# On a Question of Nearly Minimal Identification of Functions

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

**Abstract**

Suppose $\mathcal{A}$ and $\mathcal{B}$ are classes of recursive functions. $\mathcal{A}$ is said to be an *$m$-cover* (*$*$-cover*) for $\mathcal{B}$, iff for each $g \in \mathcal{B}$, there exsits an $f \in \mathcal{A}$ such that $f$ differs from $g$ on at most $m$ inputs (finitely many inputs). $\mathcal{C}$, a class of recursive functions, is *$a$-immune* iff $\mathcal{C}$ is infinite and every recursively enumerable subclass of $\mathcal{C}$ has a finite $a$-cover. $\mathcal{C}$ is *$a$-isolated* iff $\mathcal{C}$ is finite or $a$-immune.

Chen [Che81] conjectured that every class of recursive functions that is $\mathbf{MEx}^*_m$-identifiable is $*$-isolated. We refute this conjecture.

## 1 Introduction

Formal definitions of notions informally discussed below are given in Section 3. Gold's [Gol67] criterion of identification of functions may be described as follows: A learning machine $\mathbf{M}$ is said to *identify* (or learn) a function $f$ just in case $\mathbf{M}$, when presented with the graph of $f$, outputs a sequence of programs that converges (in the limit) to a program for $f$. The above criterion of identification is called **Ex**-*identification* (**Ex** stands for explains). Freivalds [Fre75] (see also [Che81, Che82]) introduced the notion of nearly minimal identification, by placing an additional restriction on size of the final programs. In this criterion, the learning machine is required to converge to a program whose size is within a recursive factor of the size of the smallest program for the input function.

The above notions of identification can be extended in the following two directions:

- **Error Bound** ([BB75, CS83]): The above model may be relaxed by allowing the learning machine to converge to a program which may make some errors in computing the input function. An error bound of a natural number $m$ means that the final program makes at most $m$ errors in computing the input function. An error bound of $*$ means that the final program makes at most finitely many errors in computing the input function.

- **Mind-Change Bound** ([CS83, BF74]): The above model may be restricted by placing a bound on the number of mind changes allowed by the learning machine. A mind change bound of a natural number $m$ means that the learning machine may make at most $m$ mind changes before converging to the final program. A mind change bound of $*$ means that the learning machine may make at most finitely many mind changes before converging to the final program (note that $*$-mind change bound is equivalent to the Gold's notion of identification in the limit).

Chen [Che81] showed that the recursively enumerable (r.e.) classes of functions that can be identified in the nearly minimal sense with $m$-errors and with a mind change bound of $n$ (where $m$ and $n$ are natural numbers) are not very complex — they can be "approximated" with at most $m$-errors using a finite class of functions. For a recursively enumerable classes, this latter notion of being approximated with at most $m$-errors, by a finite class of functions, is referred to as being $m$-isolated. Chen [Che81] also showed that classes of functions which can be nearly-minimally-identified with $*$-errors, but with only 0-mind changes, are $*$-isolated. The question of $*$-errors, but with mind change bound of $n > 0$, was left open by Chen. He conjectured that such classes would also be $*$-isolated.

In this paper we refute Chen's conjecture. Thus complex r.e. classes can be identified in the nearly-minimal sense with $*$-errors and a nonzero mind change bound.

We now proceed formally.

## 2 Notation

Recursion-theoretic concepts not explained below are treated in [Rog67]. $N$ denotes the set of natural numbers, $\{0, 1, 2, \ldots\}$. All conventions regarding range of variables apply, with or without decorations[1], unless otherwise specified. The symbols $i$, $j$, $k$, $l$, $m$, $n$, $s$, $t, u$, $x$, $y$, and $z$, range over natural numbers unless otherwise specified. $\mathrm{card}(S)$ denotes the cardinality of a set $S$. $*$ denotes a nonmember of $N$ and is assumed to satisfy $(\forall n \in N)[n < * < \infty]$. Thus, $\mathrm{card}(S) \leq *$ means that cardinality of the set $S$ is finite. $a$ and $b$ range over $N \cup \{*\}$. $\max(\ ), \min(\ )$ denote the maximum and minimum of a set, respectively. By convention $\max(\emptyset) = 0$ and $\min(\emptyset) = \infty$.

$\mathcal{R}$ denotes the set of all total recursive functions. $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$, and $\mathcal{S}$ range over subsets of $\mathcal{R}$. $h, f$, and $g$ range over total recursive functions. $\eta$ ranges over partial functions. $\mathrm{domain}(\eta)$ denotes the domain of $\eta$. For $a \in N \cup \{*\}$, we say that $\eta_1 =^a \eta_2$ (read: $\eta_1$ is an $a$-variant of $\eta_2$) iff $\mathrm{card}(\{x \mid \eta_1(x) \neq \eta_2(x)\}) \leq a$. Thus, $\eta_1 =^* \eta_2$ means that $\eta_1$ and $\eta_2$ are finite variants of each other.

We let $\varphi$ denote a standard acceptable programming system. $\varphi_i$ denotes the partial recursive function computed by the $i^{th}$ program in the standard acceptable programming system $\varphi$. We often refer to the $i^{th}$ program as program $i$. $p$ ranges over total functions, with its range being interpreted as programs. For a recursive function $f$, $\mathrm{MinProg}(f)$ denotes the minimal program for $f$ (in the $\varphi$ system), i.e., $\mathrm{MinProg}(f) = \min(\{i \mid \varphi_i = f\})$.

A class $\mathcal{S}$ of recursive functions is said to be recursively enumerable iff there exists a recursive set $Z$ such that $\mathcal{S} = \{\varphi_i \mid i \in Z\}$.

$\langle i, j \rangle$ stands for an arbitrary computable one to one encoding of all pairs of natural numbers onto $N$ [Rog67].

The quantifiers '$\exists$', '$\forall$', '$\forall^\infty$', and '$\exists^\infty$' respectively denote 'there exists', 'for all', 'for all but finitely many', and 'there exist infinitely many'.

## 3 Learning Paradigms

For any partial function $\eta$ and any natural number $n$ such that, for each $x < n$, $\eta(x)\downarrow$, we let $\eta[n]$ denote the finite initial segment $\{(x, \eta(x)) \mid x < n\}$. Let $\mathrm{SEQ} = \{f[n] \mid f \in \mathcal{R} \wedge n \in N\}$.

[1] Decorations are subscripts, superscripts, primes and the like.

**Definition 1** [Gol67] A *learning machine* is an algorithmic device which computes a mapping from SEQ into $N \cup \{?\}$ such that, if $\mathbf{M}(f[n]) \neq ?$, then $\mathbf{M}(f[n+1]) \neq ?$.

We let $\mathbf{M}$, with or without decorations, range over learning machines. In Definition 1 above, '?' denotes the situation when $\mathbf{M}$ outputs "no conjecture" on some member of SEQ.

In Definition 2 below we spell out what it means for a learning machine to converge in the limit.

**Definition 2** Suppose $\mathbf{M}$ is a learning machine and $f$ is a computable function. $\mathbf{M}(f)\downarrow$ (read: $\mathbf{M}(f)$ *converges*) just in case $(\exists i)(\forall^\infty n)$ $[\mathbf{M}(f[n]) = i]$. If $\mathbf{M}(f)\downarrow$, then $\mathbf{M}(f)$ is defined $=$ the unique $i$ such that $(\forall^\infty n)[\mathbf{M}(f[n]) = i]$, otherwise we say that $\mathbf{M}(f)$ diverges (written: $\mathbf{M}(f)\uparrow$).

### 3.1 Explanatory Function Identification

We now formally define the criteria of inference considered in this paper.

**Definition 3** [Gol67, CS83, BB75, BF74] Suppose $a, b \in N \cup \{*\}$.

(1) A learning machine $\mathbf{M}$ is said to $\mathbf{Ex}_b^a$*-identify* $f \in \mathcal{R}$ (written: $f \in \mathbf{Ex}_b^a(\mathbf{M})$) just in case $(\exists i \mid \varphi_i =^a f)$ $(\forall^\infty n)[\mathbf{M}(f[n]) = i]$ and $\mathrm{card}(\{n \mid ? \neq \mathbf{M}(f[n]) \neq \mathbf{M}(f[n+1])\}) \leq b$.

(2) $\mathbf{Ex}_b^a = \{\mathcal{C} \mid (\exists \mathbf{M})[\mathcal{C} \subseteq \mathbf{Ex}_b^a(\mathbf{M})]\}$.

For a given $f$ and $\mathbf{M}$, we refer to each instance of the case, $? \neq \mathbf{M}(f[n]) \neq \mathbf{M}(f[n+1])$ as a *mind change* by $\mathbf{M}$ on $f$. Intuitively, in $\mathbf{Ex}_b^a$, the superscript $a$ refers to the error bound on the final program, and subscript $b$ refers to the mind change bound. We often refer to $\mathbf{Ex}_*^a$ as $\mathbf{Ex}^a$, $\mathbf{Ex}_b^0$ as $\mathbf{Ex}_b$ and $\mathbf{Ex}_*^0$ as $\mathbf{Ex}$.

### 3.2 Nearly Minimal Identification

We next consider nearly minimal identification criteria.

**Definition 4** [Fre75, Che82] Suppose $a, b \in N \cup \{*\}$.

(1) Suppose $h$ is a recursive function. A learning machine $\mathbf{M}$ is said to $h$-$\mathbf{MEx}_b^a$*-identify* $f \in \mathcal{R}$ (written $f \in h$-$\mathbf{MEx}_b^a(\mathbf{M})$) iff $\mathbf{M}$ $\mathbf{Ex}_b^a$-identifies $f$ and $\mathbf{M}(f) \leq h(\mathrm{MinProg}(f))$.

(2) $\mathbf{MEx}_b^a = \{\mathcal{C} \mid (\exists \mathbf{M})(\exists h \in \mathcal{R})[\mathcal{C} \subseteq h\text{-}\mathbf{MEx}_b^a(\mathbf{M})]\}$.

We often refer to $\mathbf{MEx}_*^a$ as $\mathbf{MEx}^a$, $\mathbf{MEx}_b^0$ as $\mathbf{MEx}_b$ and $\mathbf{MEx}_*^0$ as $\mathbf{MEx}$.

**Theorem 5** [Che82, Fre75, Jai95] *For all* $m, n \in N$, $a \in N \cup \{*\}$.

*(1)* $\mathbf{Ex} - \mathbf{MEx}^m \neq \emptyset$.

*(2)* $\mathbf{Ex}_0^0 - \mathbf{MEx}_n^* \neq \emptyset$.

*(3)* $\mathbf{Ex}_n^a \subseteq \mathbf{MEx}^a$.

*(4)* $\mathbf{Ex}^* = \mathbf{MEx}^*$.

*(5)* $\mathbf{MEx}_{n+1}^0 - \mathbf{Ex}_n^* \neq \emptyset$.

*(6)* $\mathbf{MEx}_0^{m+1} - \mathbf{Ex}^m \neq \emptyset$.

### 3.3 Isolated Classes

**Definition 6** [Che81] Suppose $\mathcal{A}$ and $\mathcal{B}$ are classes of recursive functions. $\mathcal{B}$ is an *a-cover* of $\mathcal{A}$ iff for each $g \in \mathcal{B}$, there exists an $f \in \mathcal{A}$, such that $f =^a g$.

**Definition 7** [Che81, Rog67] $\mathcal{C}$ is *a-immune* iff (a) $\mathcal{C}$ is infinite and (b) every recursively enumerable subclass of $\mathcal{C}$ has a finite $a$-cover.

**Definition 8** [Che81] $\mathcal{C}$ is *a-isolated* iff $\mathcal{C}$ is finite or $\mathcal{C}$ is $a$-immune.

Chen [Che81] established the following two results.

**Theorem 9** [Che81] *Suppose $m, n \in N$, and $\mathcal{S} \in \mathbf{MEx}_n^m$. Then $\mathcal{S}$ is $m$-isolated.*

**Theorem 10** [Che81] *Suppose $\mathcal{S} \in \mathbf{MEx}_0^*$. Then $\mathcal{S}$ is $*$-isolated.*

Based on above results, Chen conjectured that, for $n \in N$, every $\mathcal{S} \in \mathbf{MEx}_n^*$ is $*$-isolated. We surprisingly refute his conjecture.

## 4 Main Theorem

**Theorem 11** *There exists an infinite recursively enumerable class $\mathcal{S} \in \mathbf{MEx}_1^*$ such that $\mathcal{S}$ is not $*$-isolated.*

Proof. Let
$\mathcal{C}_1 = \{f \mid \varphi_{f(\langle 0,0\rangle)} =^* f \wedge f(\langle 0,0\rangle) \leq \mathrm{MinProg}(f) \wedge (\forall x)[f(\langle 1,x\rangle) = 0]\}$,
$\mathcal{C}_2 = \{f \mid (\forall^\infty x)[f(x) = 0] \wedge (\exists x)[f(\langle 1,x\rangle) \neq 0]\}$,
and $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$.

Intuitively, $\mathcal{C}_1$ is a class of (nearly) self-describing functions, where a small program for a finite variant of the function is coded into the function itself. $\mathcal{C}_2$ is a subclass of almost everywhere 0 functions. Additionally, we code into the functions (using $\{\langle 1,x\rangle \mid x \in N\}$) whether it is from $\mathcal{C}_1$ or $\mathcal{C}_2$.

It is easy to verify that $\mathcal{C}$ is in $\mathbf{MEx}_1^*$. We will construct the required $\mathcal{S}$ as an appropriate recursively enumerable subset of $\mathcal{C}$. Intuitively, the idea is to use an appropriate subclass of $\mathcal{C}_1$ to ensure that $\mathcal{S}$ is $*$-isolated. $\mathcal{C}_2$ is added to this subclass, to ensure that $\mathcal{S}$ is recursively enumerable. We now continue with the formal construction of $\mathcal{S}$.

Using Operator Recursion Theorem [Cas74] we will define a recursive, 1–1, increasing function $p$ such that the functions $\varphi_{p(i)}$ satisfy the following four properties:
(A) For all $x$, $\varphi_{p(i)}(\langle 0,x\rangle) = p(i)$;
(B) For all $x$, $\varphi_{p(i)}(\langle 1,x\rangle) = 0$;
(C) $\varphi_{p(i)}$ is undefined on exactly one input; let this input be called $u_i$;
(D) For all $j < p(i)$, either $\varphi_j$ is non-total, or there exists an $x < u_i$ such that $\varphi_j(x) \neq \varphi_{p(i)}(x)$.
Let $f_i$ be defined as follows:

$$f_i(x) = \begin{cases} \varphi_{p(i)}(x), & \text{if } x \neq u_i; \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathcal{S} = \{f_i \mid i \in N\} \cup \mathcal{C}_2$.

It is easy to verify that $\mathcal{S}$ is recursively enumerable. Moreover, by property (A) $\mathcal{S}$ is not $*$-isolated. It is also easy to verify (using properties (A)–(D)) that $f_i \in \mathcal{C}_1$. Thus, $\mathcal{S} \subseteq \mathcal{C}$ and $\mathcal{S} \in \mathbf{MEx}_1^*$.

We now give the construction of $\varphi_{p(i)}$ satisfying the properties (A) to (D) above. By operator recursion theorem [Cas74] there exists a 1–1, recursive, increasing function $p$ such that $\varphi_{p(i)}$ may be defined in stages as follows.

Let $u_i^0 = \min(N - \{y, x\} \mid y \in \{0,1\} \wedge x \in N)$. Let $Cancel_i^0 = \emptyset$. Intuitively, $u_i^s$ denotes the intended value of $u_i$ as at the beginning of stage $s$. $Cancel_i^s$ is used to keep track of programs $< p(i)$, against which $\varphi_{p(i)}$ has diagonalized against before stage $s$. Go to stage 0.

Stage $s$

1. Dovetail steps 2 and 3 until step 2 succeeds. If and when step 2 succeeds, go to step 4.
2. Search for a $j < p(i)$, such that $j \notin Cancel_i^s$, and $\varphi_j(u_i^s)\downarrow$.
3. For $z = 0$ to $\infty$ Do

   If $z \neq u_i^s$ and $\varphi_{p(i)}(z)$ has not been defined upto now, Then
   
   Let $\varphi_{p(i)}(z) = p(i)$, if $z = \langle 0, x\rangle$ for some $x \in N$;
   
   Let $\varphi_{p(i)}(z) = 0$, if $z = \langle y, x\rangle$ for some $x \in N$ and $y \neq 0$;

   EndFor
4. If and when step 2 succeeds, then let $j$ be as in step 2.

   Let $Cancel_i^{s+1} = Cancel_i^s \cup \{j\}$.

   Let $\varphi_{p(i)}(u_i^s) = \varphi_j(u_i^s) + 1$.

   Let $u_i^{s+1}$ be the minimum number $z$ such that $\varphi_{p(i)}(z)$ has not been defined upto now, and $z \notin \{\langle y, x\rangle \mid y \in \{0,1\} \wedge x \in N\}$.

   Go to stage $s+1$.

End Stage $s$

We now argue that $\varphi_{p(i)}$ defined above satisfies properties (A) to (D) above. First note that there are only finitely many stages. This is so since each time a new stage $> 0$ is entered, step 4 in the previous stage must have diagonalized against a new program $j < p(i)$. Since there are at most finitely many programs less than $p(i)$, there are at most finitely many stages that are executed. Let $s$ be the last stage that is entered but never finished. Let $u_i = u_i^s$ and $Cancel_i = Cancel_i^s$. It is now easy to verify that (A), (B) and (C) are satisfied. Also, for all $j < p(i)$, either $j \in Cancel_i$, or $\varphi_j(u_i)\uparrow$. In case $j \in Cancel_i$, then by step 4 of the construction, there exists a $z < u_i$ such that $\varphi_j(z)\downarrow \neq \varphi_{p(i)}(z)\downarrow$. Thus, (D) is satisfied. This completes the proof of the theorem. ∎

**Corollary 12** *For all $n > 0$, there exists a recursively enumerable class $\mathcal{S} \in \mathbf{MEx}_n^*$ such that $\mathcal{S}$ is not $*$-isolated.*

# 5 Acknowledgements

We thank an anonymous referee for several helpful suggestions which improved the presentation of the paper. Specially acknowledged are his suggestions for improving the Introduction section to make it accessible to a wider audience.

## References

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

[BF74] J. Bārzdiņš and R. Freivalds. Prediction and limiting synthesis of recursively enumerable classes of functions. *Latvijas Valsts Univ. Zimatm. Raksti*, 210:101–111, 1974.

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

[Che81] K. J. Chen. *Tradeoffs in Machine Inductive Inference*. PhD thesis, SUNY/Buffalo, 1981.

[Che82] K. J. Chen. Tradeoffs in inductive inference of nearly minimal sized programs. *Information and Control*, 52:68–86, 1982.

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

[Fre75] R. Freivalds. Minimal Gödel numbers and their identification in the limit. In *Mathematical Foundations of Computer Science*, volume 32 of *Lecture Notes in Computer Science*, pages 219–225. Springer-Verlag, 1975.

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

[Jai95] S. Jain. On a question about learning nearly minimal programs. *Information Processing Letters*, 53(1):1–4, 1995.

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