INDUCTIVE INFERENCE OF RECURSIVE FUNCTIONS: COMPLEXITY BOUNDS

Rūsiņš Freivalds, Jānis Bārzdiņš and Kārlis Podnieks

Institute of Mathematics and Computer Science
The University of Latvia
Raiņa bulv. 29, Rīga, 226250, Latvia

Abstract. This survey includes principal results on complexity of inductive inference for recursively enumerable classes of total recursive functions. Inductive inference is a process to find an algorithm from sample computations. In the case when the given class of functions is recursively enumerable it is easy to define a natural complexity measure for the inductive inference, namely, the worst-case mindchange number for the first n functions in the given class. Surely, the complexity depends not only on the class, but also on the numbering, i.e. which function is the first, which one is the second, etc. It turns out that, if the result of inference is Goedel number, then complexity of inference may vary between $\log_2 n + o(\log_2 n)$ and an arbitrarily slow recursive function. If the result of the inference is an index in the numbering of the recursively enumerable class, then the complexity may go up to const·n. Additionally, effects previously found in the Kolmogorov complexity theory are discovered in the complexity of inductive inference as well.

The time complexity of prediction strategies (the value f(m+1) is predicted from f(0),...,f(m)) is investigated. It turns out that, if a prediction strategy F is "error-optimal" (i.e. it makes at most $\log_2 n + O(\log_2 \log_2 n)$ errors on the n-th function of the class), then the time complexity of computation of F(<f(0),...,f(m)>) (i.e. a candidate for f(m+1)) may go up, in some sense, to $2^{2^{cm}}$.

Special attention is paid to inductive inference by probabilistic algorithms. It turns out that arbitrary recursively enumerable class of total recursive functions can be identified with ln n + o(log n) mind- changes in an arbitrary numbering of the class.

1. Introduction

"Inductive inference" is the term coined for finding out the algorithm from sample computations. We restrict ourselves to the case when a total recursive function is to be identified. The first paper in this area was [Go 67], yet (sometimes indirectly) the research was influenced by the theory of experiments with finite automata [Moo 56].

There are several ways how to make this problem precise but all of them are based on the same paradigm. There is a "black box" with a given total recursive function f in it. We cannot see the program of the device computing f but we can get the values of the function. Since the function is total, with no restriction of generality we can assume that the black box outputs the values in the natural order: f(0),f(1),f(2),f(3),...

The inductive inference machine (or the *strategy*) tries to use the initial fragments of the function to figure out the algorithm computing it. Hence, from the recursion theory point of view, the strategy is a functional mapping the class of total recursive functions $\mathscr{R}$ into the set of nonnegative integers N. This functional is to be computable in some sense. Theory of recursive functions [Rog 67] has developed a precise notion for such a functional - the notion of a recursive functional. Informally, recursive functional is computed by a Turing machine with an input tape containing the graph of the function f and a work tape. The machine works for some time and then stops after finite number of steps (the machine decides itself when to stop) and produces the result needed.

Unfortunately, only very simple classes of functions are identifiable in this sense. Indeed, in finite number of steps only finite number of values of the function can be observed. If two functions differ only on a later value, then the machine nevertheless produces the same output.

A more interesting type of identification was "identification in the limit" considered in [Go 67]. Instead of being printed once forever, the output ("hypothesis") is shown on a "screenboard" and, if there is a need, it may be changed later. We say that the machine has resulted in y if at some moment it has produced the output y and after that moment this output is never changed.

Formally, the *identifying strategy* F is an arbitrary partial recursive function. $\langle x_1,x_2,\ldots,x_n\rangle$ is an effective numbering of all tuples of nonnegative integers, using as the numbers all nonnegative integers. $\{\varphi_i\}$ is a Goedel numbering of all partial recursive functions of one argument.

$F(\langle f(0),\ldots,f(n)\rangle)$ is referred to as the n-th *hypothesis* by F on the function f. The hypothesis p is called correct for f if $\varphi_p=f$.

We say that f is *identified in the limit* by F (denoted $f\in EX(F)$) if there is an n_0 such that for arbitrary $n>n_0$:

1) $F(\langle f(0),\ldots,f(n)\rangle)=F(\langle f(0),\ldots,f(n_0)\rangle)$,

2) the hypothesis $F(\langle f(0),\ldots,f(n_0)\rangle)$ is correct for f.

We say that the class U of total recursive functions is identified in the limit by F (denoted $U\subseteq EX(F)$) if every function $f\in U$ is identified in the limit by F.

We say that the class U of total recursive functions is identifiable in the limit ($U\in EX$) if there is a strategy F identifying U in the limit.

The class U of total recursive function is called *recursively enumerable* if there is a total recursive function g(i,x) such that:

1) for arbitrary i the function $\lambda x\cdot g(i,x)$ of one argument x is in the class U,

2) for arbitrary $f\in U$ there is an i such that $\lambda x\cdot g(i,x)=f(x)$.

The function g introduces a numbering $\tau=\{\tau_i\}$ of functions in U, namely, the number i is called the index of the function f if $\tau_i(x)=\lambda x.g(i,x)=f(x)$.

THEOREM 1.1. (E.M.GOLD [Go 67]) If a class U is a subclass of a recursively enumerable class of functions, then U is identifiable in the limit.

PROOF. The strategy produces as its n-th hypothesis

$$\begin{cases} \text{i, if } i\le n \text{ and i is the least} \\ \text{nonnegative integer j such that} \\ \langle f(0),\ldots,f(n)\rangle=\langle \tau_j(0),\ldots,\tau_j(n)\rangle; \\ \text{n, if there is no such i for the given n.} \end{cases}$$

It is easy to see that the strategy is total recursive and it identifies U in the limit. Moreover, our strategy never allows more than n mindchanges on the functions with indices 0,1,2,...,n.

□

The worst-case number of mindchanges for the first n functions in the class U (more precisely: in the numbering τ of the class U) can be considered as a complexity measure for the pair (U,τ). Our paper is written to find out how the numbering influences this complexity for the given recursively enumerable class U. We make a terminological distinction: recursively enumerable class U of total recursive func- tions but *enumerated class* (U,τ), i.e. U with its fixed numbering τ.

This way, we try to understand in this paper how different complexities of distinct enumerated classes (U,τ) based on the same recursively enumerable class U can be.

We will show that the linear complexity in the proof of Theorem 1.1 can be improved if we are interested only in getting a correct Goedel number for the given function. On the other hand, the proof of Theorem 1.1 yields us more than it is said in the formulation of Theorem 1.1. The strategy with the linear complexity of mindchanges produces the τ-index, one can effectively find a Goedel number for the same function but in the general case it is a recursively unsolvable problem to find a τ-index, given arbitrary Goedel number. Hence we can expect higher complexity for identification of τ-indices when compared with the identification of Goedel numbers. In Section 3 we will see that this really is the case.

We will consider also a notion which appears to be closely connected with the identification in the limit, called prediction of functions.

In the *prediction of functions* the result $F(\langle f(0),\ldots,f(n)\rangle)$ is expected to be $f(n+1)$. Nevertheless arbitrary finite number of errors is allowed (but it is not allowed for the value $F(\langle f(0),\ldots,f(n)\rangle)$ to be undefined).

Prediction turns to be closely connected with identification in the limit. Given arbitrary recursively enumerable class U of total recursive functions and its numbering τ, if (U,τ) can be predicted with $\leq g(n)$ errors, then (U,τ) can be identified in the limit with $\leq g(n)$ mindchanges (see Theorem 1.2 below).

To be able to prove this (very simple) theorem and other results like it we introduce a useful notation.

The string of integers $f(0), f(1),\ldots,f(n)$ is denoted by $f^{[n]}$. This allows us to write $F(\langle f^{[n]}\rangle)$ instead of $F(\langle f(0),\ldots,f(n)\rangle)$.

We denote by $F^{NV}(f)$ the *number of errors* while predicting f by the predicting strategy F.

We fix a Goedel numbering $\varphi=\{\varphi_i\}$ of all partial recursive functions of one argument x. We denote by $F^{EX}(f)$ the *number of mindchanges* by F on f, provided F correctly identifies in the limit a φ-index of the function f.(Please notice that for the sake of brevity we have omitted φ in the notation $F^{EX}(f)$. Of course, it should be written).

We denote by $F^{NV}_{U,\tau}(n)$ the maximum among $\{F^{NV}(\tau_0), F^{NV}(\tau_1),\ldots,F^{NV}(\tau_n)\}$. Similarly, by $F^{EX}_{U,\tau}(n)$ we denote the max among $\{F^{EX}(\tau_0), F^{EX}(\tau_1),\ldots,F^{EX}(\tau_n)\}$.

We denote by $F^{\tau}(f)$ the *number of mindchanges* by F on f,

provided F correctly identifies in the limit a τ-index of the function f. We denote by $F^{\tau}_{U,\tau}(n)$ the maximum among $\{F^{\tau}(\tau_0), F^{\tau}(\tau_1),\ldots,F^{\tau}(\tau_n)\}$.

THEOREM 1.2. For arbitrary enumerated class (U,τ) and arbitrary total recursive strategy F predicting U, there is a total recursive strategy G identifying U in the limit such that $G^{EX}_{U,\tau}(n)\leq F^{NV}_{U,\tau}(n)$.

PROOF. Let $y_0,y_1,\ldots y_n$ be a tuple of nonnegative integers and F be the total recursive strategy predicting U. We consider a partial recursive function η defined as follows

$$\eta(x)=\begin{cases} y_x, \text{ if } x\leq n, \\ F(\langle y_0,y_1,\ldots,y_n\rangle), \text{ if } x=n+1, \\ F(\langle \eta(0),\eta(1),\ldots,\eta(x-1)\rangle), \text{ if } x>n+1. \end{cases}$$

The algorithm for computing values of η is uniform in $n,y_0,y_1,\ldots,y_n$. Hence there is a total recursive function j such that $j(\langle y_0,y_1,\ldots,y_n\rangle)$ is a φ-index of the function η, corresponding the tuple $(y_0,y_1,\ldots,y_n)$.

If f is a total recursive function and the predicting strategy F makes no more errors on initial fragments $(f(0),f(1),\ldots,f(x))$ containing $(f(0),f(1),\ldots,f(n))$, then η is total and $\eta=f$.

We consider a strategy G such that

$$G(\langle y_0,y_1,\ldots,y_n\rangle)=j(\langle y_0,y_1,\ldots,y_n\rangle)$$

for all values of the argument. For every total recursive function f, the number of mindchanges by G equals the number of errors by F.

□

A strategy F identifying τ-indices for a class U is called consistent if for arbitrary n and arbitrary $f\in U$ the value $F(\langle f(0),f(1),\ldots,f(n)\rangle)$ is a τ-index i such that $\tau_i(0)=f(0)$, $\tau_i(1)=f(1)$, ... , $\tau_i(n)=f(n)$.

THEOREM 1.3. For arbitrary enumerated class (U,τ) and arbitrary consistent total recursive strategy H identifying for U τ-indices in the limit, there is a total recursive strategy F predicting U such that $F^{NV}_{U,\tau}(n)\leq H^{\tau}_{U,\tau}(n)$.

PROOF. If $H(\langle f(0),f(1),\ldots,f(n)\rangle)=i$, then set

$$F(\langle f(0),f(1),\ldots,f(n)\rangle)= \tau_i(n+1).$$

Since H is consistent, every error by F implies a mindchange by H.

□

We need a useful "folk lemma" used by nearly all authors in papers on inductive inference. We have added the complexity bounds to the argument used in this lemma.

LEMMA 1.1. For arbitrary Goedel numbering $\{F_i\}$ of all partial recursive strategies there is a family $\{F'\}$ of total recursive strategies such that for arbitrary i and total recursive f if $\lim_{n\to\infty} F_i(<f^{[n]}>)$ exists, then $\lim_{n\to\infty} F'_i(<f^{[n]}>)$ exists as well, the limits are equal, and for all n

$$(F'_i)^{\tau}_{U,\tau}(n) \le (F_i)^{\tau}_{U,\tau}(n)+1, \quad (F'_i)^{EX}_{U,\tau}(n) \le (F_i)^{EX}_{U,\tau}(n)+1.$$

PROOF. The strategy F'_i on $<f^{[n]}>$ simulates in total n steps of Turing machine computation for $F_i(<f^{[0]}>)$, $F_i(<f^{[1]}>)$, $F_i(<f^{[2]}>)$,... (in that order). The result $F'_i(<f^{[n]}>)$ equals the last completely computed value in this sequence. If time n does not suffice to compute $F_i(<f^{[0]}>)$, then $F'_i(<f^{[n]}>)=0$. □

2. Prediction and EX-identification

The proof of Theorem 1.1. provides strategies for prediction, identification in the limit and identification of τ-indices with the following complexity bounds:

$$F^{NV}_{U,\tau}(n) \le n,$$

$$G^{EX}_{U,\tau}(n) \le n,$$

$$H^{\tau}_{U,\tau}(n) \le n$$

for arbitrary enumerated classes (U,τ). We prove in this section that the first two bounds can be lowered.

THEOREM 2.1. ([BF 72], [BF 74]) For arbitrary enumerated class (U,τ) and arbitrary positive integer k, there is a total recursive strategy F such that for all n

$$F^{NV}_{U,\tau}(n) \le \log_2 n+\log_2\log_2 n+\ldots+\log_2\log_2\ldots\log_2 n+ \quad (\text{k times})$$
$$+o(\log_2\log_2\ldots\log_2 n) \quad (\text{k times})$$

PROOF. The main idea is as follows. We associate a certain weight p_i ($\Sigma p_i=1$) to every τ-index i, and, then, to predict the next value $y'_{n+1}=F(<y_0,y_1,y_2,\ldots,y_m>)$, we consider a parameter s, and for arbitrary fixed value of s we total the weights for all integers j such that

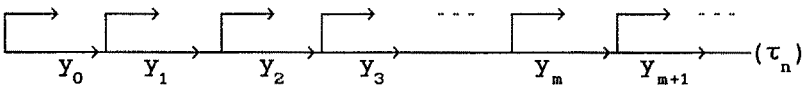
$$\tau_j(0)=y_0 \& \tau_j(1)=y_1 \& \tau_j(2)=y_2 \& \ldots \& \tau_j(m)=y_m \& \tau_j(m+1)=s.$$

Our prediction of y'_{m+1} is the value of s for which the abovedescribed total is maximal.

We assert that if our strategy of prediction makes k errors on the function τ_n, then

$$2^k p_n \leq 1. \qquad (2.1.)$$

Indeed, consider a graphical representation of the class U by an infinite tree.



The infinite path drawn here corresponds to the function τ_n (which may have more than one τ-index, by the way). The outgoing arrows correspond to functions declining from τ_n.

The function τ_n has the total weights no less than p_n. Consider the last error, the error number k. If our strategy has chosen to predict a value differing from that of τ_n, it is only because the weight of the declining arrow has had a weight no less than p_n. Hence the weight of the correct prediction at the moment of the (k-1)-th error has been at least $2 \cdot p_n$. Since the (k-1)-th error has been commited, another declining arrow has had a weight $\geq 2 \cdot p_n$. Hence the weight of the correct prediction at the moment of the (k-2)-th error has been at least $4 \cdot p_n$. Continuing this consideration we get (2.1.).

We conclude that our strategy makes no more than $\log_2 \frac{1}{p_n}$ errors on the function τ_n. If we use the distribution of weights

$$p_n = \frac{c}{n \cdot (\log_2 n)(\log_2 \log_2 n) \ldots (\log_2 \ldots \log_2 n)(\log_2 \ldots \log_2 n)^2}$$

(k-1 times) (k times)

(where c is a constant such that $\sum p_n = 1$), we get the upper bound

$$F^{NV}_{U,\tau}(n) \leq \log_2 n + \log_2 \log_2 n + \ldots + \log_2 \log_2 \ldots \log_2 n + \qquad (2.2.)$$

(k times)

$$+ o(\log_2 \log_2 \ldots \log_2 n)$$

(k times)

We have been slightly incorrect so far. We cannot guarantee the recursiveness of the strategy since absolutely precise computation of an infinite series of weights is expected. Now we redefine the strategy expecting the totals of weights being computed only approximately, namely, the totals needed for the current prediction being computed only up to a certain ε_t where ε_t depends only on the number of errors already commited.

We have that the total of weights p for the prediction at the

moment of the k-th (the last) error always satisfies

$$p+\varepsilon_k \geq p_n-\varepsilon_k$$

i.e. the weight of the right arrow at the previous moment is no less than

$$2p_n-2\varepsilon_k .$$

For the moment of the last but one error we have

$$p'+\varepsilon_{k-1} \geq 2p_n-2\varepsilon_k-\varepsilon_{k-1}$$

and for the right arrow at the previous moment we have the weight

$$\geq 2^2\cdot p_n-2^2\cdot\varepsilon_k-2\varepsilon_{k-1} .$$

Continuing this argument we finally get a weight

$$\geq 2^k\cdot p_n-2^k\cdot\varepsilon_k-\ldots-2^2\varepsilon_2-2\varepsilon_1$$

which cannot exceed 1. If we take $\varepsilon_j=2^{-2j}$, we have $2^k\cdot p_n\leq 2$ and the same inequality (2.2.).

□

THEOREM 2.2. ([BF 74]) For arbitrary enumerated class (U,τ) and arbitrary positive integer k, there is a total recursive strategy G such that for all n

$$G^{EX}_{U,\tau}(n) \leq \log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2 n}_{k \text{ times}} + o(\underbrace{\log_2\log_2\ldots\log_2 n}_{k \text{ times}}) .$$

PROOF. Immediately from Theorems 2.1 and 1.2.

□

In order to prove the lower bounds of the complexity of prediction we introduce some auxiliary notions and prove an important lemma.

We consider prediction of the values of nonrecursive functions. It is easy to see that the number of errors should equal infinity. However, we can consider the initial fragments $f^{[n]}=\langle f(0),f(1),\ldots,f(n)\rangle$. By $F_{NV}(f^{[n]})$ we denote the number of errors made by the strategy F when predicting the first n values $f(1),f(2),\ldots,f(n)$.

A.N.Kolmogorov [Kol 65] introduced a fundamental notion of complexity of finite objects. According to this idea the complexity of a function in a fixed numbering of functions is the binary logarithm of its minimum index. In the class of all partial recursive functions of one argument x, as shown by Kolmogorov [Kol 65], there is an optimal numbering x such that, if φ is an arbitrary computable numbering of partial recursive functions, then there is a

constant c_φ such that for arbitrary partial recursive function f its complexity in κ does not exceed the complexity of f in φ plus c_φ.

We consider a counterpart of this complexity for numberings of total recursive functions. Note that there may exist no optimal (in this sense) numbering.

Let $\tau=\{\tau_i\}$ be an arbitrary computable numbering of total recursive functions. We consider the complexity of initial fragments of functions. By $k_\tau(f^{[n]})$ we denote the minimum τ-index of a function h such that $h^{[n]}=f^{[n]}$. By $K_\tau(f^{[n]})$ we denote $[\log_2 k_\tau(f^{[n]})]$. If f is nonrecursive, then $K_\tau(f^{[n]})\longrightarrow\infty$ with $n\longrightarrow\infty$. We try to find out a relation between $F_{NV}(f^{[n]})$ and $K_\tau(f^{[n]})$.

LEMMA 2.1. Let (U,τ) be an arbitrary enumerated class and $\eta(p)$ be a function such that $F^{NV}_{U,\tau}(p)\le\eta(p)$. Then for arbitrary (nonrecursive) function f and arbitrary n, $F_{NV}(f^{[n]})\le\eta(k_\tau(f^{[n]}))$.

PROOF. We have $F^{NV}_{U,\tau}(p)\le\eta(p)$. Hence for arbitrary p it is true that $F_{NV}(\tau_p^{[y]})\le\eta(p)$ for all y. Let $p_n=k_\tau(f^{(n)})$. Then for $x\le n$ we have $\tau_{p_n}(x)=f(x)$. Hence $F_{NV}(f^{[n]})=F_{NV}(\tau^{[n]}_{p_n})\le\eta(p_n)=\eta(k_\tau(f^{[n]}))$.

□

THEOREM 2.3. ([BF 74]) For arbitrary enumerated class (U,τ) and arbitrary positive integer k, there is a total recursive strategy F such that for arbitrary (nonrecursive) total function f and for all n,

$$F_{NV}(f^{[n]})\le K_\tau(f^{[n]})+\log_2 K_\tau(f^{[n]})+\ldots+\underbrace{\log_2\ldots\log_2}_{k\text{ times}}K_\tau(f^{[n]})+$$
$$+o(\underbrace{\log_2\ldots\log_2}_{k\text{ times}}K_\tau(f^{[n]})).$$

PROOF. Immediately from Lemma 2.1 and Theorem 2.1.

□

THEOREM 2.4. ([BF 74]) There is an enumerated class (U,τ) such that for arbitrary strategy F and arbitrary positive integer k:

1) $(\forall n)\ (F^{NV}_{U,\tau}(n)>\log_2 n-3)$,

2) $(\exists^\infty n)(F^{NV}_{U,\tau}(n)>\log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2}_{k\text{ times}}n)$

PROOF. We define two enumerated classes (V,τ') and (W,τ'') and then join them making the class $U=V\cup W$ and the numbering

$$\tau_n=\begin{cases}\tau'_k, & \text{if } n=2k-1,\\ \tau''_k, & \text{if } n=2k.\end{cases}$$

The enumerated class (V,τ') is constructed to have the property 1.

Let binary 0-1 strings be enumerated lexicographically. The infinite string of values $\tau'_i(0)\tau'_i(1)\tau'_i(2)\ldots$ is obtained from the i-th string in the lexicographical numbering by adding infinitely many zeros after the string. It is easy to see that $(\forall n)(F^{NV}_{V,\tau'}(n)\geq\log_2 n-2)$.

To construct the class (W,τ'') and to prove 2) we make use of the following theorem by P.Martin-Löf [ML 66] (see also [ZL 70]). Let $h(n)$ be an arbitrary total recursive function such that the series $\sum 2^{-h(n)}$ diverges. Then for every 0-1 valued function f it is true that

$$(\exists^{\infty}n)(K_B(f^{[n]})\leq n-h(n)).$$

In the abovecited theorem one can take, for instance, the function $h(n)=\log_2 n+\log_2 n+\ldots+\log_2\log_2\ldots\log_2 n+a(n)$, where $a(n)$ is a function growing to infinity sufficiently slowly.

The Martin-Löf theorem uses an optimal numbering B of partial recursive functions. Hence we cannot use this result directly. On the other hand, the proof of the theorem is based on the construction of an effective coding of initial fragments of sequences. The effectiveness of the coding allows us to construct a numbering $\sigma=\{\sigma_i\}$ of total recursive functions as well, such that

$$(\exists^{\infty}n)(K_\sigma(f^{[n]})\leq n-h(n)).$$

For (W,τ'') we take the numbering $\tau''=\sigma$ and the class W numbered by σ.

Assume from the contrary that

$$(\forall^{\infty}n)(F^{NV}_{w,\sigma}(n)\leq\log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2 n}_{k\text{ times}})$$

Hence there is a constant c such that

$$(\forall n)(F^{NV}_{w,\sigma}(n)\leq\log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2 n}_{k\text{ times}}+C).$$

We denote $\log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2 n}_{k\text{ times}}+C$ by $\eta(n)$ and use Lemma 2.1. We get

$$(\forall n)(F_{NV}(f^{[n]})\leq\log_2 k_\sigma(f^{[n]})+\log_2\log_2 k_\sigma(f^{[n]})+\ldots+$$
$$+\underbrace{\log_2\log_2\ldots\log_2 k_\sigma(f^{[n]})}_{k\text{ times}}+C)\leq K_\sigma(f^{[n]})+\log_2 K_\sigma(f^{[n]})+\ldots+$$
$$+\underbrace{\log_2\log_2\ldots\log_2 K_\sigma(f^{[n]})}_{k-1\text{ times}}+C'.$$

Up to now our function f was arbitrary. Now we take a specific

one, and, namely, we take the 0-1 valued function which is predicted incorrectly at every step. Thus $(\forall n)(F_{NV}(f^{[n]})=n)$. We have

$$(\forall n)(n\le K_\sigma(f^{[n]})+\log_2 K_\sigma(f^{[n]})+\ldots+\underbrace{\log_2\log_2\ldots\log_2}_{k-1\ \text{times}}K_\sigma(f^{[n]})+C').$$

On the other hand, from the modified Martin-Löf theorem we have

$$(\exists^\infty n)(K_\sigma(f^{[n]})\le n-\log_2 n-\log_2\log_2 n-\ldots-\log_2\log_2\ldots\log_2 n-a(n)).$$

Hence

$$(\exists^\infty n)(n\le(n-\log_2 n-\log_2\log_2 n-\ldots-\log_2\log_2\ldots\log_2 n-a(n))+$$
$$+\log_2(n-\log_2 n-\log_2\log_2 n-\ldots-\log_2\log_2\ldots\log_2 n-a(n))+$$
$$+\log_2\log_2(n-\log_2 n-\log_2\log_2 n-\ldots-\log_2\log_2\ldots\log_2 n-a(n))+$$
$$+\ldots+$$
$$\log_2\log_2\ldots\log_2(n-\log_2 n-\log_2\log_2 n-\ldots-\log_2\log_2\ldots\log_2 n-$$
$$-a(n))+C').$$

Contradiction.

□

We are going to prove the counterpart of Theorem 2.4 for identification in the limit. For this, we need a counterpart of Lemma 2.1.

By $G_{EX}(f^{[n]})$ we denote the minimum (over all functions g such that $g^{[n]}=f^{[n]}$) of $G^{EX}(g)$.

LEMMA 2.2. Let (U,τ) be an arbitrary enumerated class and $\eta(p)$ be a function such that $G^{EX}_{U,\tau}(p)\le\eta(p)$. Then for arbitrary (nonrecursive) function f and arbitrary n, $G_{EX}(f^{[n]})\le\eta(k_\tau(f^{[n]}))$.

PROOF. We have $G^{EX}_{U,\tau}(p)\le\eta(p)$. Hence for arbitrary p it is true that $G_{EX}(\tau_p^{[y]})\le\eta(p)$ for all y. Let $p_n=k_\tau(f^{[n]})$. Then for $x\le n$ we have $\tau_{p_n}(x)=f(x)$. Hence $G_{EX}(f^{[n]})=G_{EX}(\tau_{p_n}^{[n]})\le\eta(p_n)=\eta(k_\tau(f^{[n]}))$.

□

THEOREM 2.5. ([BF 74]) There is an enumerated class (U,τ) such that for arbitrary strategy G and arbitrary positive integer k:

1) $(\forall n)(G^{EX}_{U,\tau}(n)>\log_2 n-\text{const})$,

2) $(\exists^\infty n)(G^{EX}_{U,\tau}(n)>\log_2 n+\log_2\log_2 n+\ldots+\underbrace{\log_2\log_2\ldots\log_2}_{k\ \text{times}}n)$.

PROOF. As in proof of Theorem 2.4. we define two enumerated classes (V,τ') and (W,τ'') and then join them making the class $V\cup W$ and the numbering

$$\tau_n=\begin{cases}\tau'_k, & \text{if } n=2k-1,\\ \tau''_k, & \text{if } n=2k.\end{cases}$$

The enumerated class (W,τ'') is defined precisely as in the proof of Theorem 2.4, only instead of Lemma 2.1 we use Lemma 2.2.

The class V is a subclass of the one as in the proof of Theorem 2.4. Now we define the numbering τ'. With pairs (i,j) we associate 2^j τ'-indices. The corresponding functions are defined in such a way that the strategy F'_i (from Lemma 1.1) either makes on one of these functions no less than $\log_2 2^i=j$ mindchanges or does not identify at least one of these functions.

We divide the sequence of all nonnegative integers (the potential τ'-indices) into segments. The integers $2^k\le m<2^{k+1}$ make the segment S_{k+1}. Every segment is associated with a strategy from $\{F'_i\}$. Namely, the segments $S_0,S_2,S_4,S_6,S_8,\ldots$ are associated with F'_0. The segments $S_1,S_5,S_9,S_{13},\ldots$ are associated with F'_1. The segments $S_3,S_{11},S_{19},S_{27},\ldots$ are associated with F'_2, etc.

Thus we have the following property. If S_r and $S_{r+2^{i+1}}$ are two adjacent segments associated with the same strategy F'_i, and $d\in S_r, l\in S_{r+2^{i+1}}$, then l exceeds d no more than constant number of times. Every τ'-index in the segment $S_{r+2^{i+1}}$ does not exceed $2^{r+1+2^{i+1}}$. Our construction allows us to assert that at least one function f in S_r is such that $(F'_i)^{EX}(f)\ge r$. Hence, for every n from the segment $S_{r+2^{i+1}}$ or from the preceding segments, it is true that $(F'_i)^{EX}(\tau'_n)\ge\log_2 n-\text{const}$.

It remains to describe the functions in the segment S_{k+1} associated with F'_i. We define them in steps, first all the functions in the segment for x=0, then for x=1, x=2, x=3,.... For $0\le x\le i+k+1$ the functions are defined to encode i and k (the string of the first i+k+1 values equals 0^i10^k1). After that one half of the functions gets the current value 0 and the other half gets 1. The strategy F'_i is to change the hypothesis at least on one of these two functions. When it has changed the hypothesis for the corresponding indices we define again one half of the functions to be equal 0, and the other half to be equal 1, etc. Either there is a function in the segment which is not identified by F'_i or F'_i has at least k mindchanges.

□

3. Identification of τ-indices

The trivial strategies for prediction and identification in the limit provided by the proof of Theorem 1.1 were improved in Section 2. However, the counterpart of these improvements for identification of τ-indices was not proved there. We will show that such a counterpart is impossible.

THEOREM 3.1. ([Ba 74-1]) There is an enumerated class (U,τ) of total recursive functions such that for arbitrary total recursive strategy H there is a constant $c>0$ such that for all n (but a finite number of them)

$$H^{\tau}_{U,\tau}(n)>\frac{n}{c}.$$

PROOF. The construction of the class $U=\{\tau_0,\tau_1,\tau_2,\ldots\}$ is based on a diagonalization. At first we divide the sequence of all nonnegative integers (the potential τ-indices) into segments. The integers $2^k\le m<2^{k+1}$ form the segment S_{k+1}. Every segment is associated with a strategy from $\{F'_i\}$ (see Lemma 1.1). Namely, the segments S_0, S_2, S_4, S_6, $S_8,\ldots$ are associated with F'_0. The segments $S_1,S_5,S_9,S_{13},\ldots$ are associated with F'_1. The segments $S_3,S_{11},S_{19},S_{27},\ldots$ are associated with F'_2, etc.

Thus we have the following property. If S_j and $S_{j+2^{i+1}}$ are two adjacent segments associated with the same strategy F'_i, and $d\in S_j$, $l\in S_{j+2^{i+1}}$, then l exceeds d and the length of S_j no more than constant number of times.

Now we define the functions τ_m where $2^k\le m<2^{k+1}$, i.e. in the segment S_{k+1}. Let this segment correspond to F'_i. Then

$$\tau_m(x)=\begin{cases}1, & \text{if } x<i,\\ 0, & \text{if } x=i,\\ 1, & \text{if } i+1\le x\le i+k,\\ 0, & \text{if } x=i+k+1,\\ \text{to be defined below}, & \text{if } x>i+k+1.\end{cases}$$

Thus we have coded i and k into an initial fragment of the function.

Let $z>0$, and we define $\tau_m(i+k+1+z)$. We consider $F'_i(<\tau_m^{[i+k+z]}>)$ supposed to be the τ-index of τ_m. If $\tau_m^{[i+k+z]}=(1^i01^k0^z)$ and $F'_i(<\tau_m^{[i+k+z]}>)=m$, then we define $\tau_m(i+k+1+z)=1$ and $\tau_m(x)=1$ for all $x>i+k+1+z$.

Let τ_m $(2^k\le m<2^{k+1})$ be either a function with i+k values 1 only or the function of this segment which has no less zeros than any

other function τ_m in this segment. Then either F_i' does not identify its τ-index correctly or F_i' makes no less than 2^k-1 mindchanges. Thus we have proved that the worst-case mindchange complexity $(F_i')^{\tau}(\tau_m)$ in the segment S_{k+1} is no less than $2^k-1 \geq \frac{n}{2}$. Hence the worst-case mindchange complexity for the first segments $S_0, S_1, \ldots, S_r$ (where $k+1<r\leq k+1+2^{i+1}$) is no less than $2^k-1 \geq n/2^{2^{i+2}}$.

□

THEOREM 3.2. There is an enumerated class (U,τ) of total recursive functions such that, for arbitrary total recursive strategy H and for infinitely many n,

$$H^{\tau}_{U,\tau}(n)>n-o(\sqrt{n}).$$

PROOF differs from the proof of Theorem 3.1 only in the length of the segments. Now the length of the segment S_k is 2^{2^k}. Hence the length and the worst-case mindchange complexity of every segment is no less than the square of the total length of all of the preceding segments.

Infinitely many segments are associated with every strategy F_i'. The functions in these segments which are the most complicated for identification of τ-indices by F_i' provide the needed complexity bound.

□

THEOREM 3.3. For arbitrary enumerated class (U,τ) of total recursive functions and for arbitrary constant $c>0$ there is a total recursive strategy H such that for infinitely many n,

$$H^{\tau}_{U,\tau}(n)<\frac{n}{c}.$$

PROOF. We denote by ρ the real number $\rho=\limsup \frac{d_n}{n+1}$, where d_n is the number of pairwise distinct functions among $\tau_0, \tau_1, \tau_2, \ldots, \tau_n$. The number ρ needs not to be a constructive real number but it can be approximated by rationals.

It is possible to find effectively infinitely many n such that $\rho-\varepsilon \leq \frac{d_n}{n+1} \leq \rho+\varepsilon$. Let $n_1, n_2, n_3, \ldots$ be effective increasing sequence of such n's. Such that for arbitrary k, $n_k > 2^{n_{k-1}}$.

The strategy H searches the τ-index for the given function f, first, among $\tau_0, \tau_1, \ldots, \tau_{n_1}$. It begins with computing the initial segments of $\tau_0, \tau_1, \ldots, \tau_n$ until $\geq(\rho-\varepsilon)(n_1+1)$ distinct functions are found. Then with no more than $2\varepsilon\cdot(n_1+1)$ mindchanges the strategy either stabilizes to the correct output or finds out that f is not

in this initial segment. In the latter case the strategy H goes on to search the τ-index among $\tau_0,\tau_1,\ldots,\tau_{n_2}$, and so on.

Any case, the total number of mindchanges does not exceed $n_k\cdot 2\varepsilon$ for every function among $\tau_0,\tau_1,\ldots,\tau_n$. For $n \in \{n_k-[n_k/2], n_k-[n_k/2]+1, \ldots, n_k\}$ this makes no more than $n\cdot\varepsilon$ mindchanges.

□

4. Influence of the numbering

We have proved several lower bounds in Sections 2 and 3. We prove in this section that most of these lower bounds express the complexity of the numbering rather than the complexity of the class of functions.

THEOREM 4.1. ([BKP 74]) If the class U of total recursive functions has a numbering τ such that the property $(\tau_i\equiv\tau_j)$ is decidable, then, for arbitrary total recursive function g(n) which nondecreasingly grows to the infinity, there is a strategy H identifying in the limit τ-indices of U such that $H^{\tau}_{U,\tau}(n)\le g(n)$ for all n.

PROOF. Let $n_1,n_2,n_3,\ldots$ be the sequence of the least numbers such that $g(n_i)\ge i$. The strategy computes initial fragments of $\tau_0,\tau_1,\ldots,\tau_{n_1}$ sufficiently long until all functions which are different (as shown by the decidable property) really turn out to be different. Then solely one of these functions can be equal to the function under identification. The first hypothesis (with insufficient information about the function) is 0, and the second hypothesis is the abovementioned sole function in the segment.

If the function turns out to be this function, then the only suitable function is found among $\tau_0,\tau_1,\ldots,\tau_{n_2}$ (at cost of one additional mindchange), and so on.

□

COROLLARY. If the class U of total recursive functions has a numbering τ such that the property $(\tau_i\equiv\tau_j)$ is decidable, then for arbitrary total recursive function g(n) which nondecreasingly grows to the infinity there is a strategy G identifying U in the limit such that $G^{EX}_{U,\tau}(n)\le g(n)$ for all n.

PROOF. Immediately from Theorems 4.1, 1.2 and 1.3.

□

For the contrast, we note that the counterpart of Theorem 4.1 for the prediction fails.

THEOREM 4.2. ([BKP 74]) If for an enumerated class (U,τ) it is true that for arbitrary total recursive function $g(n)$ which nondecreasingly grows to the infinity there is a strategy F predicting U such that $F^{NV}_{U,\tau}(n)\leq g(n)$ for all n, then there is a nonrecursive strategy K such that $K^{NV}_{U,\tau}(n)=o(1)$.

PROOF. We use the term "pxq table of (U,τ)" for the table of values $\tau_i(x)$ with $i\leq p$, $x\leq q$. All possible strategies H provide us only a finite number of variants which function isto prefered when predicting values for $\tau_0,\tau_1,\ldots,\tau_p$ and for $x\leq q$. All these variants can be enumerated and a number $S(p,q)$ be found such that:

a) arbitrary strategy H makes no less than $S(p,q)$ errors at a line of the (pxq)-table of (U,τ),

b) there is a strategy H_0 which makes at an arbitrary line of the (pxq)-table of (U,τ) no more than $S(p,q)$ errors.

Evidently, $S(p,q)$ is a total recursive function which is monotonic both in p and q. It is easy to see that

$$(\forall p)(\forall q)H^{NV}_{U,\tau}(p)\geq S(p,q) \qquad (4.1)$$

Since there is a total recursive strategy F with the property $F^{NV}_{U,\tau}(p)\leq p$, we conclude that for a fixed p the function $S(p,q)$ is bounded. Indeed, if $S(p,q)$ were unbounded, then it would be possible to find a total recursive function $t(p)$ such that $S(p,t(p))\longrightarrow\infty$ monotonically. By (4.1), this contradicts the provisions of the theorem.

We have proved $(\forall p)(\forall q)(S(p,q)\leq C)$. Now we can prove the existence of the needed strategy K.

The inequality $S(q,q)\leq C$ implies that, for every q the set $\tilde{H}_q$ of those strategies which make no more than C errors within the (qxq)-table of (U,τ), is nonempty. The set $\tilde{H}_q$ is divided into a finite system of equivalence classes where one class consists of strategies which function equally within the (qxq)-table of (u,τ). We denote this system by $\{\tilde{H}^1_q,\ldots,\tilde{H}^{kq}_q\}$. It is easy to see that

$$(\forall k\leq k_{q+1})(\exists l\leq k_q)(\tilde{H}^k_{q+1}\subseteq\tilde{H}'_q).$$

Hence from the compactness theorem for trees with the finite branching property, there is a strategy H such that

$$(\forall q)(\exists k\leq k_q)(H\in\tilde{H}^k_q)$$

or just $H \in \tilde{H}_q$ for all q . Thus H makes no more than C errors on every (qxq)-table of (U,τ), and $H^{NV}_{U,\tau}(q)=o(1)$.

□

5. Prediction and identification of finite automata

We saw in Section 2 that prediction and identification in the limit can be performed with a small number of errors (resp.,mindchanges). Section 3 contained disappointing results (Theorems 3.1 and 3.3) showing that for identification of τ-indices many mindchanges may be inevitable. On the other hand, we saw in Section 4 that the negative results just indicate that these are numberings which are complicate, not the classes of functions. Now we are about to ask whether "natural" numberings make identification easy or complicate.

For arbitrary classes of functions it is not possible to answer such a question since we do not know the criteria according to which numberings could be called "natural". Nevertheless, there is a happy exception. There are classes of objects that can be considered as recursively enumerable classes of total recursive functions, and simultaneously they have nontrivial natural numberings, the naturalness of which is widely accepted. We are talking about finite automata.

Finite automata were intensively studied in the fifties, and the pioneering paper [Moo 56] was a starting point in several directions of research, inductive inference including. Hence it is natural to consider such an example.

Initial finite automata with input and output are considered. The input alphabet is fixed X={1,2,...,a}. The output alphabet may vary. We restrict it only to be a subset of {1,2,...,n}. The class of all such automata is denoted by U_a. The subclass of U_a obtained by fixing the output alphabet to be Y={1,2,...,b} is denoted by $U_{a,b}$.

Automata are considered as "black boxes". We know only that they are in U_a. Let the sequence of the inputs of such an automaton A be

$$\omega=\{x(1), x(2), \ldots, x(t), \ldots\},$$

and

$$\{y(1), y(2), \ldots, y(t), \ldots\}$$

be the corresponding output sequence. The problem is, for an arbitrary t, given {x(1), ..., x(t)}, {y(1), ..., y(t)}, x(t+1), to predict y(t+1). Arbitrary effective rules (called strategies) are allowed.

We see that the problem cannot be solved without errors. We study the minimal number of errors needed for such a prediction. The main result of this section shows that the worst-case number of errors can be very small, namely, $o((a-1)\cdot k\cdot \log_2 k)$ for automata with k states and this estimate cannot be asymptotically improved. Note that any exhaustive search gives the upper bound of k^k type.

Let Σ be a strategy, i.e. a total recursive function of one argument. We say that Σ commits an error at moment t working on the sequence ω and the automaton A, if

$$\Sigma(<x(1),\ldots,x(t),y(1),\ldots,y(t),x(t+1)>)\neq y(t+1).$$

$\Sigma^*(\omega,A)$ is the cardinality of the set of those t when Σ commits an error at work on ω and A. For arbitrary class U of automata

$$\Sigma^*(\omega,U,k)=\max \Sigma^*(\omega,A),$$

where the maximum is taken over all automata $A\in U$ with no more than k states.

THEOREM 5.1. ([Ba 74-2]) Let $a\geq 2$. There is a strategy Σ such that for arbitrary input sequence ω,

$$\Sigma^*(\omega,U_a,k)\leq (a-1)k\cdot\log_2 k+o((a-1)\cdot k\cdot\log_2 k).$$

PROOF. Instead of automata from U_a we consider the corresponding automata graphs (see [TB 72]) with input alphabet $X=\{1,\ldots,a\}$. We take one representative per class of isomorphic graphs (isomorphism for graphs with a fixed initial vertice is considered). We order these representatives by the number of vertices. We remove the graphs for which the part reachable from the initial vertice coincides with a graph considered earlier (such graphs do not generate new automata operators). The graphs with the same number of vertices are ordered arbitrarily. We get a sequence of graphs $\mathcal{G}=(G_1,G_2,\ldots,G_i,\ldots\}$. Evidently, if the number of vertices $|G_i|$ in the graph G_i does not exceed k, then

$$i\leq I(a,k), \tag{5.1}$$

where I(a,k) is the number of all pairwise nonisomorphic initial automata graphs with k vertices and a-letter input alphabet. It follows from [Kor 67] that

$$I(a,k) \sim \begin{cases} \frac{k^{ak}}{k!}\cdot k, & \text{if } a\geq 3, \\ \frac{1}{e^{2\cdot e^{4}}}\cdot\frac{k^{ak}}{k!}\cdot k, & \text{if } a=2. \end{cases} \qquad (5.2)$$

(Since we consider initial automata graphs, we have the multiplier k in (5.2), in contrast to the original version of the formula in [Kor 67]).

Following the idea of the proof of Theorem 2.1 we associate weights

$$p(G_i)=\frac{C_0}{i(\log_2(i+1))^2} \qquad (5.3)$$

to graphs G_i. (Here the constant C_0 is chosen to have $\sum p(G_i)=s_0<1$. It is easy to see that the series converge effectively.)

First, we construct a nonrecursive "strategy" $\bar{\Sigma}$ which provides the needed complexity bound. This strategy in the computation process observes all the infinite sequence $\mathscr{G}$. Next, we use the effective convergence of $\sum p(G_i)$ and modify this "strategy" making it recursive.

The "strategy" $\bar{\Sigma}$ is described as a sequential process of predicting which ascribes output letters to the edges of the graph (thus converting the graph into an automaton). The "strategy" crosses out the graphs which have turned out to be inconsistent with the input x(1),...,x(t) and output y(1),...,y(t). Let the path x(1)...x(t) in the graph G be the path starting in the initial vertice and following the input word x(1)...x(t).

We start the prediction at t=1 when we are to predict y(2) by x(1), y(1), x(2). For the starting sequence of automata graphs we take the sequence $\mathscr{G}=\{G_1^0,\ldots,G_i^0,\ldots\}$ which is essentially the same $\mathscr{G}$, only on the edges outgoing from the initial vertice and labelled by input letter x(1) the output symbol y(1) is written. The weights of the automata graphs remain the same as before. This way, we get a sequence $\mathscr{G}=\{G_1^1,\ldots,G_i^1,\ldots\}$ with ascribed weights.

At the stage t we have the information x(1),...,x(t), y(1),...,y(t),x(t+1). We take the sequence $\mathscr{G}^{t-1}=\{G_1^{t-1},\ldots,G_i^{t-1},\ldots\}$ produced at the previous stage. All graphs in this sequence have output letters y(1), y(2),...,y(t) written on the edges of the path x(1) x(2) ... x(t), and no edges have been ascribed contradicting letters. In the general case $\mathscr{G}^{t-1}$ may have not all automata graphs,

since some of the graphs may contradict the existing information on the input-output relation. In other terms, if G_i^{t-1} is considered as a partially defined automaton, then it produces $y(1)...y(t)$ as its response to $x(1)...x(t)$ and goes to the state $g_t=g_1x(1)...x(t)$.

We say that G_i^{t-1} at input $x(t+1)$ outputs y if the edge outgoing g_t and corresponding $x(t+1)$ is on the path $x(1)...x(t)$ and has the output symbol y. If G_i^{t-1} produces an output symbol in response to $x(t+1)$, i.e., if the edge $x(t+1)$ from g_t is on the path $x(1)...x(t)$, then we say that G_i^{t-1} participates the prediction.

Additionally, the elements of $\mathscr{G}^{-1}$ have got weights $p(G_i^{t-1})$ and the total $\sum p(G_i^{t-1})=S_0<1$. The "strategy" $\bar{\Sigma}$ predicts the output symbol with the maximal total weight.

To complete the description of the current stage t we have to say that the new information is used to transform $\mathscr{G}^{-1}$ into $\mathscr{G}$. The output symbol $y(t)$ is ascribed to the edge of the graph corresponding to $x(t)$ on the path $x(1)...x(t-1)x(t)$. If this output symbol contradicts to the earlier information for this graph, then the graph is removed from the sequence.

The new weight is defined as follows. If the graph has not participated in the prediction, then its weight is not changed. If the graph has participated and has not been removed, then its weight is multiplied to s_t/r_t, where s_t is the total of weights of the automata having participated in the prediction and r_t is the total of weights of the automata having produced the right outcome. Evidently, the total of weights over all the sequence $\mathscr{G}$ has not changed, i.e. $\sum p(G_i^t)=S_0$.

Note that, if $\bar{\Sigma}$ has made an error, then

$$\frac{S_t}{r_t}\geq 2. \qquad (5.4)$$

Hence, every graph having produced a right prediction at least doubles its weight.

Let G_α be the first graph in the sequence which is consistent with the input-output information. At every moment of error, either G_α gets a new output symbol or doubles its weight. Hence the maximal number of errors does not exceed a number z such that $2^{z-ak}\cdot p(G_\alpha)=1$. From this equality, using (5.1), (5.2), (5.3), we can get

$$\bar{\Sigma}^*(\omega,U_a,k) < z \leq (a-1)k\cdot\log_2 k. \qquad (5.5)$$

It remains to modify $\bar{\Sigma}$ and to get a recursive strategy Σ which computes the infinite series only approximately and does about the

same as $\bar{\Sigma}$. We use the constructive convergence of the series of weights. This allows us to consider only finite initial fragments of this series. The strategy Σ predicts y only when it has checked that any other output symbol y′ may have the total of weights

$$p(y')\le p(y)+\frac{2}{j_t+1}\cdot p(y), \qquad (5.6)$$

where j_t is the number of errors already made up to this moment (instead of $p(y')\le p(y)$ for $\bar{\Sigma}$). This modification does not influence (5.5). □

THEOREM 5.2. ([Ba 72-2]) Let $a\ge 2$. There exists an input sequence ω_0 such that for every strategy Σ and for every $b\ge 2$

$$\Sigma^*(\omega_0,U_{a,b},k)\ge(a-1)\cdot k\cdot\log_2 k+o((a-1)\cdot k\cdot\log_2 k)$$

(consequently, $\Sigma^*(\omega_0,U_a,k)\ge(a-1)\cdot k\cdot\log_2 k+o((a-1)\cdot k\cdot\log_2 k)$).

PROOF. Let $X=\{x_1,\dots,x_a\}$ be an input alphabet and $Y=\{0,1\}$ be an output alphabet. Given any natural number $k\ge 64$, we define the automata class R_k as follows. A typical automaton in R_k is drawn in Fig.5.1 (containing only those arrows essential for further considerations). As it is shown in Fig.5.1, automata in R_k have

$$s+\mu\alpha=2[\log_2 k]+6+2^{[\log_2 k-\log_2\log_2 k]}\cdot[\log_2 k-\log_2\log_2 k]=k-o(k)$$

states. First s-1 states specify a subautomaton called k-*encipherator* (the same for all automata in R_k), the next k states form a different subautomaton called the *main*.

First we give the formal description of k-encipherator. Given the binary representation of the number k, we replace every occurrence of the symbol 1 by the word x_2x_1, replace every symbol 0 by the word x_1x_2 and add $x_1x_1x_1$ to the end of the word obtained so far. Let $\bar{k}$ denote the word we have obtained. Apparently, $s=2[\log_2 k]+6$ is the length of $\bar{k}$; let $\bar{k}=\nu_1,\nu_2,\dots,\nu_s$. The word $\bar{k}$ contains no subword $x_1x_1x_1$. $\bar{k}$-encipherator is supposed to "let through" (to the main subautomaton) only the words containing a subword $\bar{k}$, provided it starts updating in the initial state q_1. The definitions of k-encipherator (see Fig.5.1) and the word $\bar{k}$ imply that, provided x_1 repeated tree times preceeds $\bar{k}$, k-encipherator will reach the state q_1 and will stay in this state while x_1 is on input.

Now we describe the main subautomaton. It consists of many distinct blocks. The i-th block begins with the state $q_{s+i\alpha}$ (initial states in Fig.5.1 are marked by *), the length of each block (i.e.,

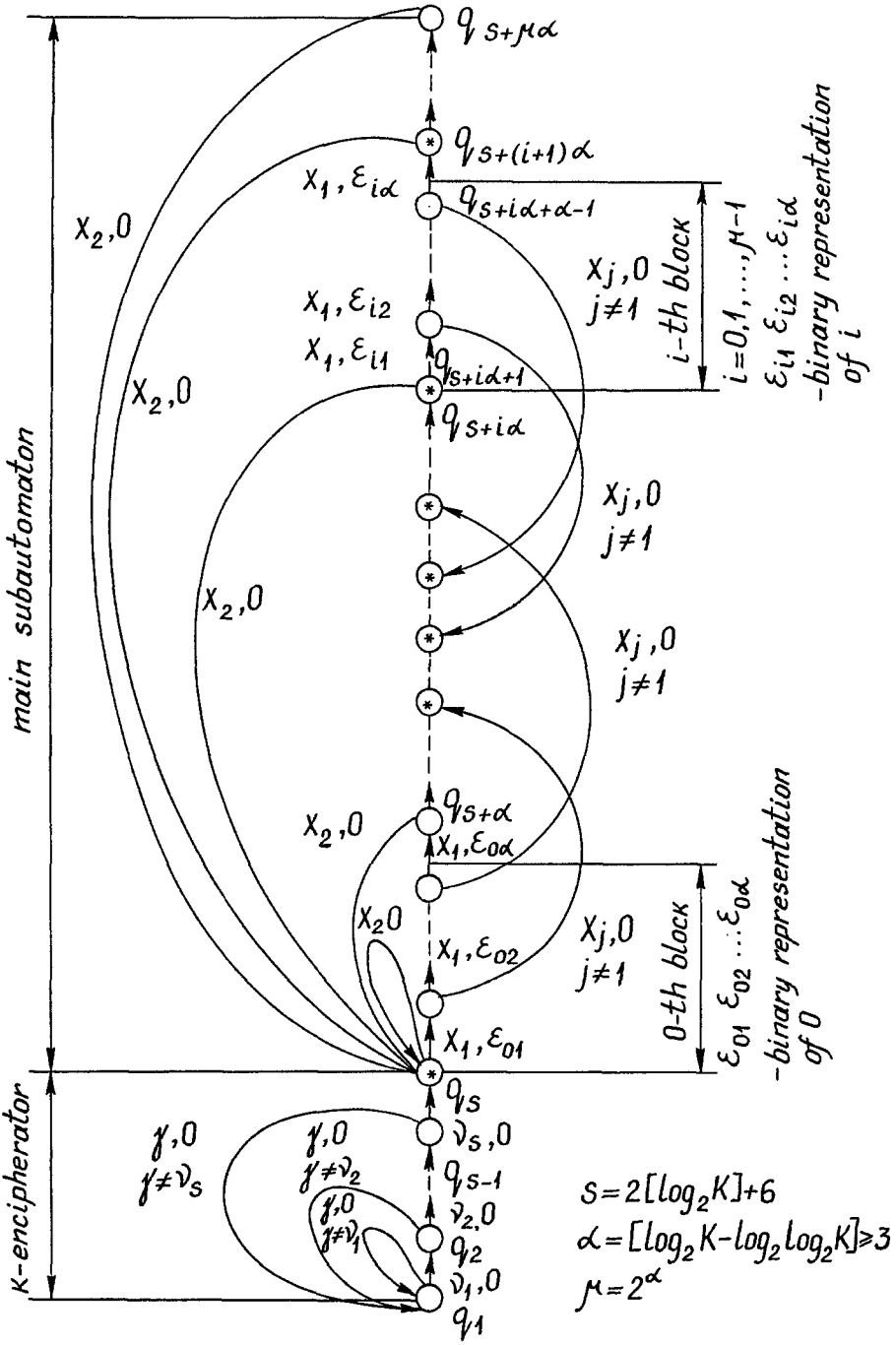

main subautomaton
K-encipherator
i-th block
0-th block
i=0,1,...,μ-1
ε_i1 ε_i2 ...ε_iα
-binary representation of i
ε_01 ε_02 ...ε_0α
-binary representation of 0
S=2[log_2 K]+6
α=[log_2 K-log_2 log_2 K]≥3
μ=2^α

Fig.5.1

the number of states) is equal to $\alpha=[\log_2 k-\log_2\log_2 k]\geq 3$, the total number of blocks is $\mu=2^{\alpha}\leq\frac{k}{\log_2 k}$. The output labels on the arrows from the states in the i-th block labelled by x_1 form the binary word $\varepsilon_{i1}\varepsilon_{i2}\ldots\varepsilon_{i\alpha}$ that is the binary representation of the number i (containing so many zeros in the beginning that the total length is α).

The word $\varepsilon_{i1}\varepsilon_{i2}\ldots\varepsilon_{i\alpha}$ is said to be the characteristic sequence of the block i. Hence, every block specifies its own characteristic sequence. Note, that the number of distinct binary words of the length α is just equal to the number of blocks; therefore, every binary word of the length α is the characteristic sequence for some block. Arrows outgoing from the initial states of the blocks (i.e., from the states $q_{s+i\alpha}$, $i=0,1,\ldots,\mu$) and labelled by the input symbol x_2 link initial states with the state q_s. Arrows outgoing from inner states of blocks and labelled by input symbols differing from x_1 (call these arrows *variable* ones) link these states with arbitrary initial states of blocks (there are 2^{α} states of this kind). Just the latter property differs any automaton in R_k from any other.

Now we consider variable arrows. The total number of these arrows is equal to $u=(a-1)\mu(\alpha-1)$. Let us fix a linear ordering of these arrows: $d_1,d_2,\ldots,d_u$. Given any main subautomaton, associate with it the binary sequence

$$\delta_{11},\ldots,\delta_{1\alpha},\ldots,\delta_{j1},\ldots,\delta_{j\alpha},\ldots,\delta_{u1},\ldots,\delta_{u\alpha}$$

of the length αu defined as follows: $\delta_{j1},\ldots,\delta_{j\alpha}$ is the characteristic sequence of the block having the initial state the arrow d_j goes to. This sequence is called the characteristic sequence of the given main subautomaton. It is easy to see (taking into account values of α and μ) that every binary sequence of the length αu is the characteristic sequence for some main subautomaton.

Now we define one specific input sequence. Let d'_j stand for the input symbol labelling d_j, and V_j be the sequence "transferring" q_s to the state the arrow d_j is outgoing from. We set

$$D_k=\{V_1,d'_1,\underbrace{x_1,\ldots,x_1}_{\alpha\ \text{times}},x_2,\ldots,V_j,d'_j,\underbrace{x_1,\ldots,x_1}_{\alpha\ \text{times}},x_2,\ldots,$$

$$V_n,d'_u,\underbrace{x_1,\ldots,x_1}_{\alpha\ \text{times}}\}.$$

Consider, for a while, the main subautomaton as an independent automaton with the initial state q_s. For input string D_k the automaton outputs the sequence

$$E=\{W_1,0,\delta_{11},\ldots,\delta_{1\alpha},0,\ldots,W_j,0,\delta_{j1},\ldots,\delta_{j\alpha},0,\ldots$$

$$\ldots,W_u,\delta_{u1},\ldots,\delta_{u\alpha}\},$$

where W_j is the output sequence corresponding to the fragment V_j and $\delta_{j1},\ldots,\delta_{j\alpha}$ corresponds to the piece $\underbrace{x_1,\ldots,x_1}_{\alpha \text{ times}}$ directly following V_j. The subsequence

$$\delta_{11},\ldots,\delta_{1\alpha},\ldots,\delta_{j1},\ldots,\delta_{j\alpha},\ldots,\delta_{u1},\ldots,\delta_{u\alpha}$$

of the sequence E is, obviously, the characteristic sequence of the given main subautomaton.

Let Σ be any strategy. Now it is not difficult to show that there is a main subautomaton A_Σ that, provided A_Σ is treated as an independent subautomaton with the initial state q_s, the strategy Σ, being applied to the automaton A_Σ and the input sequence D_k, will make mistakes just in those places corresponding to the fragments $\underbrace{x_1,\ldots,x_1}_{\alpha \text{ times}}$ of D_k, i.e., for every j and $1\le l\le\alpha$, $1\le j\le u$, the inequality

$$\Sigma\{V_1,d_1',\underbrace{x_1,\ldots,x_1}_{\alpha \text{ times}},x_2,\ldots,V_j,d_j',\underbrace{x_1,\ldots,x_1}_{l-1 \text{ times}};$$

$$_1,0,\ \delta_{11},\ldots,\delta_{1\alpha},0,\ldots,W_j,0,\delta_{j1},\ldots,\delta_{j(l-1)},x_1)\neq\delta_{j\alpha}$$

will hold.

This inequality shows how the characteristic sequence of the required automaton A_Σ should be defined. Furthermore, given the characteristic sequence, one can easily restore unambiguously the automaton A_Σ. Hence,

$$\Sigma^*(D_k,A_\Sigma) \ge u\alpha = (a-1)\mu(\alpha-1)\alpha.$$

Finally, we are able to define the required input sequence:

$$\omega_0=\{\overline{k}_0,D_{k_0},(\overline{k_0+1}),D_{k_0+1},\ldots,\overline{k},D_k,\ldots\},\quad k_0=64.$$

Let A be an arbitrary automaton in R_k. As it follows from the definition of k-encipherator the automaton A reaches for the first time the state q_s on the input string ω_0 just after the initial fragment

$$\omega_0=\{\overline{k}_0,D_{k_0},\ldots,D_{k-1},\overline{k}\}.$$

Before A has reached q_s, k-encipherator runs on $\omega_0(k)$ (let $\Omega_0(k)$ denote the sequence k-encipherator outputs on $\omega_0(k)$). The sequence ω_0 is constructed so that D_k follows $\omega_0(k)$. Therefore, after the string $\omega_0(k)$ is updated, the main subautomaton can be considered as an independent automaton with the initial state q_s, input string D_k and prediction strategy according to

$$\Sigma'(\varphi,\xi;x_i)=\Sigma(\omega_0(k),\xi;\Omega_0(k),\xi;x_i).$$

Let us choose $A_{\Sigma'}$ as the main subautomaton for A. Then, clearly,

$$\Sigma^*(\omega_0,A)\geq\Sigma'^*(D_k,A_{\Sigma'})\geq(a-1)\mu(\alpha-1)\alpha.$$

Therefore, $\Sigma^*(\omega_0,U_{a,2},s+\mu\alpha)\geq(a-1)\mu(\alpha-1)\alpha$. Using values of s,α,μ, we obtain

$$\Sigma^*(\omega_0,U_{a,2},k)\leq(a-1)k\cdot\log_2 k+o((a-1)k\cdot\log_2 k).$$

□

One can consider identification in the limit of automata instead of prediction of their behaviour. In this case, given a pair $\{x(1),\ldots,x(t)\}$, $\{y(1),\ldots,y(t)\}$, one has to construct an automaton A′ non-distinguishable from the "black box" A on the string $\omega=\{x(1),\ldots,x(t),\ldots\}$. Let $\{A_\omega\}$ stand for the class of all such A′. Strategy Σ in this case is a general recursive function which, given any string $\{x(1),\ldots,x(t)\}$, $\{y(1),\ldots,y(t)\}$, finds an automaton in U_a (more precisely, given the number of the string, it finds the number of an automaton in U_a).

$$A_t=\Sigma(x(1),\ldots,x(t);y(1),\ldots,y(t))$$

is said to be the hypothesis generated at the moment t. Let us suppose that

a) for every t, the automaton A_t transforms the input word $x(1)\ldots x(t)$ into $y(1)\ldots y(t)$, i.e. A_t is not an "explicitly" incorrect guess;

b) there exists t such that $A_t=A_{t+1}=\ldots=A'$ and $A'\in\{A_w\}$.

Then we say that the strategy Σ identifies in the limit the automaton A on the sequence ω.

By $\Sigma^{\#}(\omega,A)$ we denote the number of mindchanges, i.e. the number of moments when the automaton produced at this moment differs from the automaton produced at the previous moment. Additionally, $\Sigma^{\#}(\omega,A)=\infty$ if the strategy Σ does not identify in the limit the automaton A on ω. By analogy, we define $\Sigma^{\#}(\omega,U_a,k)=\max\Sigma^{\#}(\omega,A)$, where the maximum is taken over all automata $A\in U_a$ with no more than k states.

THEOREM 5.3. ([Ba 74-2]) Let $a\geq 2$. There exists a strategy Σ such that for every input sequence ω

$$\Sigma^*(\omega,U_a,k)\leq(a-1)k\cdot\log_2 k+o((a-1)k\cdot\log_2 k).$$

PROOF. Instead of the "strategy" $\bar{\Sigma}$ from the proof of Theorem 5.1 we use a "strategy" $\bar{\Sigma}'$ which differs only in one aspect. The "strategy" $\bar{\Sigma}'$ changes the sequence $\mathscr{G}$ only at the moments when an

error is made. It is easy to see that the estimate (5.5) remains valid since it was proved actually using only those moments t when the strategy fails (i.e. the inequality (5.4) holds). Let $t_1,t_2,\ldots t_n$ denote moments when errors were made, $n=\bar{\Sigma}'^{*}(\omega,A)$. Therefore, our strategy $\bar{\Sigma}'$ will use only subsequences $\mathscr{G}, \mathscr{G}^{t_1}, \mathscr{G}^{t_2},\ldots,\mathscr{G}^{t_n}$.

Now we are about, given the strategy $\bar{\Sigma}'$, to define an effective strategy Σ'. The symbol y, Σ' outputs at the moment t, has to satisfy the inequality 5.6. Let $t\in(t_i,t_{i+1}]$. Then inequality is transformed to

$$p(y')\leq p(y)+\frac{2}{i}\,p(y). \qquad (5.7)$$

For any given $t\in(t_i,t_{i+1}]$, the symbol y can be defined using at most an initial fragment of the sequence $\mathscr{G}^{t_i}$. This fragment is said to be essential for the given moment t. Taking into account constructive convergence of the series $\sum_j p(\mathscr{G}_j)$ one can show easily that it can be effectively computed, given the pair $\{x(1),\ldots,x(t)\}$, $\{y(1),\ldots,y(t)\}$. Note, that, if an initial fragment, essential for the moment t, is long enough, then it can be equally essential for the next moment, and so on. Now, let $t_{i_1},t_{i_2},\ldots,t_{i_n}$ be the moments in $(t_i,t_{i+1}]$ when one has to change (i.e. to make longer) the essential initial fragment chosen earlier (in order to make it possible to check the inequality (5.7)). Note, furthermore, that, if an essential initial fragment containing the required graph G_α is found and this fragment contains a sufficiently long "tail" after G_α, then at least the inequality (5.7) protects it from replacement (it will be changed when an error is made, and $\mathscr{G}$ is to be changed itself). This consideration implies that, if we choose every next essential initial fragment sufficiently longer than the preceding one (for instance, of the length 2^n, where n is the length of the preceding fragment), then the total number of changes of essential initial fragments implied by inequality (5.7) will not exceed $o(|G_\alpha|\log_2|G_\alpha|)$. On the other hand, the number of changes of essential initial fragments implied by changes of the sequence $\mathscr{G}$ is equal to the number of $\mathscr{G}$ changes, i.e. the number of errors Σ' makes on the input string. The latter number, as it follows from the proof of Theorem 5.1, does not exceed $(a-1)k\cdot\log_2 k+o(k\cdot\log_2 k)$. We obtain now that our strategy Σ' changes essential initial fragments at most

$$(a-1)k\cdot\log_2 k+o(k\cdot\log_2 k)+o(|G_\alpha|\log_2|G_\alpha|)=$$
$$=(a-1)k\cdot\log_2 k+o(k\cdot\log_2 k)$$

times. While essential initial fragment is not changed the strategy Σ' predicts the next value using only this fragment and the current vertex of each graph from the fragment. Namely, it means the following. The current vertex of G_j at the moment t is just the vertex the automaton reaches reading $x(1)\ldots x(t)$ from the initial state. Therefore, if we know the current vertex of the graph G_j, then we can find the symbol y G_j (as an automaton) outputs reading $x(t+1)$; there is no need to store information reflecting the word $x(1)\ldots x(t)$.

It means that a finite automaton is able to perform prediction which Σ' is making while essential initial fragment is not changed. The states of the required automaton are all possible orders of current vertices in the chosen initial fragments (i.e., each state is a chosen initial fragment, where just a single vertex, called current, is marked in every graph; the choice of current vertices distinguishes one state from the other). Transition from one state to another is performed according to the transition of current vertices in every graph while reading x. The automaton outputs the symbol the strategy Σ' is supposed to output in the given case.

The above automaton is just the hypothesis the required strategy Σ is suppoosed to guess during the timefragment under consideration. Evidently, the number of hypothesis changes is equal to the number of changes of essential initial fragments. Therefore,

$$\Sigma^{\#}(\omega,A)\le(a-1)k\cdot\log_2 k+o((a-1)\cdot k\cdot\log_2 k).$$

□

The lower bound proved in Theorem 5.2, clearly, holds in the given case too.

The cases considered above resemble in a way simple experiment. Now we consider the case which resembles multiple experiment. Let the sequence

$$\Omega=\{\varphi_1,\varphi_2,\ldots,\varphi_t,\ldots\}$$

be used as an input for a "black box" A and $\{\eta_1,\eta_2,\ldots,\eta_t,\ldots\}$ is the corresponding sequence of output words (A reads every new word starting from the initial state).

Prediction by the 3-tuple $\{\varphi_1,\ldots,\varphi_t\}$, $\{\eta_1,\eta_2,\ldots,\eta_t\}$, φ_{t+1} means prediction of η_{t+1}. In our case $\Sigma^*(\Omega,A)$ is the number of distinct t such that

$$\Sigma(\varphi_1,\ldots,\varphi_t;\eta_1,\eta_2,\ldots,\eta_t;\varphi_{t+1})\neq\eta_{t+1}.$$

Given any pair $\{\varphi_1,\ldots,\varphi_t\}$, $\{\eta_1,\eta_2,\ldots,\eta_t\}$, the goal of the identification in the limit is to define an automaton A' non-distinguishable from A on the input words Ω. $\Sigma^{\#}(\Omega,A)$ is defined like $\Sigma^{\#}(\omega,A)$, but the words φ_t,η_t are used instead of x(t) and y(t) respectively, and the hypothesis A_t is defined as $\Sigma(\varphi_1,\ldots,\varphi_t;\eta_1,\eta_2,\ldots,\eta_t)$.

Extending slightly proofs of Theorem 5.1 and Theorem 5.3, we obtain the following, slightly more general results.

THEOREM 5.1'. ([Ba 74-2]) Let $a\geq 2$. There exists a strategy Σ such that for every sequence Ω of input words

$$\Sigma^{*}(\Omega,U_a,k)\leq(a-1)k\cdot\log_2 k+o((a-1)k\cdot\log_2 k).$$

THEOREM 5.3'. ([Ba 74-2]) Let $a\geq 2$. There exists a strategy Σ such that, for every sequence Ω of input words,

$$\Sigma^{\#}(\Omega,U_a,k)\leq(a-1)k\cdot\log_2 k+o((a-1)k\cdot\log_2 k).$$

Theorem 5.3' is a very important tool for investigation of the synthesis of programs by hystories of their behaviour (see Section 6).

6. Notes on program synthesis from computational hystories

One of the most important problems in the theory of learning evidently is program synthesis from computational histories. Note, that even learning of such algorithms as addition and multiplication usually proceeds as follows: the teacher demonstrates how the algorithm is working on particular samples, i.e., gives the histories of computation and then the learners are synthesizing general algorithm (program) on the basis of this information themselves. In 1972 Bierman [Bie 72] proposed heuristic algorithms of synthesis from computational histories and implemented them on computer. Still the mathematical basis of the process of such synthesis have not been studied much at the time. Below we give the first results in this field we obtained in 1974 (first published in [Ba 74-3]).

As a model we consider the Post machine. All the results can be easily transformed for more general programming languages (to within multiplying constants in evaluations).

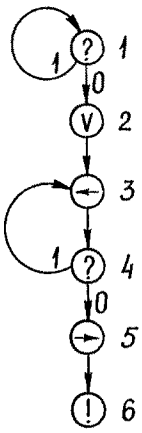


Fig.6.1

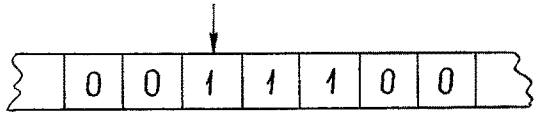


Fig.6.2

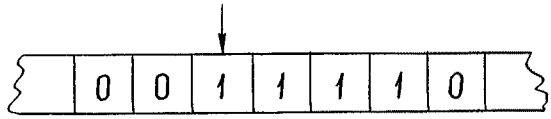


Fig.6.3

Let us consider one-tape Post machine with outer alphabet {0,1}. It is given by the instructions of the type:

← - shift the head one cell leftwards,

→ - shift the head one cell rightwards,

V - print '1' in the current cell,

0 - print '0' in the current cell,

? - conditional instruction: transfer by 1 if 1, transfer by 0 if 0,

! - instruction 'HALT'.

An example of a program is given in Fig.6.1.

Given the input x=111 (Fig.6.2), the program produces y=1111 (Fig.6.3) executing the following sequence of instructions:

? → ? → ? → ? V ← ? ← ? ← ? ← ? ← ? → !

The sequence is formed from all the instructions which are run by the program working on the given x. Such a sequence will be called *operationally-logic history* of the given program for the given x (the notion is introduced in [Er 71]).

Now let us state the problem. Let P be an arbitrary program of the Post machine and

$$\Omega=\{x_1,x_2,\ldots,x_t,\ldots\}$$

be an infinite sequence of natural numbers. We assume that the program P halts for any x_t from Ω and gives the result $P(x_t)$ (we call such Ω permissible for P). Let h_t - operationally-logic history of program P for x_t. Let there be given

$$\{(x_1,h_1),\ldots,(x_t,h_t)\}.$$

It is required to determine a program P' such that P' coincides with

P on Ω, i.e., $P'(x)=P(x)$ for $x\in\Omega$. We denote a collection of all such P' by $\{P_\Omega\}$. An arbitrary total recursive function Π mapping $\langle x_1,h_1,\ldots,x_t,h_t\rangle$ to programs for Post machines is called *strategy*. The program $P_t=\Pi(x_1,h_1,\ldots,x_t,h_t)$ is called the *hypothesis* produced in the moment t. Let:

a) a program P coincide with P for any t at least for $x_1,x_2,\ldots,x_t$,

b) there exist τ such that $P_\tau=P_{\tau+1}=\ldots=P'$ and $P'\in\{P_\Omega\}$.

Then we say that the strategy Π *synthesizes from operationally-logic histories* the program P on the sequence Ω in the limit. We denote by $\Pi^{\#}(\Omega,P)$ the number of changing the hypothesis, i.e., the number of different t, such that $P_t\neq P_{t+1}$. Otherwise, $\Pi^{\#}(\Omega,P)=\infty$. Our aim is to evaluate $\Pi^{\#}(\Omega,P)$. Let us denote by $\|P\|$ the number of conditional instructions in P.

THEOREM 6.1. ([Ba 74-3]) There exists a strategy Π such that for any program P and any sequence Ω

$$\Pi^{\#}(\Omega,P)\leq\|P\|\log_2\|P\|+o(\|P\|\log_2\|P\|).$$

Using advanced enough algorithmic languages $\|P\|$ usually is not too large. For instance, for the program of multiplication of matrices $\|P\|=3$. Therefore Theorem 6.1 shows that there exists a strategy which makes quite a few mistakes in the process of synthesis (almost comparable with the number of mistakes the programmers usually do when writing similar programs).

To prove Theorem 6.1 we associate with any program P the following automaton P_{aut} with input alphabet {0,1}. Let program P begin with a conditional instruction (this does not restrict the generality), and let us represent it as a graph. Let us keep in the graph only those vertexes corresponding to instructions "?" and "!", the paths consisting of other vertexes we replace by arrows. More precisely, if the path is of the type given in Fig.6.4a, we replace it by the arrow with *entry* label ε and *exit* label $(\gamma_1,\gamma_2,\ldots,\gamma_s,\delta)$ (Fig.6.4b). As the result we obtain a diagram of a certain automaton, which we denote by P_{aut}. For the program given in Fig.6.1 the corresponding automaton is shown in Fig.6.5, the input alphabet is $\{(\rightarrow,?),(V,\leftarrow,?),(\leftarrow,?),(\rightarrow,!)\}$.

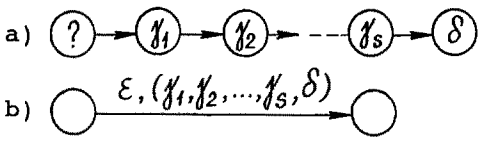


Fig.6.4

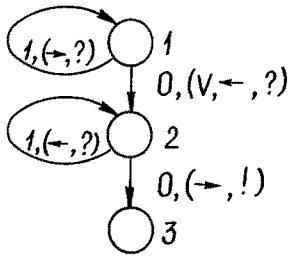


Fig.6.5

Evidently it is possible to restore the program P by P_{aut} q(we denote it by $(P_{aut})_{progr}$).

We associate the input word $\varphi_t=\varepsilon_{t_1}\varepsilon_{t_2}\ldots\varepsilon_{t l_t}$ $(\varepsilon_{tj}\in\{0,1\})$ with x_t and $h_t=?u_1?u_2?\ldots?u_{l_t}$, where u_j is the sequence of instructions (unconditional) between conditional constructions, in the following way: $\varepsilon_{t_1}\varepsilon_{t_2}\ldots\varepsilon_{t l_t}$ are the sequences of values which take the conditional instructions working on x_t in correspondence to history h_t (we assume that conditional instruction "?" takes 1, if the cell in question contains 1, and 0 otherwise). To put it differently, the word φ_t in the diagram of automaton P_{aut} determines the same path as the word x_t with history h_t in the program P. Now, substituting φ_t for x_t in $\Omega=\{x_1,\ldots,x_t,\ldots\}$ we obtain the sequence of words

$$\Omega'=\{\varphi_1,\ldots,\varphi_t,\ldots\}.$$

The following assertion stating the relationship between synthesis of automata and programs is evident now:

A. If A is an arbitrary automaton undistinguishable from P_{aut} on the sequence of input words Ω' (input of all words starts on state 1), then the program $(A)_{progr}$ obtained from automaton A is undistinguishable from program P on the sequence Ω (i.e., they have the same histories and give the same results).

Let us apply the strategy Σ from Theorem 5.3'. We obtain

$$\Sigma^{\#}(\Omega',P_{aut})<|P_{aut}|\log_2|P_{aut}| = (\|P\|+1)\log_2(\|P\|+1) \qquad (6.1)$$

The strategy Σ uses 2t-tuple

$$K_t=<\varphi_1,\ldots,\varphi_t,P_{aut}(\varphi_1),\ldots,P_{aut}(\varphi_t)>$$

to produce (t+1)-hypothesis. On the other hand, the intended strategy Π can use only 2t-tuple $N_t=<x_1,h_1,\ldots,x_t,h_t>$. Nevertheless, evidently it is possible to construct K_t effectively from N_t. Therefore the strategy Π works as follows. First, it finds the 2t-tuple K_t from N_t, then it applies the strategy Σ to K_t and finds

hypothesis $A_t=\Sigma(K_t)$ and finally it transforms the automaton A_t to program $(A_t)_{progr}$ and gives it as a result for N_t. From assertion A and (6.1) it follows that

$$\Pi^{\#}(\Omega,P)\leq\|P\|\log_2\|P\|+o(\|P\|\log_2\|P\|).$$

□

NOTE. Actually we have proved a bit stronger assertion: the obtained strategy synthesizes a program which produces not only the same results as P on Ω, but also the same operationally-logic histories.

Let us consider the so-called *operational histories* [Er 71] instead of operationally-logic histories. Usually they are the minimal necessary information given to the learner in the process of learning some algorithm. They can be obtained from operationally-logic histories by omitting all conditional instructions. For instance, operational history corresponding to the example given above equals $\rightarrow\rightarrow\rightarrow V\leftarrow\leftarrow\leftarrow\leftarrow\rightarrow !$. Let us denote the number of changing the hypothesis in this case by $\Pi^{*}(\Omega,P)$. Let $|P|$ be the number of instructions in P. Then the following theorem holds.

THEOREM 6.2. ([Ba 74-3]) There exists a strategy Π such that for any program P and any sequence Ω

$$\Pi^{*}(\Omega,P)\leq|P|\log_2|P|+o(|P|\log_2|P|).$$

Theorem 6.2 follows easily from Theorem 6.1. Note, that any program P can be transformed to an equivalent program P′ putting the conditional instruction "?" $0\longrightarrow 0\underrightarrow{\longrightarrow}0$ between any two instructions $0\longrightarrow 0$. Obviously, $\|P'\|\leq|P|$ and operational histories of P and P′ coincide. On the other hand, it is possible to restore operationally-logic history $?K_1?K_2?\ldots?K_s!$ by operational history $K_1K_2\ldots K_s!$. Consequently, it is possible to use Theorem 6.1 for program P′. Therefore $\Pi^{*}(\Omega,P')\leq\|P'\|\log_2\|P'\|\leq|P|\log_2|P|$.

□

The question whether a complete analogy with Theorem 6.1 holds in the case of operational histories is open. It is also interesting to study the synthesis of programs with small $\|P\|$: the given evaluations cannot be used reasonably for the case.

7. Errors versus complexity

Following the proof of Theorem 2.1 for an arbitrary enumerated class (U,τ) a total recursive prediction strategy F can be constructed such that for all n:

$$F^{NV}_{U,\tau}(n) \le \log_2 n + \log_2 \log_2 n + o(\log\log\log n).$$

In this chapter a general result will be proved from which it follows that for such "error-optimal" strategies the time complexity of computation of the prediction $F(\langle f(0),\ldots,f(m)\rangle)$ (i.e. a candidate for f(m+1)) may go up, in some sense, to $2^{2^{cm}}$.

To put it precisely, we investigate general algorithms of strategy construction instead of particular strategies. Such algorithms are called *uniform prediction strategies*. The precise definition is as follows.

Any numbering τ of total functions (not necessarily computable) can be treated as an oracle which answers to queries like "$\tau_i(j)=?$". Uniform prediction strategy F is a Turing machine with oracle τ which computes a candidate for f(m+1) from the given values f(0),...,f(m) (it is assumed that the function f is in the numbering τ). We denote this candidate value, as usual, by $F_\tau(\langle f(0),\ldots,f(m)\rangle)$. If the function f is not in the numbering τ, then the computation, maybe, does not halt. Thus, given any τ, F_τ is a partial recursive prediction strategy in the sense of Section 1.

The number of errors committed by the strategy F during the prediction of a function f from a numbering τ we denote, as usual, by

$$F^{NV}_\tau(f) = \text{card}\ \{m \mid F_\tau(\langle f(0),\ldots,f(m)\rangle) \neq f(m+1)\}.$$

Let h(x) be any function of a real variable x defined for all $x \ge 0$. We say that a uniform strategy F *uses* h(m) *queries*, if for any numbering τ, any function f from τ, and all $m \ge 0$ the computation process of $F_\tau(\langle f(0),\ldots,f(m)\rangle)$ issues $\le h(m)$ queries "$\tau_i(j)=?$" to oracle τ. The number of queries can be viewed as a rough lower bound for time complexity of the prediction.

Our main interest is to investigate the power of uniform prediction strategies which use h(m) queries for $h(m)=2^m$, 2^{cm}, x^x, 2^{2^m}, $2^{2^{cm}}$. However, the obtained upper and lower bounds hold for any "reasonable" function h such that $\exp \le h \le 2^{\exp}$ (i.e. h(x) grows at

least as fast as 2^{cx}, but not faster than $2^{2^{cx}}$). To put it precisely, we introduce the following conditions for h:

C1) h is a computable function of real variable, h(x) is defined, positive and twice differentiable for all sufficiently large x. For any integers $m,n \geq 0$ it can be decided effectively whether $h(m)=n$ or not.

C2) There is a real constant $a>0$ such that for all sufficiently large x:

$$(\log_2 h(x))' > a, \quad (\log_2 h(x))'' \geq 0.$$

These conditions are satisfied by any "reasonable" function growing at least as fast as 2^{cx}.

C3) There are two real constants $b,d \geq 0$ such that for all sufficiently large x:

$$(\log_2 h(x))' \leq 2^{bx+d}.$$

This condition is satisfied by any "reasonable" function growing not faster than $2^{2^{cx}}$.

One can verify easily that if the function h satisfies C1,C2,C3, then:

C4) so does the function $\frac{h(x)}{x+2}$,

C5) h(x) is strongly increasing and continuous for all sufficiently large x. This assures the existence of the inverse function $h^{-1}(x)$.

C6) For all sufficiently large integers m:

$$\frac{h(m+1)}{h(m)} \geq 1+a, \quad \sum_{i=0}^{m} h(i) < \frac{1}{a} h(m+1).$$

THEOREM 7.1.([Po 77-1]) Let function h satisfy the conditions C1,C2, and let F be a uniform prediction strategy using h(m) queries. Then there is a computable numbering τ such that for infinitely many n:

$$F_{\tau}^{NV}(\tau_n) > \log_2 n + h^{-1}(n) - O(1).$$

All functions of τ are of the type $N \longrightarrow \{0,1\}$ with a finite number of 1's.

PROOF. For the given strategy F we define a numbering τ and some function f.

First, since C6 holds for h, let m_0 be an integer such that for all $m \geq m_0$: $h(m+1) > h(m)+1$.} Then, for $i \leq h(m_0)$ let all functions τ_i equal to zero. For all $i > h(m_0)$ and $j \leq m_0$ set $f(j)=0$ and $\tau_i(j)=0$. When

during the computation of some $F_\tau(<f(0),\ldots,f(s)>)$, $s\le m_0$, F issues a query "$\tau_i(j)=?$", set $\tau_i(j)=0$.

Suppose now that for some $m\ge m_0$ we have defined:

a) the functions τ_n for all $n\le h(m)$,

b) the values $f(0),\ldots,f(m)$, such that f coincides up to m with all τ_n for a sufficiently large n, and F_τ makes $m-m_0$ false predictions on f up to m,

c) the values $\tau_i(0),\ldots,\tau_i(m)$ for all $i>h(m)$.

Maybe, we have also defined a finite number of some other values $\tau_i(j)$.

Now we define all functions τ_n for $h(m)<n\le h(m+1)$, the value $f(m+1)$ and the values $\tau_i(m+1)$ for $i>h(m+1)$. Let us simulate the computation process of $F_\tau(<f(0),\ldots,f(m)>)$. When F issues a query "$\tau_i(j)=?$" and the value $\tau_i(j)$ is not defined yet, set $\tau_i(j)=0$. The process will end up and yield the prediction $F_\tau(<f(0),\ldots,f(m)>)$. (Suppose, this is not the case. Then we can set all the values $\tau_i(j)$ and $f(j)$ (not defined yet) equal to zero. Since f is now in τ, the prediction $F_\tau(<f(0),\ldots,f(m)>)$ must be defined.)

Then we define $f(m+1)=s$ such that $s\in\{0,1\}$ and $s\ne F_\tau(<f(0),\ldots,f(m)>)$. Thus, this prediction of F_τ is false, and the total of errors is now $m+1-m_0$. Next we define $\tau_i(m+1)$ for all i (if this value is not defined yet):

$$\tau_i(m+1)=\begin{cases} s, \text{ if } \tau_i \text{ coincides with f up to m,} \\ 0, \text{ otherwise.} \end{cases} \qquad (*)$$

Since only a finite number of $\tau_i(m+1)$ has been defined before, the function f will coincide up to m+1 with all functions τ_i for a sufficiently large i.

It remains to define other values of τ_n, $h(m)<n\le h(m+1)$, which have not been defined. Set $\tau_n(j)=0$ for all j, $m+1<j\le k$, where k is such that no value $\tau_i(j)$ has been defined up to now for $i>h(m)$ and $j>k$. The functions τ_n, $h(m)<n\le h(m+1)$, fall into natural equivalence classes:

$$n_1 \simeq n_2 \longleftrightarrow (\forall j\le m+1)\tau_{n_1}(j)=\tau_{n_2}(j)$$

(the values $\tau_n(j)$, $m+1<j\le k$, are equal to zero, i.e. they do not influence the equivalence). Let A be any of these classes, set $t=[\log_2 \mathrm{card}(A)]$. If $t>0$, we define for $n\in A$ the values $\tau_n(k+1),\ldots,\tau_n(k+t)$ using all 2^t binary words of length t. For $j>k+t$ and $n\in A$ set $\tau_n(j)=0$. Thus, predicting the values $\tau_n(k+1),\ldots,\tau_n(k+t)$

any strategy will fail t times on some τ_n, $n\in A$.

Iteration of such steps gives full definition of the numbering τ. Let us show that τ is the required numbering of the theorem.

One can easy verify the following

LEMMA 7.1. If $\tau_i(j)=f(j)$ for all $j<j_0$, and $\tau_i(j_0)\neq f(j_0)$, then $\tau_i(j)=0$ for all j, $j_0<j<k$.

The *rank* $r(A)$ of an equivalence class A (see above) is defined as the maximum number $r\leq m+1$ such that

$$(\forall \tau_n\in A)(\forall j\leq r)\tau_n(j)=f(j).$$

Clearly, $r(A)\geq m_0$, and by the Lemma 7.1, different classes A have different ranks. So we can denote all these classes by A_{m_0}, ..., A_r, ..., A_{m+1}. Predicting the values $\tau_n(0)$, $\tau_n(1)$, ..., $\tau_n(r)$ ($n\in A_r$) the strategy F_τ will fail at least $r-m_0$ times. After that, predicting the values $\tau_n(k+1)$, ..., $\tau_n(k+t)$ for some $n\in A_r$, the strategy F_τ will fail another t times, $t=[\log_2 card(A_r)]$. Hence, F_τ fails on some τ_n, $n\in A_r$, at least $r+\log_2 card(A_r)-m_0-1$ times.

Some of the classes A_r are sufficiently large:

LEMMA 7.2. There exist three constants c,d,e (depending on function h) such that for all sufficiently large m there is r, $m+1-c\leq r\leq m+1$, such that

$$card(A_r)>dh(m+1)-e.$$

Having this lemma we can easily prove the assertion of Theorem 7.1. Indeed, take any sufficiently large m and the class A_r of the lemma. The strategy F_τ fails on some τ_n, $n\in A_r$, at least

$$r+\log_2 card(A_r)-m_0-1$$

times. Now recall that $h(m)<n\leq h(m+1)$:

1) $h(m+1)\geq n$, hence $m+1\geq h^{-1}(n)$ and $r\geq m+1-c\geq h^{-1}(n)-c$.

2) $h(m+1)\geq n$, hence

$$\log_2 card(A_r)\geq \log_2(dh(m+1)-e) \geq \log_2(dn-e)\geq \log_2 n-e'$$

(e' - a constant depending on d,e). Hence,

$$F_\tau^{NV}(\tau_n)>\log_2 n+h^{-1}(n)-c-e'-m_0-1.$$

□

PROOF OF LEMMA 7.2. First, let us note that the classes A_r ($m_0\leq r\leq m+1$) cover all the numbers n, $h(m)<n\leq h(m+1)$. Hence, using C6,

$$\sum_{r=m_0}^{m+1} card(A_r) > h(m+1)-h(m)-1 > (1-\frac{1}{1+a})h(m+1)-1.$$

Let us prove now that, if c is fixed but sufficiently large, then

$$\sum_{r=m_0}^{m-c} \mathrm{card}(A_r) \le \frac{1}{a}(1+a)^{-c}h(m+1),$$

i.e. most of classes A_r are relatively small. Indeed, if $r<m+1$ and $n\in A_r$, then during the computation of some $F_\tau(<f(0),\ldots,f(j)>)$, $1\le j\le r$, the query "$\tau_n(r+1)=?$" must have been issued (otherwise, according to (*), $\tau_n(r+1)$ would have been defined equal to $f(r+1)$, and the rank of A_r were not r). Hence, the total of queries issued is at least

$$\sum_{r=m_0}^{m-c} \mathrm{card}(A_r).$$

On the other hand, for the prediction $F_\tau(<f(0),\ldots,f(j)>)$ at most $h(j)$ queries could have been used, hence, using C6,

$$\sum_{r=m_0}^{m-c} \mathrm{card}(A_r) \le \sum_{j=0}^{m-c} h(j) \le \frac{1}{a}h(m+1-c)\le\frac{1}{a}(1+a)^{-c}h(m+1).$$

Now we have:

$$\sum_{m+1-c}^{m+1} \mathrm{card}(A_r) > (1-\frac{1}{1+a})h(m+1)-1-\frac{1}{a}(1+a)^{-c}h(m+1)=$$

$$=(1-\frac{1}{1+a} - \frac{1}{a}(1+a)^{-c})h(m+1)-1,$$

and for some r, $m+1-c\le r\le m+1$:

$$\mathrm{card}(A_r) > \frac{1}{c+1}(1-\frac{1}{1+a} - \frac{1}{a}(1+a)^{-c})h(m+1)-\frac{1}{c+1}.$$

It remains to make c large enough to satisfy

$$1-\frac{1}{1+a} - \frac{1}{a}(1+a)^{-c}>0.$$

□

EXAMPLES. E1) For any uniform strategy F using 2^m queries:

$$\exists\tau\exists_n^\infty\ F_\tau^{NV}(\tau_n)>2\log_2 n-O(1).$$

E2) For any uniform strategy F using 2^{cm} queries:

$$\exists\tau\exists_n^\infty\ F_\tau^{NV}(\tau_n)>(1+\frac{1}{c})\log_2 n-O(1).$$

E3) For any uniform strategy F using m^m queries:

$$\exists\tau\exists_n^\infty\ F_\tau^{NV}(\tau_n)>\log_2 n+\frac{\log_2 n}{\log_2 \log_2 n} - O(1).$$

E4) For any uniform strategy F using 2^{2^m} queries:

$$\exists\tau\exists_n^\infty\ F_\tau^{NV}(\tau_n)>\log_2 n+\log_2\log_2 n-O(1).$$

E5) For any uniform strategy F using $2^{2^{cm}}$ queries:

$$\exists\tau\exists_n^\infty\ F_\tau^{NV}(\tau_n)>\log_2 n+\frac{1}{c}\log_2\log_2 n-O(1).$$

COROLLARIES. a) If h(x) is growing slower than any exponent 2^{cx}, then no uniform strategy F using h(m) queries can provide an upper bound $F_\tau^{NV}(\tau_n)\le const\cdot\log n$.

b) If h(x) is growing as an exponent 2^{cx}, then no uniform strategy F using h(m) queries can provide an upper bound $F_\tau^{NV}(\tau_n)\le\log_2 n+o(\log n)$.

c) If h(x) is growing slower than any super-exponent $2^{2^{cx}}$, then no uniform strategy F using h(m) queries can provide an upper bound $F_\tau^{NV}(\tau_n)\le\log_2 n+const\cdot loglog\ n$.

d) The uniform strategy F defined in the proof of The- orem 2.1 uses (for some numbering τ and for infinitely many n) at least $2^{2^{cm}}$ queries to compute $F_\tau(<\tau_n(0),\dots,\tau_n(m)>)$.

Now let us turn to upper bounds. Let h(x) be a function satisfying the condition C1 and $\bar\pi=\{\pi_n\}$ be a recursive series of real numbers. By $\{h\bar\pi\}$ we denote the following modification of the uniform prediction strategy from the proof of Theorem 2.1.

The prediction $\{h\bar\pi\}_\tau(<f(0),\dots,f(m)>)$ is computed as follows. We consider the functions τ_i only for $i\le h(m)$ and the weights π_i assigned to them. Find all numbers t such that

$$E_t=\{i \mid i\le h(m) \ \&\ (\forall j\le m)(\tau_i(j)=f(j) \ \&\ \tau_i(m+1)=t)\}\ne 0.$$

If there are no such t's, set the prediction equal to zero. For each t found compute its weight

$$w_t=\sum\{\pi_i \mid i\in E_t\}$$

with the precision 2^{-2m}, i.e. find rational number r_t such that $|r_t-w_t|\le 2^{-2m}$. Now find t with maximum r_t, and set

$$\{h\bar\pi\}_\tau(<f(0),\dots,f(m)>)=t.$$

$\{h\bar\pi\}$ is a total recursive prediction strategy using $(m+2)h(m)$ queries. There are two different types of errors committed by the strategy $\{h\bar\pi\}_\tau$ during the prediction of values of the function τ_n:

- type 1:

$$\{h\bar\pi\}_\tau(<\tau_n(0),\dots,\tau_n(m)>)\ne\tau_n(m+1) \ \&\ h(m)<n$$

(i.e. when computing the prediction, the function τ_n is ignored),

- type 2:

$$\{h\bar\pi\}_\tau(<\tau_n(0),\dots,\tau_n(m)>)\ne\tau_n(m+1) \ \&\ h(m)\ge n.$$

Slightly modifying the proof of Theorem 2.1 we obtain the following

LEMMA 7.3. Let the function h satisfy conditions C1,C2 and the predictions $\{h\bar\pi\}_\tau(<\tau_n(0),\dots,\tau_n(m)>)$ be false for $m=m_1,m_2,\dots,m_s$. Let us denote: s_1 - the number of type 1 errors, s_2 - the number of

type 2 errors ($s=s_1+s_2$). Then:

a) $s_1<h^{-1}(n)$,

b) $2^{s_2}\pi_n < \sum_{i=1}^{s_2-1} 2^i\sigma_{s_1+i}+2+\sum_{j=0}^{\infty}\pi_j$,

where $\sigma_i=\sum\{\pi_j \mid h(m_i)<j\le h(m_{i+1})\}$.

Now we define a special sequence $\bar{\pi}^0$:

$$\pi^0_n = \frac{1}{h'h^{-1}(n)2^{h^{-1}(n)}}.$$

LEMMA 7.4. Let the function h satisfy conditions C1,C2 and C3. Let the strategy $\{h\bar{\pi}^0\}_\tau$ predict the function τ_n. Let us denote: s_1 - the number of type 1 errors, s_2 - the number of type 2 errors. Then:

a) $s_1<h^{-1}(n)$,

b) $s_1+s_2-\log_2 s_2 < h^{-1}(n)+\log_2 h'h^{-1}(n)+O(1)$.

PROOF. One can verify easily that π^0_x is a decreasing function, hence for all n:

$$\pi^0_n < \int_{n-1}^{n} \frac{dx}{h'h^{-1}(x)2^{h^{-1}(x)}}.$$

Summing up we have

$$\sigma_i < \int_{h(m_i)}^{h(m_{i+1})+1} \frac{dx}{h'h^{-1}(x)2^{h^{-1}(x)}}.$$

Substitute h(t) for x:

$$\int_{h(m_i)}^{\infty} \frac{dx}{h'h^{-1}(x)2^{h^{-1}(x)}} = \int_{m_i}^{\infty} \frac{h'(t)dt}{h'(t)2^t} = \int_{m_i}^{\infty} \frac{dt}{2^t} = \frac{1}{\ln 2}2^{-m_i}.$$

Thus we have:

$$\sigma_i<\frac{1}{\ln 2}2^{-m_i} \le \frac{2}{\ln 2}2^{-i}$$

(since $m_i\ge i-1$). Hence, by Lemma 7.3:

$$2^{s_2}\pi^0_n < \sum_{i=1}^{s_2-1} 2^i\sigma_{s_1+i}+2+\sum_{j=0}^{\infty}\pi^0_j < \text{const}\cdot s_2 2^{-s_1},$$

$$s_2+\log_2\pi^0_n < \log_2 s_2 - s_1+\text{const},$$

$$s_2-\log_2 h'h^{-1}(n)-h^{-1}(n) < \log_2 s_2 - s_1+\text{const}.$$

□

Now we can prove the upper bound:

THEOREM 7.2. ([Po 77-1]) Let the function h satisfy conditions C1,C2,C3. There is a total recursive uniform strategy F using h(m) queries such that for any numbering τ and all n:

$$F^{NV}_{\tau}(\tau_n) \le \log_2 n + (b+1)h^{-1}(n) + O(\text{loglog } n)$$

(the constant b is from condition C3).

PROOF. Take $h_1(x) = \frac{h(x)}{x+2}$ and the strategy $\{h_1\bar{\pi}^0\}$. Since h_1 also satisfies C1,C2,C3 (with the same constants b,d), we have:

$$h_1'(x) \le h_1(x)2^{bx+d},$$

$$h_1' h_1^{-1}(n) \le h_1 h_1^{-1}(n) 2^{bh_1^{-1}(n)+d},$$

$$\log_2 h_1' h_1^{-1}(n) \le \log_2 n + bh_1^{-1}(n) + d.$$

Hence, by Lemma 7.4:

$$s_1 + s_2 - \log_2 s_2 \le \log_2 n + (b+1)h_1^{-1}(n) + O(1).$$

Since $x - \log_2 x \le y$ implies $x \le y + \log_2 y + O(1)$, and by C6, $h_1^{-1}(n) = O(\log n)$:

$$s_1 + s_2 \le \log_2 n + (b+1)h_1^{-1}(n) + O(\text{loglog } n).$$

Since $F^{NV}_{\tau}(\tau_n) = s_1 + s_2$ and $h_1(n) = h^{-1}(n) + O(\text{loglog } n)$, the proof is completed.

□

EXAMPLES. EE1) Let $h(x) = 2^x$, then b=0 in C3. There is a uniform strategy F using 2^m queries such that

$$F^{NV}_{\tau}(\tau_n) \le 2\log_2 n + O(\text{loglog } n).$$

Compare example E1.

EE2) Let $h(x) = 2^{cx}$. There is a uniform strategy F using 2^{cm} queries such that

$$F^{NV}_{\tau}(\tau_n) \le (1+\frac{1}{c})\log_2 n + O(\text{loglog } n).$$

Compare example E2.

EE3) Let $h(x) = x^x$. There is a uniform strategy F using m^m queries such that

$$F^{NV}_{\tau}(\tau_n) \le \log_2 n + O(\frac{\log n}{\text{loglog } n}).$$

Compare example E3.

EE4) Let $h(x) = 2^{2^x}$. There is a uniform strategy F using 2^{2^m} queries such that

$$F^{NV}_{\tau}(\tau_n) \le \log_2 n + O(\text{loglog } n).$$

Compare example E4.

8. Probabilictic strategies

In Sections 3,4 the complexity of *deterministic* identification of τ-indices was investigated, and the corresponding exact estimates were obtained. In this section we obtain the exact estimate ln n for the number of mindchanges for the *probabilistic* identification of τ-indices.

The hypotheses $F(\langle f(0),\ldots,f(m)\rangle)$ of a *probabilistic strategy* F are random natural numbers which take their values over some fixed probability space P. Formally, probabilistic strategy F is a mapping which associates with each elementary event $e\in P$ some deterministic strategy F_e. Thus the hypothesis $F(\langle f(0),\ldots,f(m)\rangle)$ takes its values n with fixed probabilities

$$p_F(\langle f(0),\ldots,f(m)\rangle,n)=P\{F(\langle f(0),\ldots,f(m)\rangle)=n\}.$$

Recursive probabilistic strategies can be defined by means of probabilistic Turing machines introduced first in [LMS 56]. Let a random Bernoulli generator of some distribution (p,1-p) be fixed, $0<p<1$. The generator is switched into deterministic "apparatus" of a Turing machine. As a result, the operation of the machine becomes probabilistic, and we can speak of the probability that the operation satisfies certain conditions.

Consider the following Turing machine M operating with a fixed Bernoulli generator. With input sequence

$$f(0),f(1),\ldots,f(m),\ldots$$

this machine prints as output an empty, finite or infinite sequence of natural numbers (hypotheses):

$$h_0,h_1,\ldots,h_m,\ldots,$$

where h_m depends only on the values $f(0),\ldots,f(m)$. To each infinite realization of Bernoulli generator's output (i.e. an infinite sequence of 0's and 1's) corresponds a completely determined operation of the machine M as a deterministic strategy in the sense of Section 1.

By $P\{M,\tau,f\}$ we denote the probability that a probabilistic strategy M identifies in the limit a τ-index of the function f.

By $P\{M,f,\leq k\}$ we denote the probability that probabilistic strategy M makes no more than k mindchanges by the function f.

THEOREM 8.1. ([Po 75]) For any enumerated class (U,τ) there exists a probabilistic strategy M such that $P\{M,\tau,f\}=1$ for all $f\in U$, and as $n\longrightarrow\infty$

$$P\{M,\tau_n,\leq \ln n+O(\sqrt{\log n}\cdot \operatorname{loglog} n)\}\longrightarrow 1.$$

For a computable numbering τ, a recursive probabilistic strategy M can be constructed..

THEOREM 8.2. ([Po 75]) For any countable set Φ of probabilistic strategies there exists an enumerated class (U,τ) such that for any strategy $M\in\Phi$, if $P\{M,\tau,f\}=1$ for all $f\in U$, there is an increasing sequence $\{n_k\}$ such that as $k\longrightarrow\infty$

$$P\{M,\tau_{n_k},\leq \ln n_k-O(\sqrt{\log n_k}\cdot \operatorname{loglog} n_k)\}\longrightarrow 0.$$

For the class of all recursive probabilistic strategies a computable numbering τ can be constructed.

Let M,τ,f be given. We consider some sufficient condition for $P\{M,\tau,f\}=1$. Let us denote by $f^{[m]}$ the code $\langle f(0),\ldots,f(m)\rangle$, then the random variable $M(\langle f(0),\ldots,f(m)\rangle)$ can be denoted by $M(f^{[m]})$. By $P_m(M,f)$ we denote the probability that M changes its hypothesis at step m, i.e. $P\{M(f^{[m]})\neq M(f^{[m+1]})\}$.

We say that strategy M is *τ-consistent* on the function f if, for all m,

a) $M(f^{[m]})$ is defined with probability 1,

b) if $P\{M(f^{[m]})=n\}>0$, then $\tau_n(j)=f(j)$ for all $j\leq m$.

By Borel-Cantelli lemma, M is τ-consistent on the function f, then $\sum_m P_m(M,f)<\infty$ implies $P\{M,\tau,f\}=1$. Thus in the case of consistent strategies the fact of τ-identification can be established in terms of summing up the probabilities of mindchanges.

The upper bound ln n is proved by means of probabilistic counterpart of the strategy from the proof of Theorem 2.1. Essential difficulties arise, however, not in the construction of the strategy, but in its analysis.

Let (U,τ) be an enumerated class of total functions. Take some probability distribution $\{\pi_n\}$, where $\pi_n>0$ for all n and $\sum_n \pi_n=1$. Let $M_{\tau\pi}$ be the following τ-consistent probabilistic strategy.

If the set $E_0=\{n\mid \tau_n(0)=f(0)\}$ is empty, then we set $M_{\tau\pi}(f^{[0]})$ undefined with probability 1. If E_0 is nonempty, we put $M_{\tau\pi}(f^{[0]})=n$ with probability π_n/σ for every $n\in E_0$, where $\sigma=\Sigma\{\pi_n\mid n\in E_0\}$.

Let us assume now that the hypotheses $M_{\tau\pi}(f^{[j]})$ have already been determined for $j<m$, and $M_{\tau\pi}(f^{[m-1]})=p$. If p is "undefined",

then we set $M_{\tau\pi}(f^{[m]})$ undefined with probability 1. Else, if $\tau_p(m)=f(m)$ (i.e. the hypothesis p is correct also for the next argument m), we set $M_{\tau\pi}(f^{[m]})=p$ with probability 1. Now suppose $\tau_p(m)\neq f(m)$.

Let us take the set of all (for the time being) appropriate hypotheses, i.e.

$$E_m=\{n \mid (\forall j\leq m)\tau_n(j)=f(j)\}.$$

If E_m is empty, we put $M_{\tau\pi}(f^{[m]})$ undefined with probability 1. If E_m is nonempty, we put $M_{\tau\pi}(f^{[m]})=n$ with probability π_n/σ for every $n\in E_m$, where $\sigma=\Sigma\{\pi_n \mid n\in E_m\}$.

LEMMA 8.1. For all n,

$$\sum_m P_m(M,\tau_n)\leq \ln\frac{1}{\pi_n}.$$

From this it follows that for an arbitrary choice of distribution π, if $\pi_n>0$ for all n, the strategy $M_{\tau\pi}$ identifies in the limit τ-index of an arbitrary function in the class U with probability 1.

LEMMA 8.2. Let the function $f\in U$ be fixed. Then the following events are independent:

$$A_m=\{M_{\tau\pi}(f^{[m]})\neq M_{\tau\pi}(f^{[m+1]})\}, \quad m=0,1,2,\ldots.$$

It is curious that the events A_m (i.e. "at the m-th step strategy $M_{\tau\pi}$ changes its mind") do not display any striking indications of independence; nevertheless, they do satisfy the formal independence criterion.

If we take

$$\pi'_n=\frac{c}{n(\ln n)^2},$$

with the convention that $1/0=1$ and $\ln 0=1$, then by Lemma 8.1 the sum of the probabilities of hypothesis correcting of strategy $M_{\tau\pi'}$ with the function τ_n will not be greater than $\ln n+O(\text{loglog } n)$. Lemma 8.2 and Chebyshev inequality allow to deduce from this that, as $n\to\infty$,

$$P\{M_{\tau\pi'},\tau_n,\leq \ln n+O(\sqrt{\log n}\cdot\text{loglog } n)\}\to 1.$$

It is easy to see that if the numbering τ is computable, the strategy $M_{\tau\pi'}$ can be made recursive.

The lower bound $\ln n$ is based upon Lemma 8.3, below. Let $\{X_j\}$ be a sequence of independent random variables such that

$$P\{X_j=1\}=\frac{1}{j},\quad P\{X_j=0\}=1-\frac{1}{j}.$$

It can be shown that, as $n\to\infty$,

$$P\{\sum_1^n X_j\geq \ln n-O(\sqrt{\log n}\cdot\text{loglog } n)\}\to 1.$$

LEMMA 8.3. Let M be a probabilistic strategy, k and n natural numbers with k<n, and $\varepsilon>0$ a rational number. Then there is a set of n functions $\sigma_1,\ldots,\sigma_n$ such that if M identifies with probability 1 the σ-number of an arbitrary function of the set, then with one of these functions M changes its mind $\geq k$ times with probability

$$\geq(1-\varepsilon)P\{\sum_{j=1}^{n} X_j \geq k\}.$$

If M is recursive strategy, the set $\sigma_1,\ldots,\sigma_n$ can be constructed effectively.

Let $\{M_i\}$ be an enumeration of all probabilistic strategies from countable class Φ. With every pair (i,s) we associate the set of functions of Lemma 8.3 for $M=M_i$, $n=2^s$, $k=s \ln 2 - \sqrt{s} \log s$, $\varepsilon=2^{-s}$. Following the method of Section 4, a numbering τ can be constructed from these sets, thus proving Theorem 8.2.

For detailed proofs of lemmas see [Po 77-2].

References

[Ba 74-1] J.Barzdin. Limiting synthesis of τ-indices. Theory of Algorithms and Programs, vol.1, Latvia State University, 1974, pp.112-116 (in Russian)

[Ba 74-2] J.Barzdin. Prediction and limiting synthesis of finite automata. Theory of Algorithms and Programs, vol.1, Latvia State University, 1974, pp.129-144 (in Russian)

[Ba 74-3] J.Barzdin. A note on program synthesis from computational histories. Theory of Algorithms and Programs, vol.1, Latvia State University, 1974, pp.145-151 (in Russian)

[BF 72] J.Barzdin, R.Freivald. On the prediction of general recursive functions. Soviet Math. Dokl. 13, 1972, pp.1224-1228

[BF 74] J.Barzdin, R.Freivald. Prediction and limiting synthesis of effectively enumerable classes of functions. Theory of Algorithms and Programs, vol.1, Latvia State University, 1974, pp.101-111 (in Russian)

[BKP 74] J.Barzdin, E.Kinber, K.Podnieks. Speeding up prediction and limiting synthesis of functions. Theory of Algorithms and Programs, vol.1, Latvia State University, 1974, pp.117-128 (in Russian)

[Bie 72] A.W.Biermann. On the inference of Turing machines from sample computations. Artificial Intelligence, 1972

[Er 71] A.P.Ershov. Theory of program schemata. IFIP Congress 71, Ljubljana, 1971, 1, pp.144-163

[Go 67] E.M.Gold. Language identification in the limit. Information and Control, 10:5, 1967, pp.447-474

[Kol 65] A.N.Kolmogorov. Three approaches to the definition of the notion "quantity of information". Problemy peredachi informacii, 1:1, 1965 (in Russian)

[Kor 67] A.D.Korshunov. On asymptotic estimates of the number of finite automata. Kibernetika, 2, 1967 (in Russian)

[LMS 56] K. de Leeuw, E.F.Moore et al, Computability by probabilistic machines. Automata Studies (Ann. of Math. Studies, No.34), Princeton Univ. Press, Princeton, N.J., 1956, pp.183-212

[ML 66] P.Martin-Löf. On the notion of random sequence. Teoriya veroyatnosti i ee primeneniya, 2:1, 1966 (in Russian)

[Moo 56] E.F.Moore. Gedanken-experiments on sequential machines. Automata Studies (Ann. of Math. Studies, No.34), Princeton Univ. Press, Princeton, N.J., 1956, pp.129-153

[Po 75] K.M.Podnieks. Probabilistic synthesis of enumerated classes of functions. Soviet Math. Dokl. 16, 1975, pp.1042-1045

[Po 77-1] K.M.Podnieks. Computational complexity of prediction strategies. Theory of Algorithms and Programs, vol.3, Latvia State University, 1977, pp.89-102 (in Russian)

[Po 77-2] K.M.Podnieks. Probabilistic program synthesis. Theory of Algorithms and Programs, vol.3, Latvia State University, 1977, pp.57-88 (in Russian)

[Rog 67] H.Rogers, Jr. Theory of recursive functions and effective computability. McGraw-Hill, New York, 1967

[TB 73] B.A.Trakhtenbrot, J.M.Barzdin. Finite Automata (Behaviour and Synthesis). North-Holland, Amsterdam, 1972

[ZL 70] A.K.Zvonkin, L.A.Levin. Complexity of finite objects and foundations of the information and randomness notions by the theory of algorithms. Uspekhi matematicheskikh nauk, 25:6, 1970 (in Russian)