Refutable Inference of Functions Computed by Loop Programs

Miyahara, Tetsuhiro
Computer Science Laboratory, Kyushu University

出版情報：RIFIS Technical Report. 112, 1995-04-24. Research Institute of Fundamental Information Science, Kyushu University
バージョン：
権利関係：

RIFIS-TR-CS-112

RIFIS Technical Report

Refutable Inference of Functions Computed by Loop Programs

Tetsuhiro Miyahara

April 24, 1995

Research Institute of Fundamental Information Science
Kyushu University 33
Fukuoka 812, Japan
E-mail: miyahara@rc.kyushu-u.ac.jp

Refutable Inference of Functions Computed by Loop Programs

Tetsuhiro Miyahara

Computer Science Laboratory
Kyushu University 01, Ropponmatsu, Fukuoka 810, Japan
phone: +81-92-771-4161 ext.399
fax: +81-92-716-9392
e-mail: miyahara@rc.kyushu-u.ac.jp

April 24,1995

Abstract

Refutable inference is a process of inductive inference with refutability of hypothesis spaces and is essential in Machine Discovery(Mukouchi and Arikawa). Though some refutable inferabilities under language model are known, most scientific laws are represented by functions. Thus we investigated refutable inferability of function classes. In order to develop a realistic theory of function learning, we investigated the refutable inferability of primitive recursive functions computed by a concrete programming system, loop programs (Meyer and Ritchie).

Let $\mathcal{F}Loop(n)$ be the set of all primitive recursive functions computed by a loop program with at most n nesting of loops. We show $\mathcal{F}Loop(n)(n \geq 1)$ and a natural subclass of $\mathcal{F}Loop(1)$ are not refutably inferable. Thus the existent natural hierarchies of loop programs are shown to be not suitable for machine discovery. Then we construct two types of rich series of refutably inferable classes such that each union of the series is $\mathcal{F}Loop(1)$ and show the inside structures of these series.

1. Introduction

Machine Discovery is to discover scientific laws from very large experimental or observational data by computer. So implementing real machine discovery systems, e.g., ABACUS [FM90] and BONSAI [AMSKMS93], and establishing theoretical foundations of machine discovery are important and prospective research areas in artificial intelligence and machine learning. Recently, Algorithmic/Computational Learning Theory of Machine Discovery is originated by Mukouchi and Arikawa [MA93, MA95] and is attracting much attention [LW94, Mat94, WS95].

Mukouchi and Arikawa investigated the refutable inferability under formal language model and gave firm theoretical foundations of machine discovery from facts.

However, the general target of machine discovery system cannot be represented by formal language. In fact, most scientific laws are represented by functions, e.g., $F = ma$, $E = mc^2$, $PV = nRT$. Thus, this paper investigates the refutable inferability of recursive functions that are most general target concepts computed by programs and presents a theoretical foundation of machine discovery of scientific laws represented by functions.

Lange and Watoson [LW94] gave hierarchical results of identification types related to refutable inferability based on the conventional theory of function learning in Gold's paradigm [Gol67]. But the hierarchical results of identification types, inferring powers under criteria of success, are too abstract and have little to do with real machine discovery systems.

Our approach is completely different from the conventional approach. We investigated the refutable inferabilities of concrete programming systems for recursive functions in order to develop a realistic theory of function learning and machine discovery. We adopted two concrete programming systems, loop programs [MR67, Tsi70] and expressions of simple functions [Tsi70], for primitive recursive functions. And we investigate the refutable inferabilities of function classes computed by the two programming systems and construct rich series of refutably inferable classes.

This paper is organized as follows. In Section 2, we give our framework of refutable inference of recursive functions and formalize the notion of rich series of refutably inferable classes slicing a class which is not refutably inferable. In Section 3, we give brief review of loop programs and simple functions. In Section 4, we show $\mathcal{F}Loop(n)(n \geq 1)$ and a natural subclass of $\mathcal{F}Loop(1)$ are not refutably inferable, where $\mathcal{F}Loop(n)$ be the set of all primitive recursive functions computed by a loop program with at most n nesting of loops. Thus the existent natural hierarchies of loop programs are shown to be not suitable for machine discovery. In Section 5, we construct two types of rich series of refutably inferable classes slicing $\mathcal{F}Loop(1)$ and show the inside structures of the series and give a characterization of refutable inferabilities of simple functions computed by loop programs.

2. Refutable Inference of Recursive Functions

We give our framework of refutable inference of recursive functions according to Mukouchi and Arikawa [MA93, MA95].

A *number* is a natural number over $N = \{0, 1, 2, \ldots\}$. A function means a function with many inputs and a single output. Given a programming system, an *index* is a number which represents a program in the programming system. φ_i denote the recursive function computed by a program with index i.

Definition 1. A class $C = \{\varphi_i\}_{i \in N}$ of recursive functions is said to be an *indexed family of recursive functions* if there exists a recursive function $u_C : N \times N \to N$ such that

$$u_C(i, < x_1, \ldots, x_{arity(i)} >_{arity(i)}) = \varphi_i(x_1, \ldots, x_{arity(i)}),$$

where $<, \ldots, >_k$ denotes a fixed recursive encoding from N^k onto N, and $arity(i)$ denotes the total number of input variables of a program with index i. u_C represents the universal function for C.

An indexed family of recursive functions is a natural extension of an indexed family of recursive languages. A *class* means an indexed family of recursive functions. For a set S of programs, $\mathcal{F}(S)$ (or $\mathcal{F}S$) denotes the set of all functions computed by a program in S. $S|_{arity=n}$ denotes the set of all n-ary programs in S. For a set F of functions, $F|_{arity=n}$ denotes the set of all n-ary functions in F.

Example 1. The following sets of functions which are defined in later sections are all indexed families of recursive functions. In this paper,we investigate refutable inferabilities of these various classes. $\mathcal{F}Loop(n)$, $\mathcal{F}(Loop(1,m))$, $\mathcal{F}Simple$, $\mathcal{F}Simple(\#pred \leq c, \Pi dvs \leq d)$, $\mathcal{F}Simple((\#pred+3)\Pi dvs \leq k)$.

The set of all recursive functions is not an indexed family of recursive functions.

$arity(f)$ denotes the arity of a partial function f. The graph notation, $graph(f)$, of a partial function f denotes the set $\{(c_1, \ldots, c_{arity(f)}, d) \mid f(c_1, \ldots, c_{arity(f)})$is defined and equal to $d\} \subseteq N^{arity(f)+1}$. A *finite graph* is a graph notation of a partial function with a finite domain. A partial function f *explains* (or *is consistent with*) a partial function g if $arity(g) = arity(f)$ and $graph(g) \subseteq graph(f)$. For a set S, $card(S)$ denotes the cardinality of S. $FinFunc^1$ denotes the set $\{f \mid f$ is a single input 0,1-valued recursive function and $card(\{x \in N \mid f(x) = 1\})$ is finite $\}$.

A *presentation* σ of a total s-ary function f is an infinite sequence of $(s+1)$-ary pairs of numbers $(c_1^0, \ldots, c_s^0, f(c_1^0, \ldots, c_s^0)), (c_1^1, \ldots, c_s^1, f(c_1^1, \ldots, c_s^1)), (c_1^2, \ldots, c_s^2, f(c_1^2, \ldots, c_s^2)), \ldots$ such that the set $\{(c_1^0, \ldots, c_s^0, f(c_1^0, \ldots, c_s^0)), (c_1^1, \ldots, c_s^1, f(c_1^1, \ldots, c_s^1)), (c_1^2, \ldots, c_s^2, f(c_1^2, \ldots, c_s^2)), \ldots\}$ is $graph(f) \subseteq N^{s+1}$. $\sigma[n]$ denotes the initial segment of σ of length $n+1$.

Definition 2. [MA93, MA95] A *refutably inductive inference machine* (*RIIM*, for short) is an effective procedure that requests inputs from time to time, and either produces indices from time to time or produces the sign"refute" and stops. For an RIIM M and a presentation σ, $M(\sigma[n])$ denotes the last output produced by M which is successively presented $\sigma[n]$. M on σ *converges* to an index i, if there is a number n_0 such that for any $n \geq n_0$ $M(\sigma[n])$ is defined and equal to i.

Let $C = \{\varphi_i\}_{i \in N}$ a class. For a function $\varphi_i \in C$ and a presentation σ of φ_i, M *infers* φ_i w.r.t. C in the limit from σ if M on σ converges to an index j with $\varphi_j = \varphi_i$. M *refutes* the class C from σ if there exists a number n such that $M(\sigma[n])$ = "refute". An RIIM M *refutably infers* a class C if for any total function f and any presentation σ of f, if $f \in C$ then M infers f w.r.t. C in the limit from σ, otherwise M refutes the class C from σ. An RIIM M *refutably and finitely infers* a class C if M refutably infers C and in the process of inference of any $f \in C$, the RIIM M produced at most one index.

Lemma 1. [MA93, MA95] (1) If a class $C|_{arity=1}$ contains $FinFunc^1$ then the class C is not refutably inferable.

(2) If a class C satisfies the following conditions (a) and (b), then the class C is refutably inferable. (a) For any $f \notin C$, there exists a finite graph $g \subseteq graph(f) \subseteq N^{arity(f)+1}$ such that, no $\varphi_i \in C$ explains g. (b) For any finite graph g, whether or not there there exists $\varphi_i \in C$ that explains g is decidable.

The purpose of this paper is to construct rich series of refutably inferably classes of functions. So we give a brief review of rich inferable classes in language learning. Consider CSL, the set of context-sensitive languages. CSL has very rich power and is an importnt languages class. If a hypothesis space is refutably inferable, then in principle we can constract a machine discovery system for concepts in the hypothesis space. But CSL is easily shown to be not refuably inferable from Lemma 1,(1). Mukouchi and Arikawa [MA93, MA95] constructed a rich subhierarchy $\{L(LB^{[\leq m]})\}_{m\geq 1}$, defined in the follwoing, of CSL such that each $L(LB^{[\leq m]})$ is refutably inferable from complete data and solove this problem.

EFS is a kind of logic programming over character strings [ASY92]. Let $LB^{[\leq m]}$ be the set of length-bounded EFS's with at most m axioms and $L(LB^{[\leq m]})$ be the set of languages defined by such EFS's. $\cup_{m\in N} L(LB^{[\leq m]}) = CSL$. For any $m \geq 1$, $L(LB^{[\leq m]})$ contains infinitely many languages. $L(LB^{[\leq 1]})$ contains the set of pattern languages[Ang80], which is refutably inferable. In this case, we say that $\{L(LB^{[\leq m]})\}_{m\geq 1}$ is a 1-dimensional rich series of refutably inferable classes slincing CSL. Shinohara [Shi94] showed the similar richness of $L(LB^{[\leq m]})$ about inferability from positive data.

From the above consideration, we formalized the notion of rich series of refutably inferably classes of functions slicing a class which is not refutably inferble as follows. For a class C of functions, $\#func(C)$ denotes the number of functions in C.

Since an $RIIM$ know the arity of a target function from its presentation in our framework, it is worth nothing to consider a rich class which is not rich when the arity of functios in the class is fixed. Thus in the following definition, the cardinality of a class of functions with any fixed arity is considered.

Definition 3. (1) Let $C(m)$ be a class defined for $m \geq m_0$. Let C_* be a class which is not refutably inferable. $\{C(m)\}_{m\geq m_0}$ is said to be a *1-dimensional rich series of refutably inferable classes slicing* C_* if the following conditions are satisfied.

(a) There exsits a class B such that $B \subseteq C(m_0)$, $\#func(B|_{arity=s}) = \infty$ for any $s \geq 1$ and B is refutably inferable.

(b) For any $m \geq m_0$, $C(m)$ is refutably inferable and $\#func((C(m+1) \backslash C(m))|_{arity=s}) = \infty$ for any $s \geq 1$.

(c) $\cup_{m\geq m_0} C(m) = C_*$.

(2) Let $C(m,n)$ be a class defined for $m \geq m_0$ and $n \geq n_0$. Let C_* be a class which is not refutably inferable. $\{C(m,n)\}_{m\geq m_0, n\geq n_0}$ is said to be a *2-dimensional rich series of refutably inferable classes slicing* C_* if the following conditions are satisfied.

(a) There exsits a class B such that $B \subseteq C(m_0, n_0)$, $\#func(B|_{arity=s}) = \infty$ for any $s \geq 1$ and B is refutably inferable.

(b) For any $m \geq m_0$ and $n \geq n_0$, $C(m,n)$ is refutably inferable, $\#func((C(m+1,n) \setminus C(m,n))|_{arity=s}) = \infty$ for any $s \geq 1$, and $\#func((C(m,n+1) \setminus C(m,n))|_{arity=s}) = \infty$ for any $s \geq 0$.

(c) $\cup_{m \geq m_0, n \geq n_0} C(m,n) = C_*$.

3. Loop Programs and Simple Functions

Loop program is an abstract model of conventional procedure-oriented programming languages, e.g., Pascal or C. Thus the programming system of loop programs is very suitable to develop a realistic theory of function learning.

Definition 4. [MR67] Let X and Y be variables. A *loop program* is a finite sequence of instructions of the following 5 types with input and output instructions. X:=Y (substitute the contents of Y to X), X:=X+1 (increment X by 1), X:=0 (make the contents of X to 0), LOOP X, END. Each variable stores one arbitrary number. The contents of work variables and output variable are initialized to 0. The instructions "LOOP X" and "END" are balanced. "LOOP X subprogram A END" means executing subprogram A by the [X] times, where [X] is the contents of variable X upon entering the loop. The number of looping is not changed in executing the loop. For $n \in N$, $Loop(n)$ denotes the set of all loop programs containing at most n nesting of loops.

Example 2. In the following, the first loop program is in $Loop(1)$ and computes the function Z=2X+Y. The second loop program is in $Loop(2)$ and computes the function Z=XY.

```
INPUT X,Y; Z:=0; LOOP X Z:=Z+1; Z:=Z+1 END; LOOP Y Z:=Z+1 END; OUTPUT Z
INPUT X,Y; Z:=0; LOOP X LOOP Y Z:=Z+1 END END; OUTPUT Z
```

The set of SF-programs is another programming system for the set $\mathcal{F}Loop(1)$ and is needed later to construct rich series of refutable classes.

Definition 5. [Tsi70] The set of all *SF-programs* is the closure set of expressions (left hand sides of the following equations) constructed through finite number of applications of compositions from the following 8 types of initial expressions. $succ(x) = x+1$, $zero_n(x_1, \ldots, x_n) = 0$, $U_n^i(x_1, \ldots, x_n) = x_i$, $plus(x_1, x_2) = x_1 + x_2$ (2-ary), $pred(x) = x \dot{-} 1$, $if(x_1, x_2) = x_1$ if $x_2 = 0$, zero if $x_2 > 0$, $div(x,d) = x\ div\ d$(unary), $mod(x,d) = x\ mod\ d$(unary) (truncated devision and residue by a constant $d \geq 1$). (For integers x and y, $x \dot{-} y = x - y$ if $x \geq y$, 0 otherwise.)

A *simple function* is a function computed by an SF-program. For an SF-program g, $\#pred(g)$(or P_g) denotes the total number of occurrences of "pred" appearing in g. Let

$d_1, \ldots, d_k$ be all constants that occur in an SF-program g either as $div(x, d_i)$ or $mod(x, d_i)$ allowing repetition. $\Pi dvs(g)$(or D_g) denotes the product $d_1 \times \cdots \times d_k$ if such a constant exists, 1 otherwise. Let $(\#pred(g)+1)\Pi dvg(g)$ be denoted by $bound(g)$(or B_g).

Example 3. Let g and h be the following SF-programs. Then we have $\#pred(g) = 0$, $\Pi dvs(g) = 1$, $\#pred(h) = 2$, $\Pi dvs(h) = 36$.
g=plus(plus(if(x,y),succ(succ(zero$_3$(x,y,z)))),z),
h=if(div(mod(x,4),3),div(pred(pred(U$_3^2$(x,y,z))),3)).

Let $b \geq 1, d \geq 1$ and $n \geq 1$. Two points $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{x}' = (x'_1, \ldots, x'_n)$ are *compatible* w.r.t. b and d if the following two conditions are satisfied. (1) For each i $(1 \leq i \leq n)$ with $x_i < b$ or $x'_i < b$, we have $x_i = x'_i$. (2) For each i $(1 \leq i \leq n)$ with $x_i \geq b$ and $x'_i \geq b$, we have $x_i = x'_i (mod\ d)$. Compatibility w.r.t. two constants b and d is an equivalence relation on N^n. For $n \geq 1$ and $b \geq 0$, $Cube(n, b)$ denotes the set $\{(x_1, \ldots, x_n) \in N^n \mid 0 \leq \forall i \leq n, 0 \leq x_i < b\}$.

Let $b \geq 1$, $d \geq 1$, $n \geq 1$ and e be a graph notation of n-ary function with $Cube(n, b + 2d)$ as its domain. Let $\mathbf{x} = (x_1, \ldots, x_n)$ be any point on N^n. Consider the points $\mathbf{y} = (y_1, \ldots, y_n)$ and $\mathbf{z}^r = (z^r_1, \ldots, z^r{}_n)(1 \leq r \leq n)$ defined as follows. Let $y_i = x_i$ for i with $1 \leq i \leq n$ and $x_i < b$, let $y_i = b + (x_i\ mod\ d)$ for i with $1 \leq i \leq n$ and $x_i \geq b$, let $z^r_i = y_i$ for i with $1 \leq i \leq n$ and $i \neq r$, and let $z^r_r = y_r + d$. Define the function $L(n, b, d, e)(\mathbf{x}) = e(\mathbf{y}) + \sum_{r=1}^{n}(e(\mathbf{z}^r) - e(\mathbf{y}))(x_r - y_r)/d$.

We summarize the fundamental results of loop programs and simple functions.

Theorem 1. [MR67, Tsi70] (i) $\mathcal{F}(Loop(0)|_{arity=n}) = \{f(x_1, \ldots, x_n) = x_i + k \mid 1 \leq i \leq n$ and $k \in N\} \cup \{f(x_1, \ldots, x_n) = k \mid k \in N\}$. Thus, $\mathcal{F}Loop(0)$ is refutably inferable.

(ii) $\mathcal{F}Loop(1)$ is the set of all simple functions.

(iii) $\mathcal{F}Loop(n) \subsetneq \mathcal{F}Loop(n+1)(n \geq 0)$.

(iv) $\cup_{n \in N}\mathcal{F}Loop(n)$ is the set of all primitive recursive functions.

(v) $Loop(2)$ contains $T(x, x, y)$, where T is Kleene's T-predicate.

(vi) Let g be an SF-program with arity n. For any equivalent class C of compatible points w.r.t. $bound(g)$ and $\Pi dvs(g)$ and any i with $1 \leq i \leq n$, there exist constants $p^C_i \in Q^+_0$ such that the following equation holds, where Q^+_0 denotes the set of all non-negative rational numbers. For any $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n) \in C$, $g(\mathbf{x}) - g(\mathbf{y}) = \sum_{i=1}^{n} p^C_i (x_i - y_i)$. Further, $g = L(n, bound(g), \Pi dvs(g), graph(g)|_{Cube(n, bound(g)+2\Pi dvs(g))})$.

(vii) Consider $g_1 = L(n, b_1, d_1, e_1)$ and $g_2 = L(n, b_2, d_2, e_2)$. If g_1 and g_2 are agree on $Cube(n, max(b_1, b_2) + 2 \times gcm(d_1, d_2))$, then g_1 and g_2 are agree on N^n.

4. Existent Natural Hierarchies of Loop Programs Are Not Suitable for Machine Discovery

Let $Prim$ be the set of all primitive recursive functions. Since $Prim$ contains $FinFunc^1$, $Prim$ is not refutably inferable. The hierarchy $\{\mathcal{F}Loop(n)\}_{n\geq 0}$ is a candidate for rich series of refutably inferable classes slicing $Prim$. So we investigate refutable inferability of $\mathcal{F}Loop(n)$.

Theorem 2. (1) $FinFunc^1 \subseteq \mathcal{F}(Loop(1)|_{arity=1})$.

(2) For any $n \geq 1$, $\mathcal{F}Loop(n)$ are not refutably inferable.

Proof. Let f be a function in $FinFunc^1$ defined by $f(x) = 1$ if $x = d_i(1 \leq i \leq n), 0$ otherwise, where $d_1, \ldots, d_n$ are constants satisfying $d_1 < d_2 < \ldots < d_n(n \geq 0)$.

The following Loop(1) program P computes the function f.

```
{Program P}
INPUT X
Y:=0;I[0]:=0;I[1]:=I[1]+1;I[2]:=I[2]+1;...,I[dn-1]:=I[dn-1]+1;
I[dn]:=I[dn]+1;I[dn+1]:=I[dn+1]+1;
LOOP X  I[dn+1]:=I[dn];I[dn]:=I[dn-1]; ... I[2]:=I[1];I[1]:=0 END;
LOOP I[d1+1]  Y[1]:=Y[1]+1 END; LOOP I[d1]    Y[1]:=0   END
LOOP I[d2+1]  Y[2]:=Y[2]+1 END; LOOP I[d2]    Y[2]:=0   END
...
LOOP I[dn+1]  Y[n]:=Y[n]+1 END; LOOP I[dn]    Y[n]:=0   END;
LOOP Y[1] Y:=Y+1 END; LOOP Y[2] Y:=Y+1 END; ... LOOP Y[n] Y:=Y+1 END;
OUTPUT Y
```

□

Even the second smallest class $\mathcal{F}Loop(1)$ in the hierarchy $\{\mathcal{F}Loop(n)\}_{n\geq 0}$ is not refutably inferable. Theorem 1(i),(v) say that $\mathcal{F}Loop(0)$ is a trivial class and $\mathcal{F}Loop(2)$ is a set of functions hard to compute in realistic meaning. So we focus our attention on $\mathcal{F}Loop(1)$ and seek a rich series of refutable inferable classes slicing $\mathcal{F}Loop(1)$. Goetze and Nehrich[GN65] constructed a natural sub-hierarchy $\{\mathcal{F}Loop(1,m)\}_{m\geq 0}$ of $\mathcal{F}Loop(1)$, where $Loop(1,m)$ denotes the set of all loop programs containing at most m unnested loops. The first program in Example 2 is in $Loop(1,2)$. $\mathcal{F}Loop(1,1)$ is conceived to be the smallest class in the existent natural hierarchies of loop functions containing $\mathcal{F}Loop(0)$.

By using a simulation technique [GN65], we show that $\mathcal{F}Loop(1,1)$ is not refutably inferable. Thus, we need another view in order to construct a rich series of refutable classes slicing $\mathcal{F}Loop(1)$.

5. Refutable Inference of Simple Functions Computed by Loop Programs

We adopt SF-programs as another programming system for $\mathcal{F}Loop(1)$, and $\#pred(g)$ and $\Pi dvs(g)$ as views for measuring an SF-program g. Since $\mathcal{F}Simple(arity = n)$, defined in the following, is equal to the set of all n-ary simple functions, we use an SF-program in $Simple(arity = n)$ as a standard form of an n-ary SF-program.

Definition 6. Let $Simple(arity = n)$ be the set of expressions defined inductively as follows. Let $x_1, \ldots, x_n$ be variables. (1) $zero_n(x_1, \ldots, x_n)$, $U_n^i(x_1, \ldots, x_n)(0 \leq i \leq n) \in Simple(arity = n)$. (2) $f, f_1, f_2 \in Simple(arity = n) \Rightarrow succ(f)$, $plus(f_1, f_2)$, $pred(f)$, $if(f_1, f_2)$, $div(f, k)(k \geq 1)$, $mod(f, k)(k \geq 1) \in Simple(arity = n)$.

Let $Simple = \cup_{n \geq 1} Simle(arity = n)$. For $c \geq 0$ and $d \geq 1$, $Simple(\#pred \leq c, \Pi dvs \leq d)$ denotes the set $\{g \in Simple \mid \#pred(g) \leq c, \Pi dvs(g) \leq d\}$. For $c \geq 0$, $d \geq 1$ and $n \geq 1$, $Simple(\leq c, \leq d, = n)$ denotes the set $\{g \in Simple(arity = n) \mid \#pred(g) \leq c, \Pi dvs(g) \leq d\}$. For $k \geq 3$ and $n \geq 1$, $Simple((\#pred + 3)\Pi dvs \leq k)$ denotes the set $\{g \in Simple \mid (\#pred(g) + 3)\Pi dvs(g) \leq k\}$.

First, we show the inside structure of $\mathcal{F}Simple(arity = n)$. The following lemma says that $\mathcal{F}Simple(\leq c, \leq d, = n)$ is a fundamental subclass with 2 independent views.

Lemma 2. Let $c, c' \geq 0$,and $d, d' \geq 1$,and $n \geq 1$.

(1) $\mathcal{F}(Loop(0)|_{arity=n}) \subsetneq \mathcal{F}Simple(\leq 0, \leq 1, = n)$.

(2) $\mathcal{F}Simple(\leq c, \leq d, = n) \subsetneq \mathcal{F}Simple(\leq c + 1, \leq d, = n)$.

(3) $\mathcal{F}Simple(\leq c, \leq d, = n) \subsetneq \mathcal{F}Simple(\leq c, \leq d + 1, = n)$.

Further, each gap in (1),(2),(3) contains infinitely many functions.

(4) [c=c' and d=d'] $\Leftrightarrow \mathcal{F}Simple(\leq c, \leq d, = n) = \mathcal{F}Simple(\leq c', \leq d', = n)$.

Proof. (1) Let $\mathbf{x} = (x_1, \ldots, x_n)$. Consider $g_s(\mathbf{x}) = if(s, x_1) \in \mathcal{F}Simple(\leq 0, \leq 1, = n)(s \geq 1)$. By Theorem 1, $g_s \notin \mathcal{F}Loop(0)|_{arity=n}$.

(2) Consider $g_s(\mathbf{x}) = if(x_1, div(x_1, d) \dot{-} (c+1)) + s \in \mathcal{F}Simple(\leq c+1, \leq d, = n)(s \in N)$. Assume that there exists $h \in Simple(\leq c, \leq d, = n)$ such that $g_s(\mathbf{x}) = h(\mathbf{x})$. Let $x_0 = (c+2)d-1$, $\mathbf{x_0} = (x_0, 0, \ldots, 0)$, $\mathbf{x_1} = (x_0 + D_h, 0, \ldots, 0)$. We have $g_s(\mathbf{x_0}) = x_0 + s, g_s(\mathbf{x_1}) = s$. Since $\mathbf{x_0}, \mathbf{x_1}$ are compatible w.r.t. B_h and D_h, Theorem 1 shows that there exist a constant $q \in Q_0^+$ such that $g_s(\mathbf{x_1}) - g_s(\mathbf{x_0}) = qD_h$ and Thus $q = -x_0/D_h$ $(x_0 \geq 1, D_h \geq 1)$. This is a contradiction.

(3) Consider $g_s(\mathbf{x}) = div(x_1, d+1) + s \in \mathcal{F}Simple(\leq c, \leq d+1, = n)\,(s \in N)$. Assume that there exists $h \in Simple(\leq c, \leq d, = n)$ such that $g_s(\mathbf{x}) = h(\mathbf{x})$. Let $x_0 = min\{t(d+1) \mid t \in N, t(d+1) \geq B_h\}$, $\mathbf{x_0} = (x_0, 0, \ldots, 0)$, $\mathbf{x_1} = (x_0 + D_h, 0, \ldots, 0)$, $\mathbf{x_2} = (x_0 + (d\ div\ D_h)D_h, 0, \ldots, 0)$, $\mathbf{x_3} = (x_0 + (d\ div\ D_h + 1)D_h, 0, \ldots, 0)$. We have $g_s(\mathbf{x_0}) = g_s(\mathbf{x_1}) = g_s(\mathbf{x_2}) = t + s$, $g_s(\mathbf{x_3}) = t + 1 + s$. Since $\mathbf{x_0}, \mathbf{x_1}, \mathbf{x_2}$ and $\mathbf{x_3}$ are compatible w.r.t. B_h and

D_h, Theorem 1 shows that there exist a constant $q \in Q_0^+$ such that $g_s(\mathbf{x_1}) - g_s(\mathbf{x_0}) = qD_h$ and $g_s(\mathbf{x_3}) - g_s(\mathbf{x_2}) = qD_h$. Thus $q = 0$ and $q = 1/D_h (D_h \geq 1)$. This is a contradiction.

(4) By arguments similar to (2) and (3). □

procedure $RIIM\_SF(c, d)$;
input: a presentation of a total function;
given: integers $c \geq 0, d \geq 1$;
output: an SF-program in $Simple(\#pred \leq c, \Pi dvs \leq d)$ or "refute";
begin
  $EX := \emptyset$; $read\_store(EX)$;
  let n be the arity of a presentation in EX;
  repeat $read\_store(EX)$ **until** $\{(x_1, \ldots, x_n) \mid (x_1, \ldots, x_n, y) \in EX\}$ contains $Cube(n, (c+3)d)$;
  $g := \{(x_1, \ldots, x_n, y) \in EX \mid (x_1, \ldots, x_n) \in Cube(n, (c+3)d)\}$;
  $H := \{h = L(n, (P+1)D, D, g) \mid 0 \leq P \leq c, 1 \leq D \leq d, h \; explains \; EX\}$;
  while H contains two equivalent functions **do** remove the redundant function from H;
  /* $card(H)$ denotes the number of elements in H */
  if $card(H) = 0$ **then output** "refute" and **stop**;
  while $card(H) > 1$ **do begin**
    $read\_store(EX)$;
    remove from H all functions that do not explain EX;
    if $card(H) = 0$ **then output** "refute" and **stop**
  end;
  let h be the only function left in H ;
  alternately execute the following two processes 1 and 2;
  begin /* process 1 */
    repeat $read\_store(EX)$ **until** h does not explain EX;
    output "refute" and **stop**
  end;
  begin /* process 2 */
    enumerate all programs q in $Simple(\#pred \leq c, \Pi dvs \leq d, arity = n)$
    and search for a program q that is equivalent to h;
    if such a program q is found **then output** q and **stop** the process 2
  end;
end;
procedure $read\_store(EX)$;
begin
  read the next fact $(x_1, \ldots, x_n, y)$; $EX := EX \cup \{(x_1, \ldots, x_n, y)\}$
end

An RIIM for the class $\mathcal{F}Simple(\#pred \leq c, \Pi dvs \leq d)$

We give a procedure $RIIM\_SF(c, d)$ that refutably infers class $\mathcal{F}Simple(\#pred \leq c, \Pi dvs \leq d)$ and produces only one SF-program. An RIIM for $\mathcal{F}Simple((\#pred{+}3)\Pi dvs \leq$

k) is similarly obtained. Thus, we have the following refutable inferabilities and rich series of refutably inferable classes slicing $\mathcal{F}Loop(1)$.

Theorem 3. (1) For any $c \geq 0$ and $d \geq 1$, the class $\mathcal{F}Simple(\#pred \leq c, \Pi dvs \leq d)$ is refutably and finitely inferable. Thus, $\{\mathcal{F}Simple(\#pred \leq c, \Pi dvs \leq d)\}_{c\geq 0, d\geq 1}$ is a 2-dimensional rich series of refutably inferable classes slicing $\mathcal{F}Loop(1)$.

(2) For any $k \geq 3$, the class $\mathcal{F}Simple((\#pred + 3)\Pi dvs \leq k)$ is refutably and finitely inferable. Thus, $\{\mathcal{F}Simple((\#pred + 3)\Pi dvs \leq k)\}_{k\geq 3}$ is a 1-dimensional rich series of refutably inferable classes slicing $\mathcal{F}Loop(1)$. And this slicing is maximal w.r.t. the size of finite examples needed to determine all candidates.

6. Concluding Remarks

In this paper, we investigated refutable inferabilities of primitive recursive functions computed by a concrete programming system, loop programs, in order to develop a realistic theory of machine discovery for scientific laws represented by functions. The existent natural hierarchies of loop programs are shown to be not suitable for machine discovery. Then we construct two types of rich series of refutably inferable classes and gave a characterization of refutable inferabilities of simple functions computed by loop programs.

Acknowledgments

The author wishes to thank Professor Setsuo Arikawa for his essential suggestions which initiated this work. He also wishes to thank Takayoshi Shoudai, Hiroki Arimura, Eiju Hirowatari, Takeshi Shinohara and Hiroki Ishizaka for their productive suggestions and encouragements.

References

[Ang80] Angluin,D., Inductive inference formal languages from positive data, Information and Control, vol.45, pp.117-135, 1980.

[AS83] Angluin,D., Smith,C.H., Inductive inference:Theory and methods, Computing Survey, vol.15, pp.237-269, 1983.

[ASY92] Arikawa,S.,Shinohara,T.,Yamamoto,A., Learning elementary formal systems, Theoretical Computer Science, vol.95, pp.97-113, 1992.

[AMSKMS93] Arikawa,S.,Miyano,.S.,Shinohara,A.,Kuhara,S.,Mukouchi,Y.,Shinohara,T., A machine discovery from amino acid sequences by decision trees over regular patterns, New Generation Computing, vol.11,pp.361-375,1993.

[BB75] Blum,L.,Blum.,M., Toward a mathematical theory of inductive inference, Information and Control, vol.28, pp.125-155, 1975.

[CS83] Case,J.,Smith,C., Comparison of identification criteria for machine inductive inference, Theoretical Computer Science, vol.25, pp.193-220, 1983.

[FM90] Falkanbaineer,B.C., Michalski,K.S., Integrating quantative and qualitative discovery in ABACUS system, Kodratoff,Y., Michalski,R. ed. Machine Learning, An artificial intelligence, vol.3, pp.153-190, 1990.

[GN65] Goetze,B., Nehrich,W., The number of loops necessary and sufficient for computing simple functions, EIK, vol.17, 363-376, 1981.

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

[LW94] Lange,S.,Watson.,P., Machine discovery in the presence of incomplete or ambiguous data, Proc. of ALT 94, Springer-Verlag,LNAI 872.,438-452, 1994.

[Mat94] S.Matsumoto,A.Shinohara, Refutably Probably Approximately Correct Learning, Proc. of ALT 94, Springer-Verlag,LNAI 872.,469-483, 1994.

[MA93] Mukouchi,Y., Arikawa,S., Inductive inference machines that can refute hypothesis spaces, Proc. of ALT'93, LNAI 744, pp.123-136, 1993.

[MA95] Mukouchi,Y., Arikawa,S., Towards a mathematical theory of machine discovery from facts, Theoretical Computer Science, 1995.

[MR67] Meyer,A.R., Ritchie,D.M., The complexity of loop programs, Proc. of 22nd National Conference of ACM, pp.465-469, 1967

[Shi94] Shinohara,T., Rich class inferable from positive data: length-bounded elementary formal systems, Information and Computation, vol.108, pp.175-186, 1994.

[Tsi70] Tsichritzis,D., The equivalence problem of simple programs, J.ACM, vol.17, No.4, pp.729-738, 1970.

[WS95] Watanabe, N., Sato, M., On a class of regular languages refutably inferable from complete date in polynomial-time, LA Symposium, Winter, 1995, Kyoto.