Robust Behaviourally Correct Learning

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

Abstract

Intuitively, a class of functions is robustly learnable if not only the class itself, but also all of the transformations of the class under natural transformations (such as via general recursive operators) are learnable.

Fulk [Ful90] showed the existence of a non-trivial class which is robustly learnable under the criterion **Ex**. However, several of the hierarchies (such as the anomaly hierarchies for **Ex** and **Bc**) do not stand robustly. Fulk left open the question about whether **Bc** and **Ex** can be robustly separated. In this paper we resolve this question positively.

1 Introduction

The learning situation often studied in inductive inference may be described as follows. A learner receives as input a graph of a function f, one element at a time. As the learner is receiving its input, it conjectures a sequence of programs as hypotheses. To be able to learn the function f, the sequence of programs conjectured by the learner must have some desirable property with respect to the input function f. By appropriately choosing this desirable property one gets different criteria of learning. One of the first such criteria studied is called **Ex**-identification ([Gol67]). The learner is said to **Ex**-identify f iff the sequence of programs output by it converges to a program for f (see the formal definitions in Section 2). A learner is said to **Ex**-identify a class iff it **Ex**-identifies each function in the class. A class of functions is **Ex**-identifiable iff some machine **Ex**-identifies the class.

Even though one cannot **Ex**-identify the class of all recursive functions, several interesting and important classes, such as the class of polynomials, are **Ex**-identifiable. Gold [Gol67] showed that one can **Ex**-identify every recursively enumerable class of recursive functions using the following technique. Suppose $\mathcal{C}$ is a recursively enumerable class of recursive functions, and $p_0, p_1, \ldots$ is an effective sequence of programs which compute exactly the functions in $\mathcal{C}$. Consider a machine **M** which, on any input data, searches for the least i such that the function computed by program p_i is consistent with the input data; **M** then outputs p_i. It is easy to verify that **M** acting as above will **Ex**-identify each function in $\mathcal{C}$. The above technique is known as identification by enumeration. The naturalness of this strategy led Gold to conjecture that *any* class of functions which can be **Ex**-identified, can also be **Ex**-identified using identification by enumeration. That is, every **Ex**-identifiable class is contained in a recursively enumerable class of functions. Bārzdiņš [Bar71] showed the above conjecture to be false using the "self-describing" class, SD $= \{f \mid f(0)$ is a program for $f\}$. A machine can **Ex**-identify each function

f in SD by just outputting the program $f(0)$. On the other hand, no recursively enumerable class of recursive functions contains SD. In defense of Gold's intuitions, SD and other self-referential classes used to refute his conjecture seem like artificial tricks. After all, to identify these self-referential classes, a learning device need only find some coded value from its input. On the other hand, one could argue that self-description is quite natural in that every cell of every organism contains a "program" that completely describes that organism. The reader is directed to [JSW98] for further discussion on this issue. Some of the following motiviation is from [JSW98].

Bārzdiņš formulated a more sophisticated version of Gold's conjecture designed to transcend such counterexamples as above. He reasoned that if a class of functions is identifiable only by way of a self-referential property, then there would be a general recursive operator (i.e. an effective and total mapping from total functions to total functions) that would transform the class into an *unidentifiable* one. The idea is that if a learning device is able to find the embedded self-referential information in the elements of a class, so can a general recursive operator, which can then remove this information. To see this in the context of SD, consider the operator Θ which removes the self-referential information $f(0)$ as follows: $\Theta(f) = g$, where $g(x) = f(x+1)$. One can show that $\Theta(\text{SD}) = \{\Theta(f) \mid f \in \text{SD}\} = \mathcal{R}$, the class of all the recursive functions. Thus, $\Theta(\text{SD})$ is *not* **Ex**-identifiable [Gol67]. Informally stated, Bārzdiņš' conjecture is: If all the projections of a class of functions under all general recursive operators are **Ex**-identifiable (or, in other words, if the class is *robustly* **Ex**-identifiable), then the class is contained in a recursively enumerable class of recursive functions and, consequently, it is identifiable by enumeration. Fulk [Ful90] showed that Bārzdiņš' conjecture is false by exhibiting a class of functions which is robustly **Ex**-identifiable, but not contained in any recursively enumerable class of recursive functions. This result can be taken as the first non-trivial step to show that, in the model of inductive inference, robust learning may be really interesting.

Since Gold [Gol67] many criteria of inference have been proposed by researchers all over the world, see [AS83, BB75, CS83, Fre91, KW80, OSW86]. One such criterion is **Bc**-identification (cf. e.g. [CS83, Bar74]) informally described below (see formal definition in Section 2). In **Bc**-identification of a function f by machine $\mathbf{M}$ one requires that, $\mathbf{M}$ on receiving a graph of the function f, outputs an infinite sequence of programs p_0, p_1, ..., such that all but finitely many programs in this sequence are programs for f. Intuitively, **Bc**-identification requires semantic convergence rather than syntactic convergence. The criteria of inference, such as **Bc** above, that have been studied in literature have usually been accompanied by proofs showing the differences between the new and old criteria of inference. The proof techniques used to show separations between the criteria often involve classes with self-referential properties. The same criticism of the class SD above applies to such separations. Thus, it would be interesting to study whether these separations hold robustly. For example, Fulk [Ful90] showed that the anomaly hierarchies for **Ex** and **Bc**-identification (see formal definitions in Section 2) do not hold robustly. Hence, one may expect some results of inductive inference (especially the hierarchies) not to stand robustly, and it would be interesting to study which results do hold robustly.

Besides the issue of self-reference discussed above, it is also philosophically interesting to study robust learning [JSW98]. Herman Weyl [Wey52] described the famous Erlangen program on founding geometry algebraically due to Felix Klein as follows: "If you are to find deep properties of some object, consider all the natural transformations that preserve your object (i.e. under which the object remains invariant)." Since general recursive operators can be looked

upon as natural transformations, it is interesting to consider robust identification from a purely philosophical point of view too. We direct the reader to [JSW98] for further motivations for studying robust learning.

Ex and **Bc** are perhaps two of the most widely studied identification criteria. Thus, it is important to study whether these criteria are robustly separated or not. In this paper we show that **Bc** and **Ex**-identification are separated robustly.

In related work, [JSW98] presented several results on robust learning. For example, they showed that the mind change hierarchy for **Ex**-identification, and team-hierarchy for **Ex** and **Bc**-identification stands robustly. Furthermore, they even showed that there are 'self-referential' classes which can be robustly **Ex**-identified. [CJO$^+$98] addresses robust learning in the presence of context (see [AGS89, KSVW95]). Zeugmann [Zeu86] and Kurtz and Smith [KS89] have studied a slightly different version of robust identification. Recently Ott and Stephan [OS99] have studied a different version of robust identification (called hyper-robust identification) and independently obtained Corollary 2 of this paper.

We now proceed formally.

2 Notation and Preliminaries

Recursion-theoretic concepts not explained below are treated in [Rog67]. N denotes the set of natural numbers. $*$ denotes a non-member of N and is assumed to satisfy $(\forall n)[n < * < \infty]$. Let $\in, \subseteq, \subset, \supseteq, \supset$, respectively denote membership, subset, proper subset, superset and proper superset relations for sets. $\emptyset$ denotes the emptyset. $\mathrm{card}(S)$ denotes the cardinality of set S. So "$\mathrm{card}(S) \leq *$" means that $\mathrm{card}(S)$ is finite. $\min(S)$ and $\max(S)$, respectively, denote the minimum and maximum element in S. We take $\min(\emptyset)$ to be ∞ and $\max(\emptyset)$ to be 0.

$\langle \cdot, \cdot \rangle$ denotes a 1-1 computable mapping from pairs of natural numbers onto natural numbers. π_1, π_2 are the corresponding projection functions. $\langle \cdot, \cdot \rangle$ is extended to n-tuples in a natural way.

Λ denotes the empty function. η, with or without decorations, ranges over partial functions. $\eta(x)\downarrow$ denotes that $\eta(x)$ is defined. $\eta(x)\uparrow$ denotes that $\eta(x)$ is not defined. For $a \in N \cup \{*\}$, $\eta_1 =^a \eta_2$ means that $\mathrm{card}(\{x \mid \eta_1(x) \neq \eta_2(x)\}) \leq a$. $\eta_1 \neq^a \eta_2$ means that $\neg[\eta_1 =^a \eta_2]$. (If η_1 and η_2 are both undefined on input x, then, as is standard, we take $\eta_1(x) = \eta_2(x)$.) If $\eta =^a f$, then we often call a program for η as an a-error program for f. $\mathrm{domain}(\eta)$ and $\mathrm{range}(\eta)$ respectively denote the domain and range of the partial function η.

f, g and h, with or without decorations, range over total functions. $\mathcal{R}$ denotes the class of all *recursive* functions, i.e., total computable functions with arguments and values from N. $\mathcal{C}$ and $\mathcal{S}$, with or without decorations, range over subsets of $\mathcal{R}$. $\mathcal{P}$ denotes the class of all *partial recursive* functions. φ denotes a *fixed* acceptable programming system. φ_i denotes the partial computable function computed by program i in the φ-system. Note that in this paper all programs are interpreted with respect to the φ-system. We let Φ be an arbitrary Blum complexity measure [Blu67] associated with the acceptable programming system φ; many such measures exist for any acceptable programming system [Blu67]. For this paper, without loss of generality, we assume that $\Phi_i(x) \geq x$, for all i, x.

A class $\mathcal{C} \subseteq \mathcal{R}$ is said to be recursively enumerable (r.e.) iff there exists an r.e. set X such that $\mathcal{C} = \{\varphi_i \mid i \in X\}$. Zero is the everywhere 0 function, i.e., $\mathrm{Zero}(x) = 0$, for all $x \in N$.

2.1 Function Identification

We first describe inductive inference machines. We assume, without loss of generality, that the graph of a function is fed to a machine in canonical order. For any partial function η and $n \in N$ such that, for all $x < n$, $\eta(x)\downarrow$, we let $\eta[n]$ denote the finite initial segment $\{(x, \eta(x)) \mid x < n\}$. Clearly, $\eta[0]$ denotes the empty segment. SEG denotes the set of all finite initial segments, $\{f[n] \mid f \in \mathcal{R} \wedge n \in N\}$. We let σ and τ, with or without decorations, range over SEG. Let $|\sigma|$ denote the length of σ. We often identify (partial) functions with their graphs. Thus for example, for $\sigma = f[n]$ and for $x < n$, $\sigma(x)$ denotes $f(x)$. An *inductive inference machine* (IIM) [Gol67] is an algorithmic device that computes a mapping from SEG into $N \cup \{?\}$. Intuitively, "?" above denotes the case when the machine may not wish to make a conjecture. Although it is not necessary to consider learners that issue "?" for identification in the limit, it becomes useful when the number of mind changes a learner can make is bounded. In this paper, we assume, without loss of generality, that once an IIM has issued a conjecture on some initial segment of a function, it outputs a conjecture on all extensions of that initial segment. This is without loss of generality because a machine wishing to emit "?" after making a conjecture can instead be thought of as repeating its previous conjecture. We let $\mathbf{M}$, with or without decorations, range over learning machines. Since the set of all finite initial segments, SEG, can be coded onto N, we can view these machines as taking natural numbers as input and emitting natural numbers or ?'s as output. We say that $\mathbf{M}(f)$ converges to i (written: $\mathbf{M}(f)\downarrow = i$) iff $(\forall^\infty n)[\mathbf{M}(f[n]) = i]$; $\mathbf{M}(f)$ is undefined if no such i exists. The next definitions describe several criteria of function identification.

Definition 1 [Gol67, BB75, CS83] Let $a, b \in N \cup \{*\}$. Let $f \in \mathcal{R}$.

(a) $\mathbf{M}$ $\mathbf{Ex}^a_b$*-identifies* f (written: $f \in \mathbf{Ex}^a_b(\mathbf{M})$) just in case there exists an a-error program i for f such that $\mathbf{M}(f)\downarrow = i$ and $\mathrm{card}(\{n \mid ? \neq \mathbf{M}(f[n]) \neq \mathbf{M}(f[n+1])\}) \leq b$ (i.e., $\mathbf{M}$ makes no more than b mind changes on f).

(b) $\mathbf{M}$ $\mathbf{Ex}^a_b$*-identifies* $\mathcal{S}$ iff $\mathbf{M}$ $\mathbf{Ex}^a_b$-identifies each $f \in \mathcal{S}$.

(c) $\mathbf{Ex}^a_b = \{\mathcal{S} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{S} \subseteq \mathbf{Ex}^a_b(\mathbf{M})]\}$.

We often write $\mathbf{Ex}_b$ for $\mathbf{Ex}^0_b$, $\mathbf{Ex}^a$ for $\mathbf{Ex}^a_*$, and $\mathbf{Ex}$ for $\mathbf{Ex}^0_*$. $\mathbf{Ex}_0$ is also refered to as *finite* identification.

By definition of convergence, only finitely many data points from a function f had been observed by an IIM $\mathbf{M}$ at the (unknown) point of convergence. Hence, some form of learning must take place in order for $\mathbf{M}$ to learn f. For this reason, hereafter the terms *identify*, *learn* and *infer* are used interchangeably.

Definition 2 [Bar74, CS83] Let $a \in N \cup \{*\}$. Let $f \in \mathcal{R}$.

(a) $\mathbf{M}$ $\mathbf{Bc}^a$*-identifies* f (written: $f \in \mathbf{Bc}^a(\mathbf{M})$) iff, for all but finitely many $n \in N$, $\mathbf{M}(f[n])$ is an a-error program for f.

(b) $\mathbf{M}$ $\mathbf{Bc}^a$*-identifies* $\mathcal{S}$ iff $\mathbf{M}$ $\mathbf{Bc}^a$-identifies each $f \in \mathcal{S}$.

(c) $\mathbf{Bc}^a = \{\mathcal{S} \subseteq \mathcal{R} \mid (\exists \mathbf{M})[\mathcal{S} \subseteq \mathbf{Bc}^a(\mathbf{M})]\}$.

We often write $\mathbf{Bc}$ for $\mathbf{Bc}^0$.

Definition 3 [Gol67]

(a) A machine $\mathbf{M}$ is said to be *identifying by enumeration* iff there exists an effective sequence of programs $p_0, p_1, \ldots$, such that (i) each φ_{p_i} is total, and (ii) for all $\sigma \in \text{SEG}$, $\mathbf{M}(\sigma) = p_i$, where $i = \min(\{j \mid \sigma \subseteq \varphi_{p_j}\})$.

(b) $\mathbf{NUM} = \{\mathcal{C} \mid (\exists \mathcal{C}' \supseteq \mathcal{C})[\mathcal{C}' \text{ is recursively enumerable}]\}$.

Note that **NUM** is the collection of all the classes which can be identified using identification by enumeration, [Gol67].

Some relationships between the above criteria are summarized in the following theorem.

Theorem 1 [CS83, BB75, Bar71]

(a) $\mathbf{Ex}^0 \subset \mathbf{Ex}^1 \subset \cdots \subset \mathbf{Ex}^* \subset \mathbf{Bc} \subset \mathbf{Bc}^1 \subset \cdots \subset \mathbf{Bc}^* = 2^{\mathcal{R}}$.

(b) Let $a, b, c, d \in N \cup \{*\}$. *Then,* $\mathbf{Ex}^a_b \subseteq \mathbf{Ex}^c_d$ *iff* $a \leq c$ *and* $b \leq d$.

(c) $\mathbf{NUM} \subseteq \mathbf{Ex}$.

We let $\mathbf{I}$ and $\mathbf{J}$ range over identification criteria defined above.

There exists an r.e. sequence $\mathbf{M}_0, \mathbf{M}_1, \mathbf{M}_2, \ldots$, of inductive inference machines such that, for all criteria $\mathbf{I}$ of inference considered in this paper,

for all $\mathcal{C} \in \mathbf{I}$, there exists an $i \in N$ such that $\mathcal{C} \subseteq \mathbf{I}(\mathbf{M}_i)$.

[OSW86] shows the above for $\mathbf{I} = \mathbf{Ex}$. Essentially, the same proof can be used for all $\mathbf{I}$ considered in this paper. We assume $\mathbf{M}_0, \mathbf{M}_1, \mathbf{M}_2, \ldots$ to be one such sequence of machines.

2.2 Operators

Definition 4 [Rog67] A *recursive operator* is an effective total mapping, Θ, from (possibly partial) functions to (possibly partial) functions, which satisfies the following properties:

(a) Monotonicity: For all functions η, η', if $\eta \subseteq \eta'$ then $\Theta(\eta) \subseteq \Theta(\eta')$.

(b) Compactness: For all η, if $(x, y) \in \Theta(\eta)$, then there exists a finite function $\alpha \subseteq \eta$ such that $(x, y) \in \Theta(\alpha)$.

(c) Recursiveness: For all finite functions α, one can effectively enumerate (in α) all $(x, y) \in \Theta(\alpha)$.

Definition 5 [Rog67] A recursive operator Θ is called *general recursive* iff Θ maps all total functions to total functions.

Remark 1 *For each recursive operator* Θ, *we can effectively (from* Θ*) find a recursive operator* Θ' *such that,*

(a) for each finite function α, $\Theta'(\alpha)$ *is finite, and its canonical index can be effectively determined from* α, *and*

(b) for all total functions f, $\Theta'(f) = \Theta(f)$.

Definition 6 [JSW98] Let $\mathbf{I}$, $\mathbf{J}$ be identification criteria.
$(\mathbf{I}, \mathbf{J})$-*robust* $= \{\mathcal{C} \mid \mathcal{C} \in \mathbf{I} \ \wedge \ (\forall \text{ general recursive operators } \Theta)[\Theta(\mathcal{C}) \in \mathbf{J}]\}$.

Note that traditionally only $(\mathbf{I},\mathbf{I})$-robust identification is considered and referred to as robust $\mathbf{I}$-identification. The above definition is a generalization of this notion. The reason we consider such a generalization is that there are classes which are not in $(\mathbf{I},\mathbf{I})$-robust, but they are in $(\mathbf{I},\mathbf{J})$-robust for $\mathbf{J}$ a weaker identification criterion than $\mathbf{I}$, i.e. $\mathbf{I} \subset \mathbf{J}$. Alternatively, one may interpret a positive result on $(\mathbf{I},\mathbf{J})$-robustness as "how simple" (namely, even from $\mathbf{I}$) a robustly $\mathbf{J}$-identifiable class can be. Also, as seen by the following proposition, we always have $(\mathbf{I},\mathbf{J})$-robust as a subset of $(\mathbf{J},\mathbf{J})$-robust.

Proposition 1 *[JSW98]*
(a) Suppose $\mathbf{I} \subseteq \mathbf{I}'$, $\mathbf{J} \subseteq \mathbf{J}'$. *Then* $(\mathbf{I},\mathbf{J})$*-robust* $\subseteq (\mathbf{I}',\mathbf{J}')$*-robust.*
(b) $(\mathbf{I},\mathbf{J})$*-robust* $= (\mathbf{I}\cap\mathbf{J},\mathbf{J})$*-robust.*

Barzdin had conjectured that every class $\mathcal{C}$ in $(\mathbf{Ex},\mathbf{Ex})$-robust is in **NUM**, that is $\mathcal{C}$ is contained in a recursively enumerable class. Fulk refuted this conjecture by constructing a class $\mathcal{C} \in (\mathbf{Ex},\mathbf{Ex})$-robust which is not in **NUM**. Fulk also showed that:

Theorem 2 *[Ful90]*
(a) For all $a \in N \cup \{*\}$, $(\mathbf{Ex}^a,\mathbf{Ex}^a)$*-robust* $= (\mathbf{Ex},\mathbf{Ex})$*-robust.*
(b) For all $n \in N$, $(\mathbf{Bc}^n,\mathbf{Bc}^n)$*-robust* $= (\mathbf{Bc},\mathbf{Bc})$*-robust.*

Since, $\mathcal{R} \in \mathbf{Bc}^*$, it follows that $\mathcal{R} \in (\mathbf{Bc}^*,\mathbf{Bc}^*)$-robust. Thus, for $a \in N \cup \{*\}$ and $n \in N$,
$(\mathbf{Ex},\mathbf{Ex})$-robust$= (\mathbf{Ex}^a,\mathbf{Ex}^a)$-robust$\subseteq (\mathbf{Bc},\mathbf{Bc})$-robust$= (\mathbf{Bc}^n,\mathbf{Bc}^n)$-robust$\subset (\mathbf{Bc}^*,\mathbf{Bc}^*)$-robust.

It was left open by Fulk whether $(\mathbf{Ex},\mathbf{Ex})$-robust $\subset (\mathbf{Bc},\mathbf{Bc})$-robust. We solve this question in this paper.

3 Main Result

We first show the following Lemma.

Lemma 1 *There exists a recursive function* p *such that, for all* $i \in N$, *following five properties are satisfied:*
(A) $(\forall x < i)[\varphi_{p(i)}(x) = 0]$.
(B) $\varphi_{p(i)}(i) = 1$.
(C) $\mathrm{card}(\{x \mid \varphi_{p(i)}(x)\!\uparrow\} \leq 1)$.
(D) $\mathrm{range}(\varphi_{p(i)}) \subseteq \{0,1\}$.
(E) For $z \in \{0,1\}$, *let*

$$h_i^z(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } \varphi_{p(i)}(x)\!\downarrow; \\ z, & \text{otherwise.} \end{cases}$$

Then $\{h_i^0, h_i^1\} \not\subseteq \mathbf{Ex}(\mathbf{M}_i)$.

Proof. This proof is based on a modification of proof of $\mathbf{Ex}^1 - \mathbf{Ex} \neq \emptyset$ in [CS83]. By operator recursion theorem [Cas74] there exists a recursive p such that $\varphi_{p(i)}$ may be defined in stages as follows.

Initially, for $x < i$, $\varphi_{p(i)}(x) = 0$, and $\varphi_{p(i)}(i) = 1$. Let $x_0 = i+1$. Intuitively, x_s denotes the least x such that $\varphi_{p(i)}(x)$ has not been defined before stage s. Go to stage 0.

Stage s

1. For $z \in \{0,1\}$, let
$$f_z(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } x < x_s; \\ z, & \text{if } x = x_s; \\ 0, & \text{otherwise.} \end{cases}$$
2. For $x = x_s + 1$ to ∞ Do
   2.1. If $\mathbf{M}_i(f_0[x]) \neq \mathbf{M}_i(f_1[x])$, then go to step 3.
   2.2. Else let $\varphi_{p(i)}(x) = 0$.
   EndFor
3. Let $z \in \{0,1\}$ be such that $\mathbf{M}_i(f_z[x]) \neq \mathbf{M}_i(\varphi_{p(i)}[x_s])$.
   Set $\varphi_{p(i)}(x_s) = z$, and let $x_{s+1} = x$.
   Go to stage $s+1$.

End stage s

Fix i. It is easy to verify that (A) to (D) of lemma are satisfied. We show (E). We consider two cases:

Case 1: There exist infinitely many stages.

In this case $h_i^0 = h_i^1 = \varphi_{p(i)}$, and $\mathbf{M}_i$ on $\varphi_{p(i)}$ makes infinitely many mind changes (due to execution of step 3 infinitely often).

Case 2: Stage s starts but does not finish.

In this case $\varphi_{p(i)}$ is not defined on exactly one point x_s. Also, for all $x > x_s$, $\mathbf{M}_i(h_i^0[x]) = \mathbf{M}_i(h_i^1[x])$. Thus $\mathbf{M}_i$ fails to $\mathbf{Ex}$-identify at least one of h_i^0 and h_i^1.

From the above cases, it follows that (E) holds. ∎

Theorem 3 *There exists a $\mathcal{C} \subseteq \mathcal{R}$ such that $\mathcal{C} \in (\mathbf{Ex}^1 - \mathbf{Ex}, \mathbf{Bc})$-robust.*

Proof. Let p be as given by Lemma 1. For $z \in \{0,1\}$, let
$$h_i^z(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } \varphi_{p(i)}(x)\downarrow; \\ z, & \text{otherwise.} \end{cases}$$

Let Zero denote the everywhere zero function.

Let $\mathcal{C} = \{\text{Zero}\} \cup \{h_i^z \mid z \in \{0,1\} \ \wedge \ i \in N\}$. It is easy to verify, using Lemma 1, that $\mathcal{C} \in \mathbf{Ex}^1 - \mathbf{Ex}$. Fix a general recursive operator Θ. We assume without loss of generality (by Remark 1) that for all f, n, $\Theta(f[n])$ is finite and a (canonical index) for $\Theta(f[n])$ can be obtained effectively from $f[n]$. We will show below that $\Theta(\mathcal{C}) \in \mathbf{Bc}$.

For all $i, k \in N$ and $z \in \{0,1\}$, let
$$g_{i,k}^z(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } x \neq k; \\ z, & \text{if } x = k. \end{cases}$$

Let $progp(i)$ denote a program (obtained effectively from i) for $\Theta(\varphi_{p(i)})$. Let $progq(i,k,z)$ be a program (obtained effectively from i,k,z) for $\Theta(g_{i,k}^z)$. Let $progr(i,k)$ denote a program (obtained effectively from i,k) such that $\varphi_{progr(i,k)}$ is defined as follows:

$$\varphi_{progr(i,k)}(x) = \begin{cases} \varphi_{progp(i)}(x), & \text{if } \varphi_{progp(i)}(x)\downarrow; \\ \varphi_{progq(i,k,0)}(x), & \text{if } \varphi_{progq(i,k,0)}(x)\downarrow = \varphi_{progq(i,k,1)}(x)\downarrow; \\ \uparrow, & \text{otherwise} \end{cases}$$

Note that the second clause above is consistent with the first clause, since $\varphi_{p(i)}$ is a subset of at least one of $g^0_{i,k}$ and $g^1_{i,k}$.

Define $\mathbf{M}$ as follows.

$\mathbf{M}(f[n])$

1. If $f[n]$ is consistent with $\Theta(\text{Zero}[n])$, then output a program for $\Theta(\text{Zero}[n])$.
2. Otherwise, let m be the least number such that $f[n]$ is inconsistent with $\Theta(\text{Zero}[m])$.
3. For each $i < m$, let $k_i = \min(\{x \mid x > i \ \wedge \ \Phi_{p(i)}(x) > n\})$.
   For each $i < m$, let $u_i = \min(\{x \mid x > k_i \ \wedge \ \Phi_{p(i)}(x) > n\})$.
   (* Note that $\min(\{x \mid x > i \ \wedge \ \Phi_{p(i)}(x) > n\})$, and $\min(\{x \mid x > k_i \ \wedge \ \Phi_{p(i)}(x) > n\})$ are non-empty, since by our assumption on Φ, $\Phi_{p(i)}(x) \geq x$. *)
4. (* Note that, for all $x < u_i$, g^z_{i,k_i} is defined. *)
   Let $i < m$ be the least number such that, for some $z \in \{0,1\}$, $\Theta(g^z_{i,k_i}[u_i])$ is consistent with $f[n]$. (If no such i exists, then output 0).
5. If both $\Theta(g^0_{i,k_i}[u_i])$ and $\Theta(g^1_{i,k_i}[u_i])$ are consistent with $f[n]$, then output $progr(i,k_i)$.
   Else output $progq(i,k_i,z)$, where $\Theta(g^z_{i,k_i}[u_i])$ is consistent with $f[n]$.

End

We claim that the above $\mathbf{M}$ $\mathbf{Bc}$-identifies $\Theta(\mathcal{C})$. Suppose $f \in \Theta(\mathcal{C})$. If $f = \Theta(\text{Zero})$, then clearly, $\mathbf{M}$ $\mathbf{Bc}$-identifies f. So suppose $f \neq \Theta(\text{Zero})$. Let m_0 be the least number such that $\Theta(\text{Zero}[m_0])$ is inconsistent with f. Then, clearly, for large enough n, m as computed in step 2 of $\mathbf{M}(f[n])$ will be m_0. Let i_0 be the least number such that, for some $z_0 \in \{0,1\}$, $f = \Theta(h^{z_0}_{i_0})$. Note that $i_0 < m_0$. Let $n_0 > m_0$ be large enough so that, for all $n \geq n_0$, the following hold (note that there exists such an n_0):

(A) $f[n]$ is inconsistent with $\Theta(\text{Zero}[m_0])$,

(B) For $i < m_0$, let k_i, u_i be as computed in step 3 of $\mathbf{M}(f[n])$. Then, the following three hold:

(B.1) For all $i < i_0$, and $z \in \{0,1\}$, $\Theta(g^z_{i,k_i}[u_i])$ is inconsistent with $f[n]$,

(B.2) If $\varphi_{p(i_0)} \notin \mathcal{R}$, then $k_{i_0} = \min(\{x \mid \varphi_{p(i_0)}(x)\uparrow\})$,

(B.3) For $z \in \{0,1\}$, if $\Theta(h^z_{i_0})$ is inconsistent with f, then $\Theta(h^z_{i_0}[u_{i_0}])$ is inconsistent with $f[n]$.

Now, it is easy to verify that, for all $n > n_0$, i as computed in step 4 is the same as i_0. We now claim that, for $n > n_0$, $\mathbf{M}(f[n])$ is a program for f.

Fix $n > n_0$. Let k_{i_0}, u_{i_0} be as computed in step 3 of $\mathbf{M}(f[n])$. Note that if $\varphi_{p(i_0)} \notin \mathcal{R}$, then $k_{i_0} = \min(\{x \mid \varphi_{p(i_0)}(x)\uparrow\})$. We now consider the following two cases:

Case 1: Both $\Theta(g^0_{i_0,k_{i_0}}[u_{i_0}])$ and $\Theta(g^1_{i_0,k_{i_0}}[u_{i_0}])$, are consistent with $f[n]$.

Note that, in this case, our assumption on n_0 implies that both $\Theta(h^0_{i_0})$ and $\Theta(h^1_{i_0})$ are consistent with f.

Case 1a: $\varphi_{p(i_0)}$ is not total.

In this case, our assumption on n_0 implies that $k_{i_0} = \min(\{x \mid \varphi_{p(i_0)}(x)\uparrow\})$. Thus, $h^z_{i_0} = g^z_{i_0,k_{i_0}}$. Hence, by definition of $progr(i_0, k_0)$, for all $z \in \{0,1\}$, $\varphi_{progr(i_0,k_{i_0})} = \Theta(h^z_{i_0}) = \Theta(g^z_{i_0,k_{i_0}})$.

Case 1b: $\varphi_{p(i_0)}$ is total.

In this case, $h^0_{i_0} = h^1_{i_0}$, and thus $\varphi_{progr(i_0,k_{i_0})} = \varphi_{progp(i_0)} = \Theta(h^0_{i_0}) = \Theta(h^1_{i_0}) = f$.

Case 2: For some $z \in \{0,1\}$, $\Theta(g^z_{i_0,k_{i_0}}[u_{i_0}])$ is consistent with f but $\Theta(g^{1-z}_{i_0,k_{i_0}}[u_{i_0}])$ is inconsistent with f.

In this case, $h^z_{i_0} = g^z_{i_0,k_{i_0}}$, and $f = \Theta(g^z_{i_0,k_{i_0}})$. Thus, $\mathbf{M}(f[n]) = progq(i_0, k_{i_0}, z)$ is a program for f.

From the above cases, it follows that $\mathbf{M}(f[n])$ is a program for f. Thus, $\mathbf{M}$ **Bc**-identifies $\Theta(\mathcal{C})$. ∎

Corollary 1 *There exists a $\mathcal{C}' \subseteq \mathcal{R}$ such that $\mathcal{C}' \in (\mathbf{Ex}^1_0 - \mathbf{Ex}, \mathbf{Bc})$-robust.*

Proof. Let $\mathcal{C}$ be as in the proof of Theorem 3. Let $\mathcal{C}' = \mathcal{C} - \{\text{Zero}\}$. One can now easily verify that $\mathcal{C}' \in \mathbf{Ex}^1_0 - \mathbf{Ex}$. The corollary now follows from Theorem 3. ∎

Corollary 2 $(\mathbf{Bc}, \mathbf{Bc})$*-robust* $-\mathbf{Ex} \neq \emptyset$.

The above solves an open problem from [Ful90].

Corollary 3 *For all $n \in N$, there exists a $\mathcal{S} \subseteq \mathcal{R}$ such that $\mathcal{S} \in (\mathbf{Ex}^{n+1}_0 - \mathbf{Ex}^n, \mathbf{Bc})$-robust.*

Proof. For $f \in \mathcal{R}$, let f' be defined as follows:

$$f'(\langle x, y\rangle) = \begin{cases} f(x), & \text{if } y \leq n; \\ 0, & \text{otherwise.} \end{cases}$$

Suppose, $\mathcal{C} \in (\mathbf{Ex}^1_0 - \mathbf{Ex}, \mathbf{Bc})$-robust (Corollary 1 gives such a class). Let $\mathcal{S} = \{f' \mid f \in \mathcal{C}\}$. Since $\mathcal{C} \in \mathbf{Ex}^1_0$, it follows that $\mathcal{S} \in \mathbf{Ex}^{n+1}_0$. Also, $\mathcal{S} \in \mathbf{Ex}^n \Leftrightarrow \mathcal{S} \in \mathbf{Ex} \Leftrightarrow \mathcal{C} \in \mathbf{Ex}$. Thus, $\mathcal{S} \notin \mathbf{Ex}^n$.

Also, since there exists a general recursive operator Θ such that, $\Theta(\mathcal{C}) = \mathcal{S}$, it follows that $\mathcal{S} \in (\mathbf{Bc}, \mathbf{Bc})$-robust. The corollary follows. ∎

Corollary 4 *There exists a $\mathcal{S} \subseteq \mathcal{R}$ such that $\mathcal{S} \in (\mathbf{Bc} - \mathbf{Ex}^*, \mathbf{Bc})$-robust.*

Proof. For $f \in \mathcal{R}$, let f' be defined as follows: $f'(\langle x, y\rangle) = f(x)$. Suppose, $\mathcal{C} \in (\mathbf{Ex}^1 - \mathbf{Ex}, \mathbf{Bc})$-robust (Theorem 3 gives such a class). Let $\mathcal{S} = \{f' \mid f \in \mathcal{C}\}$. Now, $\mathcal{S} \in \mathbf{Ex}^* \Leftrightarrow \mathcal{S} \in \mathbf{Ex} \Leftrightarrow \mathcal{C} \in \mathbf{Ex}$. Thus, $\mathcal{S} \notin \mathbf{Ex}^*$.

Also, since there exists a general recursive operator Θ such that, $\Theta(\mathcal{C}) = \mathcal{S}$, it follows that $\mathcal{S} \in (\mathbf{Bc}, \mathbf{Bc})$-robust. The corollary follows. ∎

Upto now we have confined ourselves to *general recursive* operators for realizing the transformations of the classes to be learned. Clearly, this is not the most general approach. One could allow *recursive* operators instead, which map the functions in the class to be learned to total functions, but need not map functions outside the class to total functions.

Definition 7 $(\mathbf{I}, \mathbf{J})$-*recrobust* $= \{\mathcal{C} \mid \mathcal{C} \in \mathbf{I} \wedge (\forall \text{ recursive operators } \Theta \mid \Theta(\mathcal{C}) \subseteq \mathcal{R})[\Theta(\mathcal{C}) \in \mathbf{J}]\}$.

It is easy to verify that the proof of Theorem 3 also shows that

Theorem 4 *There exists a* $\mathcal{C} \subseteq \mathcal{R}$ *such that* $\mathcal{C} \in (\mathbf{Ex}_1^1 - \mathbf{Ex}, \mathbf{Bc})$-*recrobust.*

Further note that the proof of Theorem 3 shows that, for $\mathcal{C}$ as constructed in proof of Theorem 3, one can find a machine **Bc**-identifying $\Theta(\mathcal{C})$ effectively from a program for Θ. Thus not only one can **Bc**-identify $\Theta(\mathcal{C})$, for all general recursive operator Θ, but one can effectively find a program for **Bc**-identifying $\Theta(\mathcal{C})$, from a program for Θ. Similar observation also holds for Theorem 4, for recursive operators mapping $\mathcal{C}$ to a subset of $\mathcal{R}$.

References

[AGS89] D. Angluin, W. Gasarch, and C. Smith. Training sequences. *Theoretical Computer Science*, 66:255–272, 1989.

[AS83] D. Angluin and C. Smith. A survey of inductive inference: Theory and methods. *Computing Surveys*, 15:237–289, 1983.

[Bar71] J. Bārzdiņš. *Complexity and Frequency Solution of Some Algorithmically Unsolvable Problems.* PhD thesis, Novosibirsk State University, 1971. In Russian.

[Bar74] J. Bārzdiņš. Two theorems on the limiting synthesis of functions. In *Theory of Algorithms and Programs, vol. 1*, pages 82–88. Latvian State University, 1974. In Russian.

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

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

[Cas74] J. Case. Periodicity in generations of automata. *Mathematical Systems Theory*, 8:15–32, 1974.

[CJO+98] J. Case, S. Jain, M. Ott, A Sharma, and F. Stephan. Robustly learning aided by context. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 44–55. ACM Press, 1998.

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

[Fre91] R. Freivalds. Inductive inference of recursive functions: Qualitative theory. In J. Bārzdiņš and D. Bjorner, editors, *Baltic Computer Science*, volume 502 of *Lecture Notes in Computer Science*, pages 77–110. Springer-Verlag, 1991.

[Ful90] M. Fulk. Robust separations in inductive inference. In *31st Annual IEEE Symposium on Foundations of Computer Science*, pages 405–410. IEEE Computer Society Press, 1990.

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

[JSW98] Sanjay Jain, Carl Smith, and Rolf Wiehagen. On the power of learning robustly. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, pages 187–197. ACM Press, 1998.

[KS89] S. Kurtz and C. Smith. On the role of search for learning. In R. Rivest, D. Haussler, and M. Warmuth, editors, *Proceedings of the Second Annual Workshop on Computational Learning Theory*, pages 303–311. Morgan Kaufmann, 1989.

[KSVW95] E. Kinber, C. Smith, M. Velauthapillai, and R. Wiehagen. On learning multiple concepts in parallel. *Journal of Computer and System Sciences*, 50:41–52, 1995.

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

[OS99] Matthias Ott and Frank Stephan. Avoiding coding tricks by hyperrobust learning. In *Fourth European Conference on Computational Learning Theory*, volume 1572 of *Lecture Notes in Artificial Intelligence*, pages 183–197. Springer-Verlag, 1999.

[OSW86] D. Osherson, M. Stob, and S. Weinstein. *Systems that Learn: An Introduction to Learning Theory for Cognitive and Computer Scientists*. MIT Press, 1986.

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

[Wey52] H. Weyl. *Symmetry*. Princeton University Press, 1952.

[Zeu86] T. Zeugmann. On Bārzdiņš' conjecture. In K. Jantke, editor, *Analogical and Inductive Inference, Proceedings of the International Workshop*, volume 265 of *Lecture Notes in Computer Science*, pages 220–227. Springer-Verlag, 1986.