii*)

$$REC \in [a,b]BC^n[A] \Rightarrow \widetilde{REC} \in [a,b]BC^n[A] \Rightarrow REC \in [a,b]BC[A].$$

v) By Theorem 5.7 $\emptyset'' \leq_{\mathrm{T}} A \oplus K \Rightarrow REC \in EX[A*] \subseteq [a,b]BC^n[A*]$. By Fact 6.43 (parts *ii* and *viii*)

$$REC \in [a,b]BC^n[A*] \Rightarrow \widetilde{REC} \in [a,b]BC^n[A*] \Rightarrow REC \in [a,b]BC[A*].$$

By Theorem 6.6 $\emptyset'' \leq_{\mathrm{T}} A \oplus K$.
▮

Theorem 6.46 *For all a, b, n, m, A the following hold.*

i. $[a,b]EX_m[A] = [a,b]EX_m$ *iff* $[a,b]EX_m[A*] = [a,b]EX_m$ *iff A is recursive.*

ii. $[a,b]EX_m^n[A] = [a,b]EX_m^n$ *iff* $[a,b]EX_m^n[A*] = [a,b]EX_m^n$ *iff A is recursive.*

iii. $[a,b]EX^n[A] = [a,b]EX^n$ *iff* $[a,b]EX^*[A] = [a,b]EX^*$ *iff* $\mathcal{G}(A)$.

iv. $[a,b]EX^n[A*] = [a,b]EX^n$ *iff* $[a,b]EX^*[A*] = [a,b]EX^*$ *iff* $[a,b]BC^n[A*] = [a,b]BC^n$ *iff* $A \leq_{\mathrm{T}} K$.

Proof:
i) Clearly if A is recursive then all the equalities hold. Assume $[a,b]EX_m[A*] = [a,b]EX_m$ or $[a,b]EX_m[A] = [a,b]EX_m$. Clearly $EX_0[A[2b(m+1)-1]] \subseteq [a,b]EX_m[A*] = [a,b]EX_m$. By Lemma 6.42 we know $[a,b]EX_m[A*] \subseteq EX_{2b(m+1)-2}$. Hence $EX_0[A[2b(m+1)-1]] \subseteq EX_{2b(m+1)-2}$. By Lemma 6.34 A is recursive.

ii) Clearly if A is recursive then all the equalities hold. Assume A is not recursive. By part i of this Theorem there exists $S \in [a,b]EX_m[A*] - [a,b]EX_m$. By Fact 6.43 (parts v and vii) $\tilde{S} \in [a,b]EX_m[A*] - [a,b]EX_m^n$. Hence $[a,b]EX_m^n \subset [a,b]EX_m^n[A]$, $[a,b]EX_m^n \subset [a,b]EX_m^n[A*]$.

iii and iv) By a simple modification of the proofs of Theorems 6.3 and 6.4 we have the following.

$$\begin{array}{rl} \mathcal{G}(A) & \Rightarrow ([a,b]EX^n[A] = [a,b]EX^n \text{ and } [a,b]EX^*[A] = [a,b]EX^*). \\ A \le_{\mathrm{T}} K & \Rightarrow ([a,b]EX^n[A*] = [a,b]EX^n \text{ and } [a,b]EX^*[A*] = [a,b]EX^* \\ & \quad \text{and } [a,b]BC^n[A*] = [a,b]BC^n). \end{array}$$

We show that

$$\begin{array}{rl} \neg\mathcal{G}(A) & \Rightarrow ([a,b]EX^n[A] \ne [a,b]EX^n \text{ and } [a,b]EX^*[A] \ne [a,b]EX^*). \\ A \not\le_{\mathrm{T}} K & \Rightarrow ([a,b]EX^n[A*] \ne [a,b]EX^n \text{ and } [a,b]EX^*[A*] \ne [a,b]EX^* \\ & \quad \text{and } [a,b]BC^n[A*] \ne [a,b]BC^n). \end{array}$$

If $\mathcal{G}(A)$ does not hold then by Theorem 6.3 there exists $S \in [a,b]EX[A] - [a,b]EX$. By Fact 6.43 (parts iii, $viii$) $\tilde{S} \in [a,b]EX[A] - [a,b]EX^*$. Hence $[a,b]EX^*[A] \ne [a,b]EX^*$ and $[a,b]EX^n[A] \ne [a,b]EX^n$. The proof for $A \not\le_{\mathrm{T}} K$ is similar (using Theorem 6.2 and Fact 6.43 (parts ii, iii). ∎

To prove results about when $[a,b]EX_m^*[A] = [a,b]EX_m^*$ requires a different technique than that used for $[a,b]EX_m^n$.

Lemma 6.47 *If $[a,b]EX_m[A[1]] \subseteq [a,b]EX_m^*$ then A is recursive.*

Proof: We will assume throughout the proof that a divides b. The modifications needed for the case where a does not divide b are easy.

Let $k = \frac{b}{a}(m+1) - 1$. Let

$$\begin{array}{rl} S = & \{e0^*1^*\cdots(p-1)^*p^\omega : e \in A \oplus \overline{A} \wedge 0 \le p \le k\} \cup \\ & \{e0^*1^*\cdots(p-1)^*p^\omega : e \notin A \oplus \overline{A} \wedge 1 \le p \le k+1\}. \end{array}$$

For $1 \le i \le \frac{b}{a}$ let

$$\begin{array}{rl} T_i = & \{0^*1^*\cdots(p-1)^*p^\omega : (i-1)m + i - 1 \le p \le im + i - 1\} \\ T_i' = & \{0^*1^*\cdots(p-1)^*p^\omega : (i-1)m + i \le p \le im + i\} \end{array}$$

Clearly for $1 \le i \le \frac{b}{a}$ both T_i and T'_i are in EX_m. If $f \in S$ then

$$\begin{array}{ll} f(0) \in A \oplus \overline{A} & \Rightarrow f \in \bigcup_{i=1}^{b/a} T_i, \\ f(0) \notin A \oplus \overline{A} & \Rightarrow f \in \bigcup_{i=1}^{b/a} T'_i. \end{array}$$

$S \in [a,b]EX_m[A[1]]$ by a team of b machines which consist of $\frac{b}{a}$ groups of a machines each where the ith group of machines are all machines that do the following: first ask '$f(0) \in A$?', and if YES then use an inference machine for T_i, else use an inference machine for T'_i.

Let $\tilde{S}$ be the cylindrification of S as in Definition 6.27. Let $\tilde{\sigma}(\langle x,y\rangle) = \sigma(x)$ for $x < |\sigma|$, $\langle x,y\rangle < \langle|\sigma|,0\rangle$ and $|\tilde{\sigma}| = \langle|\sigma|,0\rangle$.

Since $S \in [a,b]EX_m[A[1]]$, by Fact 6.43.vii $\tilde{S} \in [a,b]EX_m[A[1]]$. By the hypothesis $\tilde{S} \in [a,b]EX_m^*$. We assume $\tilde{S} \in [a,b]EX_m^*$ via $M_1,\cdots,M_b$. We can assume $(\forall i \ne j)(\forall \sigma,\tau)[M_i(\sigma) \ne M_j(\tau)]$ and that no M_i ever generates more than $m+1$ guesses. We show that A is recursive by showing that $A \oplus \overline{A}$ is r.e. Let $N(\tilde{\sigma})$ denote the total number of legal guesses ($\le b(m+1)$) that are made by $M_1,\ldots,M_b$ on the initial segment $\tilde{\sigma}$. We claim

$$e \in A \oplus \overline{A} \text{ iff } (\exists\sigma)[\sigma \in e0^*1^*\cdots k^* \wedge N(\tilde{\sigma}) = a(k+1)].$$

Assume $e \in A \oplus \overline{A}$. Let $U_k = \{e0^*1^*\cdots(p-1)^*p^\omega : 0 \le p \le k\}$. Since $U_k \subseteq S$ we have $\tilde{U_k} \subseteq \tilde{S}$, so $\tilde{U_k}$ is $[a,b]EX_m^*$-inferred by the team. By a modification of the proof of Lemma 6.32.ii there exists $\sigma \in e0^*\cdots k^*$ such that $N(\tilde{\sigma}) = a(k+1)$.

Assume there is a $\sigma \in e0^*1^*2^*\cdots k^*$ such that $N(\tilde{\sigma}) = a(k+1) = b(m+1)$. Then if $\tilde{\sigma}$ is fed to $M_1,\ldots,M_b$ there are $b(m+1)$ guesses generated, hence all b machines generate $m+1$ guesses. Therefore for all $\tilde{f}$ such that $\tilde{\sigma} \preceq \tilde{f}$, for all i, $1 \le i \le b$, $M_i(\tilde{\sigma})$ is the final guess that M_i outputs when trying to infer $\tilde{f}$. Consider the cylindrification of $V = \{\sigma k^i(k+1)^\omega : i \in \mathbb{N}\}$. For every function $\tilde{f} \in \tilde{V}$, we have $\tilde{\sigma} \preceq \tilde{f}$. Also note that there are an infinite number of functions in $\tilde{V}$ that are $=^*$ inequivalent. Hence there exists $\tilde{f} \in \tilde{V}$ such that $\tilde{f}$ is not EX^*-inferred by any of $M_1,\ldots,M_b$. Hence $\tilde{f} \notin \tilde{S}$, so $f \notin S$. Since $f \in V$, $f \in e0^*1^*\ldots k^*(k+1)^\omega$. Since $f \notin S$ we know that $e \notin A \oplus \overline{A}$. ∎

Corollary 6.48 *If* $[a,b]EX_m^*[A] = [a,b]EX_m^*$ *or* $[a,b]EX_m^*[A*] = [a,b]EX_m^*$ *then* A *is recursive*

7 Conclusions and Open Problems

The table below tells, for several notions of inference, when a set A will be trivial, $*$-trivial, omniscient, and $*$-omniscient (e.g., if the notion is EX then the table tells, for which A, $EX[A] = EX$, $EX[A*] = EX$, $REC \in EX[A]$, and $REC \in EX[A*]$). The entries that are unknown are marked with subscripted $U(A)$. If the same subscript is used in two places then we know the entries are the same, though we do not know what they are. All the results in the table are either from [SS91], [KS93b], [AB91], or this paper.

We have several open questions that we state as conjectures.

i. $REC \in BC[A]$ iff $REC \in [a,b]BC[A]$.

ii. $BC^n[A] = BC^n$ iff $\mathcal{G}(A)$.

iii. $[a,b]BC^n[A] = [a,b]BC^n$ iff $BC^n[A] = BC$.

iv. If A is n-r.e. then $REC \in BC[A]$ iff $\emptyset'' \leq_{\mathrm{T}} A'$. (Open for $n \geq 2$.)

v. If A is n-r.e. then $REC \in [a,b]BC[A]$ iff $\emptyset'' \leq_{\mathrm{T}} A'$. (Open for $n \geq 2$.)

There have been may variations on EX and BC in the inductive inference literature. Questions about the inference degrees that these definitions induce could be addressed. We discuss one such variation. Gasarch and Smith [GS92] have defined inference classes that allow the machine to ask questions about the function in a language L, denoted $QEX[L]$, $QBC[L]$, $Q_iEX[L]$, and $Q_iBC[L]$ (where i bounds the number of alternations of quantifiers allowed.) It would be of interest know (1) for which A does $QEX[S,<][A] = QEX[A]$ hold, and (2) for which A does $REC \in QEX[S,<][A]$ hold?

	TRIV	$*$-TRIV	OMNI	$*$-OMNI
PEX	$U_1(A)$	$U_1(A)$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
EX_m	$A \equiv_{\mathrm{T}} \emptyset$	$A \equiv_{\mathrm{T}} \emptyset$	Never	Never
EX	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
EX^n	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
EX^*	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
BC	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$U_2(A)$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
BC^n	$U_3(A)$	$A \leq_{\mathrm{T}} K$	$U_2(A)$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
BC^*	Always	Always	Always	Always
$[a,b]EX_m^n$	$A \equiv_{\mathrm{T}} \emptyset$	$A \equiv_{\mathrm{T}} \emptyset$	Never	Never
$[a,b]EX_m^*$	$A \equiv_{\mathrm{T}} \emptyset$	$A \equiv_{\mathrm{T}} \emptyset$	Never	Never
$[a,b]EX$	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
$[a,b]EX^n$	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
$[a,b]EX^*$	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$\emptyset'' \leq_{\mathrm{T}} A'$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
$[a,b]BC$	$\mathcal{G}(A)$	$A \leq_{\mathrm{T}} K$	$U_5(A)$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$
$[a,b]BC^n$	$U_4(A)$	$A \leq_{\mathrm{T}} K$	$U_5(A)$	$\emptyset'' \leq_{\mathrm{T}} A \oplus K$

$\mathcal{G}(A)$ means that either A is recursive or $A \leq_{\mathrm{T}} K$ and A is in a 1-generic degree. U_4 and U_5 may depend on the parameters a, b, n.

We know more than what is in the table:

i. If $A \leq_{\mathrm{T}} K$ or A is in a hyperimmune-free degree then $U_1(A)$ holds.

ii. There are low sets A such that $U_2(A)$ and $U_5(A)$ hold.

iii. For all sets X there exists a set A such that $X \leq_{\mathrm{T}} A''$ but neither $U_2(A)$ nor $U_5(A)$ holds.

iv. If A is r.e. then $U_2(A)$ iff $U_4(A)$ iff $\emptyset'' \leq_{\mathrm{T}} A'$.

v. $U_3(A)$ implies $U_4(A)$, and $U_4(A)$ implies $\mathcal{G}(A)$.

8 Acknowledgments

We would like to thank Steve Fenner, John Guthrie, Tamara Hummel, Georgia Martin, and Arun Sharma for proofreading and commentary.

References

[AB91] Lenny Adleman and Manuel Blum. Inductive inference and unsolvability. *Journal of Symbolic Logic*, 56(3):891–900, September 1991.

[Ang88] Dana Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

[BB75] Lenore Blum and Manuel Blum. Towards a mathematical theory of inductive inference. *Information and Control*, 28:125–155, 1975.

[BGGO93] Richard Beigel, William I. Gasarch, John T. Gill, and James C. Owings. Terse, superterse, and verbose sets. *Information and Computation*, 103:68–85, 1993.

[CDF+92] Peter Cholak, Rodney Downey, Lance Fortnow, William Gasarch, Efim Kinber, Martin Kummer, Stuart Kurtz, and Theodore Slaman. Degrees of inferability. In *Fifth Annual Conference on Computational Learning Theory*, pages 180–192. ACM, 1992.

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

[EHK81] Richard L. Epstein, Richard Haas, and Richard L. Kramer. Hierarchies of sets and degrees below $\mathbf{0}'$. In *Logic Year 1979–80*, volume 859 of *Lecture Notes in Mathematics*, pages 32–48, Berlin, 1981. Springer-Verlag.

[FSV89] Rusins Freivalds, Carl Smith, and Mahendra Velauthapillai. Trade-off among parameters affecting inductive inference. *Information and Computation*, 82(3):323–349, September 1989.

[GJPS91] William Gasarch, Sanjay Jain, Mark Pleszkoch, and Robert Solovay. Learning via queries to an oracle, 1991. Manuscript. The results of this Manuscript have been subsumed by *Extremes in the Degrees of Inferability*.

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

[GP89] William Gasarch and Mark Pleszkoch. Learning via queries to an oracle. In *Second Annual Conference on Computational Learning Theory*, pages 214–229. Morgan Kaufman, 1989.

[GS92] William Gasarch and Carl H. Smith. Learning via queries. *Journal of the ACM*, 39(3):649–675, July 1992. A shorter version is in 29th FOCS conference, 1988, pp. 130-137.

[Joc72] Carl G. Jockusch. Degrees in which recursive sets are uniformly recursive. *Canadian Journal of Mathematics*, 24:1092–1099, 1972.

[Joc80] Carl G. Jockusch. Degrees of generic sets. In F.R. Drake and S.S. Wainer, editors, *Recursion Theory: its generalizations and applications*, pages 110–139. Cambridge University Press, 1980.

[JS] Sanjay Jain and Arun Sharma. On the non-existence of maximal inference degrees for language identification. *Information Processing Letters*. to appear.

[Kin90] Efim Kinber. Some problems of learning with an oracle. In *Third Annual Conference on Computational Learning Theory*, pages 176–186. Morgan Kaufman, 1990.

[KS93a] Martin Kummer and Frank Stephan. Effective search problems. 1993. submitted.

[KS93b] Martin Kummer and Frank Stephan. On the structure of the degrees of inferability. In *Sixth Annual Conference on Computational Learning Theory*, 1993.

[Kum92] Martin Kummer. A proof of Beigel's cardinality conjecuture. *Journal of Symbolic Logic*, 57(2):677–681, June 1992.

[PS88] Lenny Pitt and Carl Smith. Probability and plurality for aggregations of learning machines. *Information and Computation*, 77:77–92, 1988.

[Smi82] Carl Smith. The power of pluralism for automatic program synthesis. *Journal of the ACM*, 29(4):1144–1165, october 1982.

[Soa87] Robert I. Soare. *Recursively Enumerable Sets and Degrees*. Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987.

[SS91] Theodore Slaman and Robert Solovay. When oracles do not help. In *Fourth Annual Conference on Computational Learning Theory*, pages 379–383. Morgan Kaufman, 1991.

[Val84] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27:1134–1142, 1984.