ON THE INFLUENCE OF TECHNOLOGY ON LEARNING PROCESSES*

ILJA KUCEVALOVS, OJĀRS KRASTS, RŪSIŅŠ FREIVALDS

Institute of Mathematics and Computer Science, University of Latvia, Raiņa bulvāris 29, Riga, LV-1459, Latvia

THOMAS ZEUGMANN

Division of Computer Science, Hokkaido University, N-14, W-9, Sapporo 060-0814, Japan

Received (January 6, 2013)
Revised (May 9, 2014)

ABSTRACT

Probabilistic computations and frequency computations were invented for the same purpose, namely, to study possible advantages of technology involving random choices. Recently several authors have discovered close relationships of these generalizations of deterministic computations to computations taking advice. Various forms of computation taking advice were studied by Karp and Lipton [1], Damm and Holzer [2], and Freivalds [3]. In the present paper, we apply the nonconstructive, probabilistic, and frequency methods to an inductive inference paradigm originally due to Gold [4] and investigate their impact on the resulting learning models. Several trade-offs with respect to the resulting learnability are shown.

Keywords: probabilistic computations; frequency computations; nonconstructive methods; algorithmic learning; inductive inference

1. Introduction

Probabilistic algorithms and automata were invented to simulate random processes in living organisms. A fundamental question related to this research is whether or not this simulation is adequate, i.e., whether or not indeterministic processes in living beings are probabilistic in the sense described by Pascal and Fermat. Quantum theory showed that indeterministic processes in the microworld surely need a more complicated description. This makes the usage of such generalized computations a practical problem in Computer Science.

Frequency algorithms were introduced by Rose [5] with a clear aim to have a completely deterministic mechanism with properties of probabilistic algorithms.

*The research was supported by the project ERAF Nr.2DP/2.1.1.1/13/APIA/VIAA/027 and the Invitation Fellowship for Research in Japan S12052 by Japan Society for the Promotion of Science

Frequency computations became popular because of their relation to the notion of *autoreducibility*.

In his paper [3], Freivalds examined nonconstructive proofs in computation theory and introduced the notion of nonconstructive computation. The notion was based on Freivalds' observation of nonconstructive proofs as of a *construction* of algorithms that do not rely exclusively on the "natural" input data and their own internal mechanisms, but utilize some additional "help from the outside" as well. For instance, consider a version of Bārzdiņš' lemma [6] that sets an upper bound of $\log n$ on Kolmogorov complexity of characteristic words of recursively enumerable sets. The proof of this lemma is nonconstructive (which is natural due to the Kolmogorov complexity notion used), but in [3] it is reinterpreted as a construction of an algorithm that utilizes an amount of $\log n$ of additional information.

In [3] a simple definition of nonconstructive computation was introduced, considering language recognition by finite automata using additional information. It was pointed out that other computational models could be utilized in nonconstructive computation, too. Clearly, Freivalds' [3] definitions should be revised in this case.

As far as inductive inference is concerned, nonconstructive proofs are not uncommon. The "formal theory" of inductive inference due to Solomonoff [7, 8] is based on the notion of algorithmic probability, an uncomputable function closely related to Kolmogorov complexity [6]. The learning model considered in this paper is Gold's [4] identification in the limit model. This model is less abstract and more closely related to computational models than Solomonoff's [7, 8] "formal theory." However, nonconstructive proofs can be found here as well.

For example, Case and Smith [9] explicitly mention that the proof of their Theorem 2.3 is nonconstructive. Indeed, it is based on observation of two possible cases in a certain construction such that there is no way to determine which of the cases actually holds. A similar proof has, for example, Theorem 22 in [10]. Using the terminology of [3], we may say that the amount of nonconstructivity assumed in these proofs is 1 bit.

On the other hand, some versions of classical identification criteria utilizing "additional information" have already been considered (see, for example, [11, 12, 13]). However, the additional information considered is usually of some specific kind and/or the objects of identification are limited to a certain class (say, identification of recursive functions). Second, to our knowledge there were no references to formalizing nonconstructive proofs in the existing literature.

In the present paper we study the problem of what impact do probabilistic computations and frequency computations have on inductive inference paradigms. The resulting learning models are compared to one another. Additionally, a learning model is considered in which the learner receives in addition to the usual graph of a target function a certain amount of *nonconstructive* information.

2. Notations and Definitions

Unspecified notations follow Rogers [14]. In addition to or in contrast with [14] we use the following. Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ be the set of all natural numbers, and let furthermore $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$ be the set of all positive natural numbers. We use $\mathbb{N}^*$ to denote the set of all finite sequences of natural numbers. By $|S|$ and $\wp(S)$ we denote the cardinality and power set of a set S, respectively. Let $\emptyset$, $\in$, $\subset$, $\subseteq$, $\supset$, $\supseteq$, and $\#$ denote the empty set, element of, proper subset, subset, proper superset, superset, and incomparability of sets, respectively.

Let $n \in \mathbb{N}$ be such that $n \geq 2$. The set of all partial recursive functions and of all recursive functions of one respectively n variables over $\mathbb{N}$ is denoted by $\mathcal{P}$, $\mathcal{R}$, $\mathcal{P}^n$, $\mathcal{R}^n$, respectively. Note that the recursive functions are total. For every $f \in \mathcal{P}$ we use $\mathtt{dom}(f)$ to denote the *domain* of the function f, i.e., $\mathtt{dom}(f) = \{x \mid x \in \mathbb{N},\ f(x) \text{ is defined}\}$. By $\mathtt{Val}(f)$ we denote the *range* of f, that is, $\mathtt{Val}(f) = \{f(x) \mid x \in \mathtt{dom}(f)\}$.

Any function $\psi \in \mathcal{P}^2$ is called a *numbering*. Let $\psi \in \mathcal{P}^2$, then we write ψ_i instead of $\lambda x.\psi(i, x)$ and set $\mathcal{P}_\psi = \{\psi_i \mid\ i \in \mathbb{N}\}$ as well as $\mathcal{R}_\psi = \mathcal{P}_\psi \cap \mathcal{R}$. So if $f \in \mathcal{P}_\psi$, then there is a number $i \in \mathbb{N}$ such that $f = \psi_i$. If $f \in \mathcal{P}$ and $i \in \mathbb{N}$ are such that $\psi_i = f$, then i is called a *ψ–program* for the function f. Let ψ be any numbering, and let $i, x \in \mathbb{N}$; if $\psi_i(x)$ is defined (abbr. $\psi_i(x)\downarrow$) then we also say that $\psi_i(x)$ *converges*. Otherwise, $\psi_i(x)$ is said to *diverge* (abbr. $\psi_i(x)\uparrow$). Let $\psi \in \mathcal{P}^2$ and $f \in \mathcal{P}$; then we use $\min_\psi f$ to denote the least number i such that $\psi_i = f$.

A numbering $\varphi \in \mathcal{P}^2$ is called a *Gödel numbering* (cf. Rogers [14]) if $\mathcal{P}_\varphi = \mathcal{P}$, and for every numbering $\psi \in \mathcal{P}^2$, there is a *compiler* $c \in \mathcal{R}$ such that $\psi_i = \varphi_{c(i)}$ for all $i \in \mathbb{N}$. We use *Göd* to denote the set of all Gödel numberings. For further information concerning Gödel numberings and their properties we refer the reader to [15, 14].

By $NUM = \{\mathcal{U} \mid$ there is a $\psi \in \mathcal{R}^2$ such that $\mathcal{U} \subseteq \mathcal{P}_\psi\}$ we denote the family of all subsets of all recursively enumerable classes of recursive functions. Furthermore, we use $NUM! = \{\mathcal{U} \mid$ there is a $\psi \in \mathcal{R}^2$ such that $\mathcal{U} = \mathcal{P}_\psi\}$ to denote the family of all recursively enumerable classes of recursive functions. The elements of $NUM!$ are referred to as *indexed families*.

Let $\langle \ldots \rangle$ be any recursive encoding of $\mathbb{N}^*$ onto $\mathbb{N}$ (cf. Rogers [14]). We write f^n instead of $\langle (f(0), \ldots, f(n)) \rangle$, for all $n \in \mathbb{N}$ and all $f \in \mathcal{R}$.

A sequence $(j_n)_{n \in \mathbb{N}}$ of natural numbers is said to *converge* to the number j if $j_n = j$ for all but finitely many $n \in \mathbb{N}$. Moreover, a sequence $(j_n)_{n \in \mathbb{N}}$ of natural numbers is said to *finitely converge* to the number j if it converges to j and for all $n \in \mathbb{N}$, $j_n = j_{n+1}$ implies $j_k = j$ for all $k \geq n$. Now we are ready to define the learning models considered in this paper.

2.1. *Deterministic Learning of Recursive Functions*

When learning recursive functions growing initial segments f^n, where $n = 0, 1, 2, \ldots$, are fed to the learning algorithm, henceforth called *strategy*. For each initial segment

the strategy has then to compute a hypothesis i_n which is a natural number. These hypotheses are interpreted with respect to a suitably chosen hypothesis space ψ which is a numbering. The interpretation of the hypothesis i_n is that the strategy conjectures program i_n in the numbering ψ to compute the target function f. Furthermore, one requires the sequence $(i_n)_{n\in\mathbb{N}}$ of all computed hypotheses to converge to a program i correctly computing the target function f, i.e., $\psi_i = f$. A strategy learns a class of recursive functions provided it can learn every function from it. The model just explained is basically *learning in the limit* as introduced by Gold [4]. Many variations of this model have been studied (cf., e.g., [9, 16, 17, 18], and the references therein). More formally, we have the following definition.

Definition 2.1 (Gold [19, 4]) Let $\mathcal{U} \subseteq \mathcal{R}$ and let $\psi \in \mathcal{P}^2$. The class $\mathcal{U}$ is said to be *learnable in the limit with respect to* ψ if there is a strategy $S \in \mathcal{P}$ such that for each function $f \in \mathcal{U}$,

(1) for all $n \in \mathbb{N}$, $S(f^n)$ is defined,
(2) there is a $j \in \mathbb{N}$ with $\psi_j = f$ and the sequence $(S(f^n))_{n\in\mathbb{N}}$ converges to j.

If $\mathcal{U}$ is learnable in the limit with respect to ψ by S then we write $\mathcal{U} \in \mathit{LIM}_\psi(S)$. Furthermore, let $\mathit{LIM}_\psi = \{\mathcal{U} \mid \mathcal{U} \text{ is learnable in the limit with respect to } \psi\}$, and let LIM be the collection of all classes learnable in the limit with respect to some hypothesis space, i.e., $\mathit{LIM} = \bigcup_{\psi\in\mathcal{P}^2} \mathit{LIM}_\psi$.

Some remarks are mandatory here. Let us start with the semantics of the hypotheses produced by a strategy S. If S is defined on input f^n, then we always interpret the number $S(f^n)$ as a ψ–number. This convention is adopted to all the definitions below. Furthermore, it is easy to show that $\mathit{LIM}_\varphi = \mathit{LIM}$ for every Gödel numbering φ (cf. [20, 9]). In the above definition LIM stands for "limit." Moreover, in accordance with the definition of convergence, only finitely many data of the graph of a function f were available to the strategy S up to the unknown point of convergence. Therefore, some form of learning must have taken place. Thus, the use of the term "learn" in the above definition is indeed justified. Note that instead of LIM sometimes the notation EX is used in the literature, where EX stands for "explain" (cf., e.g., [9, 17]).

In general it is not decidable whether or not a strategy has already converged when successively fed some graph of a function. With the next definition we consider a special case where it has to be decidable whether or not a strategy has learned its input function. That is, we replace the requirement that the sequence of all created hypotheses "has to *converge*" by "has to *converge finitely*."

Definition 2.2 (Gold [4], Trakhtenbrot and Barzdin [21]) Let $\mathcal{U} \subseteq \mathcal{R}$ and let $\psi \in \mathcal{P}^2$. The class $\mathcal{U}$ is said to be *finitely learnable with respect to* ψ if there is a strategy $S \in \mathcal{P}$ such that for each function $f \in \mathcal{U}$,

(1) for all $n \in \mathbb{N}$, $S(f^n)$ is defined,

(2) there is a $j \in \mathbb{N}$ such that $\psi_j = f$ and the sequence $(S(f^n))_{n\in\mathbb{N}}$ finitely converges to j.

If $\mathcal{U}$ is finitely learnable with respect to ψ by a strategy S, we write $\mathcal{U} \in \mathit{FIN}_\psi(S)$. The learning types FIN_ψ and FIN are defined analogously to the above.

It is easy to prove that $\mathit{FIN}_\varphi = \mathit{FIN}$ for every Gödel numbering φ. Moreover, we have $\mathit{FIN} \subset \mathit{LIM}$ (cf., e.g., [18]). Since we are mainly interested in finite learning in the present paper, we shall consider all variations defined below for finite learning only.

In the following modification of Definition 2.2 we require the strategy to finitely converge to $\min_\psi f$ instead of converging finitely to any program for the target function f. This modification goes back to Freivalds [22] and Kinber [23] who considered it for learning in the limit.

Definition 2.3. Let $\mathcal{U} \subseteq \mathcal{R}$ and let $\psi \in \mathcal{P}^2$. The class $\mathcal{U}$ is said to be *finitely ψ-minimal learnable with respect to ψ* if there is a strategy $S \in \mathcal{P}$ such that for each function $f \in \mathcal{U}$,

(1) for all $n \in \mathbb{N}$, $S(f^n)$ is defined,
(2) the sequence $(S(f^n))_{n\in\mathbb{N}}$ finitely converges to $\min_\psi f$.

If $\mathcal{U}$ is finitely ψ-minimal learnable with respect to ψ by a strategy S then we write $\mathcal{U} \in \mathit{MIN\text{-}FIN}_\psi(S)$. The learning types $\mathit{MIN\text{-}FIN}_\psi$ and $\mathit{MIN\text{-}FIN}$ are defined analogously to the above.

Definition 2.4. Let $\mathcal{U} \subseteq \mathcal{R}$ and let $\psi \in \mathcal{P}^2$. The class $\mathcal{U}$ is said to be *finitely ψ-nearly minimal learnable with respect to ψ* if there is a strategy $S \in \mathcal{P}$ and a constant $c \geq 0$ such that for each function $f \in \mathcal{U}$,

(1) for all $n \in \mathbb{N}$, $S(f^n)$ is defined,
(2) the sequence $(S(f^n))_{n\in\mathbb{N}}$ finitely converges to a program i such that $\varphi_i = f$ and $i \leq \min_\psi f + c$.

If $\mathcal{U}$ is finitely ψ-nearly minimal learnable with respect to ψ by a strategy S then we write $\mathcal{U} \in \mathit{NEARLY\text{-}MIN\text{-}FIN}_\psi(S)$. The learning types $\mathit{NEARLY\text{-}MIN\text{-}FIN}_\psi$ and $\mathit{NEARLY\text{-}MIN\text{-}FIN}$ are defined analogously to the above.

Note that there is also slightly different version of nearly minimal learnability in the literature, where nearly minimal refers to "within a recursive fundge factor h." That is, one requires the learner to finitely converge to a correct program i such that $i \leq h(\min_\psi f$, where f is the target function (cf., e.g., Chen [24], and the references therein).

2.2. *Nonconstructive Learning of Recursive Functions*

The next model we shall consider in this paper derives its motivation from the fact that $\mathcal{R} \notin \mathit{LIM}$, and thus $\mathcal{R} \notin \mathit{FIN}$ and $\mathcal{R} \notin \mathit{MIN\text{-}FIN}$. Freivalds and Zeug-

mann [25] introduced a new measure to classify the difficulty of learning the class $\mathcal{R}$ or to learn indexed families of recursive functions under additional constraints such as finite learning or finite minimal learning. This new measure is the amount of nonconstructivity needed to achieve the specified learning goal.

The strategies used for nonconstructive inductive inference take as input not only the encoded graph of a function $f \in \mathcal{R}$ but also a help-word w. The help-words are assumed to be encoded in binary. So, for such strategies we write $S(f^m, w)$ to denote the program output by S. Since there are infinitely many functions to learn, a parameterization is necessary. That is, we allow for every n a possibly different help-word w and we require the strategy to learn every recursive function contained in $\{\psi_0, \ldots, \psi_n\}$ with respect to the numbering ψ.

Definition 2.5 (Freivalds and Zeugmann [25]) Let $\psi \in \mathcal{P}^2$, let $\mathcal{U} \subseteq \mathcal{R}$, and let $d \in \mathcal{R}$. A strategy $S \in \mathcal{P}^2$ *finitely infers $\mathcal{U}$ with nonconstructivity $d(n)$ with respect to ψ*, if for each $n \in \mathbb{N}$ there is a help-word w of length at most $d(n)$ such that for every $f \in \mathcal{U} \cap \{\psi_0, \psi_1, \ldots, \psi_n\}$

(1) $S(f^m, w)$ is defined for all $m \in \mathbb{N}$, and
(2) the sequence $(S(f^m, w))_{m \in \mathbb{N}}$ finitely converges to a program i satisfying $\psi_i = f$.

It follows from this definition that a help word w for every n that is larger than the minimal ψ-program i of the target function f also produces a correct result j but it is allowed that the produced result is different from i. This is quite natural, since there may be many distinct ψ-programs for f in the numbering ψ. In particular, if ψ is a Gödel numbering then there are infinitely many ψ-programs for f.

Nonconstructive finite minimal inference and nonconstructive finite nearly minimal identification are defined in a way analogous to the above.

To simplify notation, we make the following convention. Whenever we talk about nonconstructivity $\log n$, we assume that the logarithmic function $\lg n$ to the base 2 is replaced by its integer valued counterpart. That is, we set $\log n =_{df} \lfloor \lg n \rfloor + 1$ and $\log 0 =_{df} 1$, where $\lfloor x \rfloor$ denotes the *floor function*.

As far as the present paper is concerned, the following results obtained in [25] are relevant. Below we use succ to denote the successor function, i.e., $\mathrm{succ}(n) =_{df} n+1$ for all $n \in \mathbb{N}$.

Theorem 1. Let $\varphi \in \mathit{Göd}$ be arbitrarily fixed. Then there is a strategy $S \in \mathcal{P}^2$ such that the class $\mathcal{R}$ can be φ-minimal finitely identified with nonconstructivity $\mathrm{succ}(n)$ with respect to φ.

Theorem 2. Let $\mathcal{U}$ be any indexed family, and let $\psi \in \mathcal{R}^2$ be any numbering for $\mathcal{U}$. Then there is a strategy $S \in \mathcal{P}^2$ such that the class $\mathcal{U}$ can be ψ-minimal finitely identified with nonconstructivity $2 \cdot \log n$ with respect to ψ.

Theorem 3. There is an indexed family $\mathcal{U}$ and a numbering $\psi \in \mathcal{R}^2$ for it such that no strategy $S \in \mathcal{P}^2$ can ψ-minimal finitely identify the class $\mathcal{U}$ with nonconstructivity $c \cdot \log n$ with respect to ψ, where $c \in (0, 1)$ is any constant.

2.3. *Finite Frequency Learning*

The notion of frequency computation was introduced by G. Rose [5] as an attempt to have an absolutely deterministic mechanism with properties similar to probabilistic algorithms. The definition was as follows. Let $\mathbb{N} = \{0, 1, 2, \ldots\}$ denote the set of all natural numbers. A function $f{:}\,\mathbb{N} \to \mathbb{N}$ is said to be (m,n)-*computable*, where $1 \leq m \leq n$, $m, n \in \mathbb{N}$, iff there exists a recursive function $R{:}\,\mathbb{N}^n \to \mathbb{N}^n$ such that, for all n-tuples $(x_1, \cdots, x_n) \in \mathbb{N}^n$ of mutually distinct natural numbers,

$$|\{i \mid (R(x_1, \cdots, x_n))_i = f(x_i)\ ,\ 1 \leq i \leq n\}| \geq m\ ,$$

where $(R(x_1, \cdots, x_n))_i$ denotes the ith component of $R(x_1, \cdots, x_n)$.

McNaughton [26] cites in his survey a problem (posed by Myhill) whether f has to be recursive if m is close to n. This problem was answered by Trakhtenbrot [27] who showed the following.

Theorem 4. If a function $f{:}\,\mathbb{N} \to \mathbb{N}$ is (m,n)-computable, where $2m > n$, then f is recursive.

On the other hand, Trakhtenbrot [27] proved that, if $2m = n$ then nonrecursive functions can be (m,n)-computed. Kinber [28, 29] extended the research by considering frequency enumeration of sets, and this was further studied by Austinat *et al.* [30]. The class of (m,n)-computable sets equals the class of recursive sets if and only if $2m > n$. The notion of frequency computation can be extended to other models of computation. Frequency computation in polynomial time was discussed in full detail by Hinrichs and Wechsung [31]. Frequency computations became increasingly popular when the relation between frequency computation and computation with a small number of queries was discovered [32, 33, 34, 35].

So it is only natural to study finite frequency learning, too. The corresponding learning model was introduced by Kinber *et al.* [36] and called parallel learning. In (m,n)-*finite frequency learning* the strategy takes as input an n-tuple $(f_1^x, \ldots, f_n^x)$ of initial segments of pairwise different target functions and outputs an n-tuple $(i_1^x, \ldots, i_n^x)$ of hypotheses. The notion of finite convergence of sequences of numbers directly carries over to n-tuples of numbers. Therefore, one requires the sequence of n-tuples $(i_1^x, \ldots, i_n^x)_{x \in \mathbb{N}}$ to converge finitely. The learner is successful if at least m functions have been learned finitely.

However, in the present paper we shall deal only with finite minimal frequency learnability which is formally defined as follows.

Definition 2.6. Let $\mathcal{U} \subseteq \mathcal{R}$, let $\psi \in \mathcal{P}^2$, and let $m, n \in \mathbb{N}^+$, where $m \leq n$. The class $\mathcal{U}$ is said to be *finitely ψ-minimal learnable with frequency (m,n) with respect to ψ* if there is a strategy $S \in \mathcal{P}^n$ that takes as input growing initial segments of pairwise different functions $f_1, \ldots, f_n \in \mathcal{U}$ and outputs n-tuples of natural numbers such that

(1) $S(f_1^x, \ldots, f_n^x)$ is defined for all $x \in \mathbb{N}$,

(2) the sequence $(S(f_1^x, \ldots, f_n^x))_{x \in \mathbb{N}}$ finitely converges to an n-tuple $(i_1, \ldots, i_n)$, and
(3) $|\{j \mid 1 \leq j \leq n \text{ and } i_j = \min_\psi f_j\}| \geq m$.

If $\mathcal{U}$ is finitely ψ-minimal learnable with frequency (m, n) with respect to ψ by a strategy S then we write $\mathcal{U} \in \mathit{MIN\text{-}FIN}_\psi^{(m,n)}(S)$. The learning types $\mathit{MIN\text{-}FIN}_\psi^{(m,n)}$ and $\mathit{MIN\text{-}FIN}^{(m,n)}$ are defined analogously to the above.

2.4. *Probabilistic Learning of Recursive Functions*

Finite identification of classes of recursive functions by probabilistic strategies was introduced by Freivalds [37] and has found considerable attention ever since (cf., e.g., [38, 39, 40] and the references therein).

Intuitively speaking, a probabilistic strategy is allowed to flip a fair coin in each step of its computation and then to branch its computation in dependence on the outcome of its coin-flip. So we can assume that a probabilistic strategy is realized by a three tape Turing machine. On its first semi-infinite and read-only tape the results of its coin-flips are written. Once such a sequence s is fixed the machine works as a deterministic strategy with the additional information s. On the second semi-infinite and read-only tape all the values $f(0), f(1), f(2), \ldots$ of the target function f are written. The third tape of the machine is write-only. The machine computes then a sequence of natural numbers which is written on its third tape. If this sequence converges finitely to an index j such that $\psi_j = f$ then we say that the machine was successful. The probability that S finitely learns f is then defined by the usual Borel measure on the set of all infinite 0-1-sequences s such that, on input s and f, the strategy outputs a sequence that finitely converges to a correct ψ-program for f.

Definition 2.7. Let $\psi \in \mathcal{P}^2$ be any numbering. We say that a probabilistic strategy S *FIN-learns a function f with probability p with respect to ψ* if with probability no less than p there is a computation path such that the strategy S produces a correct result on f, i.e., a number j such that $\psi_j = f$, and
A probabilistic strategy S *FIN-learns a class $\mathcal{U} \subseteq \mathcal{R}$ with probability p with respect to ψ* if it *FIN*-learns every function $f \in \mathcal{U}$ with probability p with respect to ψ.

A class $\mathcal{U} \subseteq \mathcal{R}$ is said to be *FIN-learnable with probability p with respect to ψ* if there exists a strategy $S \in \mathcal{P}$ that *FIN*-learns $\mathcal{U}$ with probability p with respect to ψ.

Definition 2.7 can be directly generalized to probabilistic *MIN-FIN*-learning and to probabilistic *NEARLY-MIN-FIN*-learning in the obvious way.

3. Results

After having defined various learning models we are in the position to investigate the impact of the different technological choices on the resulting learning paradigms.

We start this section by establishing several trade-offs between deterministic inference, probabilistic identification, frequency learning, and learning allowing for a certain amount of nonconstructivity. In order to achieve good trade-offs, we also vary the learning goal to a certain extent. That is, we look just at finite learning, at finite minimal inference, and at finite nearly minimal identification, while varying or not varying between deterministic learning methods, probabilistic inference methods, nonconstructive learning methods, and frequency learning (cf. Theorem 5 and Theorem 6).

Theorem 5. There is a Gödel numbering $\varphi \in \mathcal{P}^2$ and an indexed family $\mathcal{U}$ of recursive functions such that

(1) $\mathcal{U} \in$ *FIN*,
(2) $\mathcal{U}$ can be probabilistically *MIN-FIN*$_\varphi$-identified with probability $\frac{1}{2}$,
(3) $\mathcal{U}$ can be probabilistically *NEARLY-MIN-FIN*$_\varphi$-identified with probability $\frac{2}{3}$,
(4) $\mathcal{U}$ cannot be deterministically *MIN-FIN*$_\varphi$-identified,
(5) $\mathcal{U}$ cannot be *MIN-FIN*$_\varphi^{(m,n)}$-identified for all $m, n \in \mathbb{N}$, $m \leq n$, provided $\frac{m}{n} > \frac{1}{2}$,
(6) $\mathcal{U}$ cannot be deterministically *MIN-FIN*$_\varphi$-identified with nonconstructivity $o(\log n)$.

Proof. Our proof consists of four parts. First, we construct an indexed family V of recursive functions such that V cannot be *MIN-FIN*$_\varphi$-inferred with nonconstructivity $o(\log n)$. Second, we define a subfamily V' of V consisting of constant functions only. Third, we construct a Gödel numbering φ and a new indexed family of constants $\mathcal{U}$ being a subfamily of V' with all the needed properties. Fourth, we prove the existence of the needed probabilistic strategies.

Part I. We consider a specific bijection $c\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ which can be traced back to Pepis [41] and Kalmár [42], i.e., $c(u, v) = (2^u - 1) + 2^{u+1}v$. In this way, if $w = c(u, v)$ for any arbitrarily fixed u, then the value of w corresponding to (u, v) is linear in v.

The class V is defined by constructing a numbering $\tau \in \mathcal{R}^2$ and then defining the class V to be $V = \mathcal{R}_\tau$. Let $k \in \mathbb{N}$ be arbitrarily fixed and let $u, v \in \mathbb{N}$ be the uniquely determined numbers such that $k = c(u, v)$.

First, we define the values $\tau_{3k}(0) = \tau_{3k+1}(0) = \tau_{3k+2}(0) = k$. Hence, for any arbitrarily fixed k, the minimal τ-program of any function f with a value $f(0) = k$ can be only $3k, 3k + 1$ or $3k + 2$.

Furthermore, we interpret every function in $\mathcal{P}^2$ as a strategy and obtain thus an effective enumeration S_0, S_1, S_2, ... of all possible strategies. We proceed inductively. Below, for $\ell \in \mathbb{N}$, we use the shortcut $k^{\ell+1}$ to denote the encoding of f^ℓ of the initial segment of any function f for which $f(z) = k$ for $z = 0, \ldots, \ell$.

In order to define $\tau_{3k}(n), \tau_{3k+1}(n), \tau_{3k+2}(n)$ for $n > 1$ we dovetail the following computations. We successively define the function values

$$\tau_{3k}(n-1) = \tau_{3k+1}(n-1) = \tau_{3k+2}(n-1) = k$$

for $n = 2, 3 \ldots$, until we find the least n such that the following Conditions (A) and (B) are satisfied.

(A) There is an $\ell < n$ such that each of the values $S_u(k, v)$, $\ldots$, $S_u(k^{\ell+1}, v)$ turns out to be computable in at most n steps of computation.
(B) $S_u(k, v) \neq S_u(k^2, v) \neq \cdots \neq S_u(k^{\ell}, v) = S_u(k^{\ell+1}, v)$.

If Conditions (A) and (B) never turn out to be satisfied then the function values $\tau_{3k}(n), \tau_{3k+1}(n), \tau_{3k+2}(n)$ are defined for all $n \in \mathbb{N}$, and thus $\tau_{3k}, \tau_{3k+1}, \tau_{3k+2} \in \mathcal{R}$.

On the other hand, if Conditions (A) and (B) turn out to be satisfied then Condition (B) indicates the sequence $S_u(k, v)$, $\ldots$, $S_u(k^{\ell+1}, v)$ may have converged finitely. That is, it either has converged finitely or it cannot converge finitely at all.

Assuming that Conditions (A) and (B) turn out to be satisfied, we continue to define the functions $\tau_{3k}, \tau_{3k+1}, \tau_{3k+2}$ as follows. We define $z(k) = S_u(k^{\ell}, v)$. If $z(k) \in \{3k, 3k+1, 3k+2\}$ then we define $\tau_{z(k)}(x) = k+1$ for all $x \geq n$ and for the remaining $m \in \{3k, 3k+1, 3k+2\} \setminus \{z(k)\}$ we set $\tau_m(x) = k$ for all $x \geq n$. Thus, by construction we have $\tau \in \mathcal{R}^2$ and so $V = \mathcal{R}_\tau$ is an indexed family.

Now suppose that S_u is an arbitrary strategy that finitely τ-minimal learns V with a certain amount of nonconstructivity. We wish to estimate the amount of nonconstructivity needed to achieve this.

To do this, we wish to estimate a number q such that s bits of nonconstructivity do not suffice for S_u to learn all functions in $\{\tau_0, \tau_1, \ldots, \tau_q\}$ in the desired sense. Let $t = 2^s - 1$ be the largest natural number in binary notation which does not exceed s bits. Consider the values $k_0 = c(u, 0), k_1 = c(u, 1), k_2 = c(u, 2), \ldots k_t = c(u, t)$. By the choice of the bijection c we know that $k_t = c(u, t)$ equals $(2^u - 1) + 2^{u+1}t$ and thus k_t does not exceed $2^{u+2}t$.

For each $i \in \{0, 1, \ldots, t\}$ either $z(k_i)$ is defined or it is not defined. If $z(k_i)$ is defined then S_u does not learn $\tau_{z(k_i)}$ finitely with respect to τ. If $z(k_i)$ is not defined then S_u does not learn finitely any of the functions $\tau_{3k_i}, \tau_{3k_i+1}, \tau_{3k_i+2}$. Hence there always is a function in $\{\tau_0, \tau_1, \ldots, \tau_{3k_t+2}\} \subseteq \{\tau_0, \tau_1, \ldots, \tau_{3 \cdot 2^{u+2} k_t + 2}\}$ which is not learnable by the strategy S_u with at most s bits of nonconstructivity.

So, if $q \geq 3 \cdot 2^{u+2} k_t + 2 = \text{const} \cdot t = \text{const} \cdot 2^s$ then the additional help information fails to produce a correct result by the strategy S_u for at least one function in $\{\tau_0, \tau_1, \ldots, \tau_q\}$. It follows that S_u needs at least $\log q$ bits of nonconstructivity to learn all the functions in $\{\tau_0, \tau_1, \ldots, \tau_q\}$.

Part II. We define the class $V' \subseteq V$ to be the subclass of V that consists of all the constant functions in V. This family is *FIN*-learnable because the family of all the constant functions is *FIN*-learnable with respect to any Gödel numbering φ.

Part III. We construct a Gödel numbering φ using some standard Gödel numbering ψ of $\mathcal{P}$, e.g., the numbering using all possible Turing machines and our indexed family $V = \{\tau_n \mid n \in \mathbb{N}\}$ of recursive functions. For all $n \in \mathbb{N}$ and

all $i \in \{0, 1, \ldots, 14\}$ we define

$$\varphi_{15n+i} =_{df} \begin{cases} \psi_{3n+i}, & \text{if } i = 0, 1, 2 \ ; \\ \tau_{12n+i-3}, & \text{if } i \in \{3, \ldots, 14\} \ . \end{cases}$$

The family $\mathcal{U}$ consists of those constants from the family V' whose minimal program in the numbering φ is defined to equal some τ_j but not defined to equal some ψ_k (i.e., we demand the minimal program in the numbering φ to be congruent to $3, 4, \ldots, 14$ modulo 15). Notice that the family $\mathcal{U}$ is non-empty because, for arbitrary n, among the φ-programs $15n, 15n+1, \ldots, 15n+14$ there are at least 4 distinct programs for constant functions but only 3 programs for functions defined as $\varphi_{15n} = \psi_{3n}$, $\varphi_{15n+1} = \psi_{3n+1}$, $\varphi_{15n+2} = \psi_{3n+2}$.

Part IV. For all functions in $\mathcal{U}$ the minimal program in the numbering φ is to be congruent to $3, 4, \ldots, 14$ modulo 15 but the construction of V ensures that no strategy with an amount of nonconstructivity $o(\log n)$ can produce this result.

The probabilistic strategy producing minimal φ-programs for the functions in $\mathcal{U}$ reads the value $f(0) = k$ of the target function. Then it computes $d =_{df} 3k \bmod 12$ and $n =_{df} \lfloor k/4 \rfloor$ and outputs $15n + 3 + d$ with probability $1/2$, and $15n + 3 + d + 1$ with probability $1/2$. For every function in $\mathcal{U}$ one of these results is the correct minimal φ-program.

The probabilistic strategy producing nearly minimal φ-programs for the functions in $\mathcal{U}$ reads the value $f(0) = k$ of the target function, computes $d =_{df} 3k \bmod 12$ and $n =_{df} \lfloor k/4 \rfloor$ and outputs $15n + 3 + d$, $15n + 3 + d + 1$, and $15n + 3 + d + 2$ each with probability $1/3$. For every function in $\mathcal{U}$ at least two of these results are correct φ-programs and all of them are nearly minimal.

It remains to show Assertion (4). Recall that all functions in $\mathcal{U}$ are constant. If we know that the target function is constant, we have all the information about this function merely from $f(0)$. *MIN-FIN*$_\varphi$-identification of a constant function allows only one possible correct result for any target function. Hence this learning can be described by a function $g(n)$ transforming $n = f(0)$ into the minimum index of f in the numbering φ. Suppose to the contrary that $\mathcal{U}$ is *MIN-FIN*$_\varphi^{(m,n)}$-identified with $\frac{m}{n} > \frac{1}{2}$. Then by Theorem 4, the function g is recursive. However, we have already proved in Assertion (3) that $\mathcal{U}$ cannot be deterministically *MIN-FIN*$_\varphi$-identified. So, we have a contradiction, and thus our supposition must be false. Assertion (4) is shown. □

Theorem 6. There is a Gödel numbering φ and a family $\mathcal{U}$ of recursive functions such that

(1) $\mathcal{U}$ cannot be probabilistically *MIN-FIN*$_\varphi$-identified with a probability exceeding $\frac{1}{2}$,
(2) $\mathcal{U}$ can be *MIN-FIN*$_\varphi^{(n-1,n)}$-identified for every $n \in \mathbb{N}^+$,
(3) $\mathcal{U}$ cannot be deterministically *MIN-FIN*$_\varphi$-identified.

Proof. Let $\varphi' \in \mathcal{P}^2$ be any arbitrarily fixed Gödel numbering of $\mathcal{P}$. Without loss of generality we can assume that φ'_0 is the nowhere defined function.

We interprete every φ'_i, $i \in \mathbb{N}$, as a probabilistic strategy and thus obtain an effective enumeration $(S_i)_{i\in\mathbb{N}}$ of all probabilistic strategies. Note that S_0 is then the nowhere defined probabilistic strategy. Therefore, S_0 does not *MIN-FIN*-identify any class of recursive functions. Now, our goal is to construct gradually a class $\mathcal{U}$ by eliminating all strategies S_i as the strategy which could *MIN-FIN*-identify $\mathcal{U}$ with a probability exceeding $\frac{1}{2}$. The class $\mathcal{U}$ will be defined non-algorithmically but the functions in $\mathcal{U}$ will be recursive. In order to do so, we start with the definition of the desired Gödel numbering φ. We set $\varphi_0 = \varphi'_0$ and

$$\varphi_{(2v)!} = \varphi'_v \quad \text{for all } v \in \mathbb{N} \; . \tag{1}$$

So, we have already defined φ_0, φ_1, φ_2, φ_{24}, $\varphi_{720}, \ldots$, and it remains to define the all the functions φ_u, where u is not of the form $(2v)!$ for some $v \in \mathbb{N}$. In order to do so we consider the remaining indices of functions in the numbering φ divided in fragments

$$H(v) = [(2v)! + 1, (2(v+1))! - 1 \quad \text{for all } v \in \mathbb{N}^+ \; . \tag{2}$$

For example, we have $H(1) = [3, 23]$ and $H(2) = [25, 719]$. When all these functions in the numbering φ are defined then the definition given in (1) ensures that the numbering φ is a Gödel numbering of $\mathcal{P}$. The definition of the desired class $\mathcal{U}$ and of the remaining functions in the numbering φ is done simultaneously.

The functions $f \in \mathcal{U}$ may have only one value x such that $f(x) = 0$. This value of x separates the *head* and the *tail* of the function f. Each function $f \in \mathcal{U}$ is constructed to eliminate some strategy S_i.

The head of any function $f \in \mathcal{U}$ is such that the values $f(0), f(1), \cdots, f(x-1)$ contain full information about the minimum φ-programs of all functions $f \in \mathcal{U}$ included to eliminate the strategies $S_0, S_1, S_2, \ldots, S_{i-1}$ but not about the function included to eliminate the strategy S_i. All the values $f(0), f(1), \cdots, f(x-1)$ equal or exceed 1. All the values $f(x+1), f(x+2), \cdots$ in the tail of the function $f \in \mathcal{U}$ exceed 1.

The construction of φ is divided in time-steps $t = 1, 2, 3, \ldots$. In each of these time-steps we deal with one fragment $H(v)$ during a finite number of steps of computation. During this time-step we try to eliminate the strategy S_v.

By $G(v,t)$ we denote a current conjecture $G(v,t) \subseteq H(v)$ which functions f from the fragment $H(v)$ are likely to enter the class $\mathcal{U}$. For any $v \in \mathbb{N}^+$ and $t_1, t_2 \in \mathbb{N}^+$ we have

$$(t_2 > t_1) \text{ implies } G(v,t_2) \subseteq G(v,t_1) \; .$$

A current conjecture $G(v,t)$ may be changed during the subsequent time-steps, but only a finite number of times.

We consider these fragments $H(v)$ during t steps of computation in the following order:

$$(v = 1, t = 1)$$

$$(v = 1, t = 2), (v = 2, t = 3)$$

$$(v = 1, t = 4), (v = 2, t = 5), (v = 3, t = 6)$$

$$\dots$$

Hence we return to every fragment $H(v)$ infinitely many times. When considering the fragment $H(v)$ during t steps of computation we distinguish whether or not the fragment has been considered earlier. If *not*, then we take the current conjectures

$$G(1, t), G(2, t), \dots, G(v - 1, t)$$

which functions from preceding fragments are likely to enter $\mathcal{U}$. We start $v + 1$ distinct new functions in this fragment. We define every such new function φ_u containing in its head $\varphi_u(0), \varphi_u(1), \varphi_u(t-1)$ all the information about all the functions in the fragments

$$H(1), \dots, H(v - 1) \tag{3}$$

that have not yet been in construction. (There would be no difficulty to incorporate all this information into $\varphi_u(0)$ alone.) We add the value $\varphi_u(t) = 0$.

We start constructing the tails of these $v + 1$ functions by using distinct constants $c > 0$. Since only v functions $\varphi_{2!}, \varphi_{4!}, \dots, \varphi_{(2v)!}$ precede the fragment $H(v) = [(2v)! + 1, (2(v + 1))! - 1]$, at least one of these new functions is not among $\varphi_{2!}, \varphi_{4!}, \dots, \varphi_{(2v)!}$.

If *yes* then for every function φ_u in this fragment whose construction is not yet stopped, we perform the strategy S_v on this function during t steps of computation. (By saying "we perform the strategy S_v on this function" we mean that if the strategy asks for values of the target function already defined, we give the correct value. However, if the strategy asks for values of the target function not yet defined, we use values c for the tail of the function and we add values $\varphi_u(y) = c$ to the definition of this function for all $y \geq t$.) Since S_v is a probabilistic strategy, we compute all possible computation paths during t steps of computation and obtain the probabilities of all possible results.

We define $G(v, t + 1)$ consisting of all these $v + 1$ newly added functions φ_u. Additionally we define $G(a, t+1) = G(a, t)$ for all nonempty $G(a, t)$ defined earlier.

If S_v has produced the result u with a probability exceeding $\frac{1}{2}$ then we stop the construction of φ_u leaving infinitely many values not defined. We remove u from $G(v, t+1)$. Simultaneously, we start a new function φ_w in the same fragment $H(v)$ using the same values where φ_u was defined and giving this function a different φ-index $w \in H(v)$ taking a constant c not used in earlier stages of our construction and

defining $\varphi_w(y) = c$ for all $y \geq t$. In this way, the strategy S_v has produced a wrong result u with a probability exceeding $\frac{1}{2}$ for this function. We add the function φ_w to the class $\mathcal{U}$ and remove all remaining numbers but w from $G(v, t+1)$.

If S_v has produced the result different from u with a probability exceeding $\frac{1}{2}$ then we define $\varphi_u(y) = \varphi_t$ for all $y \geq t$, add the function φ_u to the class $\mathcal{U}$ and remove all remaining numbers but u from $G(v, t+1)$. The strategy S_v has produced a wrong result with a probability exceeding $\frac{1}{2}$ for this function.

In all three cases we end the consideration of the fragment $H(v)$ during t steps of computation by going to the next time-step.

If a strategy S_v fails to *MIN-FIN*-identify some function from the class $\mathcal{U}$ by producing a wrong result then we see this after a finite number of time-steps. However, it is possible that for some v the strategy S_v never produces any result with a probability exceeding $\frac{1}{2}$ for a function in $H(v)$. Then our construction returns to the fragment $H(v)$ infinitely many times and several functions in this fragment become total. In this case we non-algorithmically choose one of these total functions to enter the class $\mathcal{U}$. The strategy S_v fails on this function because it produces no result. Hence, Assertion (1) is shown.

Clearly, Assertion (1) directly implies Assertion (3).

Assertion (2) is implied by the sentence containing the specification (3) of the fragments to be considered when starting the definition of φ_u in our proof. Since the indices of the strategies $S_0, S_1, S_2, \ldots, S_i$ are well-ordered, and all the target functions in the n-tuple are in the class $\mathcal{U}$, one of the target functions contains this information for all the other functions but not for itself. Hence we can output the correct minimal programs for all the other functions but not for this one. This gives us the frequency $(n-1, n)$. □

4. Conclusions and Open Problems

In the present paper we considered four different models of finite learning of classes of recursive functions, i.e., deterministic finite learning algorithms, probabilistic finite learning strategies, inference algorithms that learn finitely with a certain frequency, and finite learners that are allowed to use a certain amount of nonconstructive information during the learning process. The resulting learning models have been compared to one another, and several trade-off have been shown. Basically, the results obtained showed that probabilistic learners, frequency inference algorithms, and finite learners using a certain amount of nonconstructivity all extend the learning power of deterministic finite learning, but in *different* directions.

It remained open to characterize these learning models in terms of computable numberings and/or complexity theoretic properties. Such characterizations have been undertaken for various learning models and turned out to be extremely useful for a deeper understanding of what properties allow a class of recursive functions to be learnable in one model or another (cf. Zeugmann and Zilles [18] and the references therein for more information).

It would also be desirable to extend finite learning to quantum finite learning and to investigate the problem whether or not finite learning quantum strategies again extend deterministic finite learning in a direction different from the ones considered in this paper.

Furthermore, recently Freivalds [43] and his students have introduced *ultrametric algorithms* using p-adic numbers instead of real numbers to describe transition "probabilities." This model has been developed further in [44] to finite query learning. That is, instead of receiving the graph of a target function as input, the learner is allowed to ask value queries, where the input is any argument x and the teacher has then to return $f(x)$. It has been shown that ultrametric algorithms have advantages even over *nondeterministic* algorithms for certain learning problems in this finite query learning setting. So, it would be interesting to know whether or not similar results could be obtained in the setting considered in this paper.

Acknowledgments. We would like to express our gratitude to the anonymous reviewers for their careful reading and the helpful suggestions made.

References

[1] R. M. Karp and R. J. Lipton, "Turing machines that take advice," *L' Enseignement Mathématique*, vol. 28, pp. 191–209, 1982.

[2] C. Damm and M. Holzer, "Automata that take advice," in *Mathematical Foundations of Computer Science 1995, 20th International Symposium, MFCS '95, Prague, Czech Republic, August 28 - September 1, 1995, Proceedings*, vol. 969 of *Lecture Notes in Computer Science*, (Berlin), pp. 149–158, Springer, 1995.

[3] R. Freivalds, "Amount of nonconstructivity in deterministic finite automata," *Theoret. Comput. Sci.*, vol. 411, no. 38-39, pp. 3436–3443, 2010.

[4] E. M. Gold, "Language identification in the limit," *Inform. Control*, vol. 10, no. 5, pp. 447–474, 1967.

[5] G. F. Rose, "An extended notion of computability," in *International Congress for Logic, Methodology and Philosophy of Science, Stanford University, Stanford, California, August 24 - September 2, 1960, Abstracts of contributed papers*, 1960.

[6] M. Li and P. Vitányi, *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 3rd ed., 2008.

[7] R. J. Solomonoff, "A formal theory of inductive inference: Part 1," *Inform. Control*, vol. 7, no. 1, pp. 1–22, 1964.

[8] R. J. Solomonoff, "A formal theory of inductive inference: Part 2," *Inform. Control*, vol. 7, no. 2, pp. 224–254, 1964.

[9] J. Case and C. Smith, "Comparison of identification criteria for machine inductive inference," *Theoret. Comput. Sci.*, vol. 25, no. 2, pp. 193–220, 1983.

[10] G. Baliga, J. Case, and S. Jain, "Language learning with some negative information," *J. of Comput. Syst. Sci.*, vol. 51, no. 2, pp. 273–285, 1995.

[11] R. Freivald and R. Wiehagen, "Inductive inference with additional information," *Elektronische Informationsverarbeitung und Kybernetik*, vol. 15, no. 4, pp. 179–185, 1979.

[12] S. Jain and A. Sharma, "Learning in the presence of partial explanations," *Inform. Comput.*, vol. 95, no. 2, pp. 162–191, 1991.

[13] S. Jain and A. Sharma, "Learning with the knowledge of an upper bound on program size," *Inform. Comput.*, vol. 102, no. 1, pp. 118–166, 1993.

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

[15] Y. L. Ershov, "Theory of numberings," in *Handbook of Computability Theory* (E. R. Griffor, ed.), vol. 140 of *Studies in Logic and the Foundations of Mathematics*, pp. 473–503, Elsevier, 1999.

[16] R. Freivalds, "Inductive inference of recursive functions: Qualitative theory," in *Baltic Computer Science* (J. Bārzdiņš and D. Bjørner, eds.), vol. 502 of *Lecture Notes in Computer Science*, pp. 77–110, Berlin: Springer, 1991.

[17] S. Jain, D. Osherson, J. S. Royer, and A. Sharma, *Systems that Learn: An Introduction to Learning Theory, second edition*. Cambridge, Massachusetts: MIT Press, 1999.

[18] T. Zeugmann and S. Zilles, "Learning recursive functions: A survey," *Theoret. Comput. Sci.*, vol. 397, no. 1-3, pp. 4–56, 2008. Special issue Forty Years of Inductive Inference: Dedicated to the 60th Birthday of Rolf Wiehagen.

[19] E. M. Gold, "Limiting recursion," *J. Symbolic Logic*, vol. 30, pp. 28–48, 1965.

[20] R. V. Freivalds and E. B. Kinber, "Identification in the limit of minimal gödel numbers," in *Teoriya Algoritmov i Programm* (Y. M. Barzdin, ed.), vol. III, pp. 3–34, Latvian State University, 1977. (in Russian).

[21] B. Trakhtenbrot and Y. Barzdin, *Finite Automata, Behavior and Synthesis*. Amsterdam: North Holland, 1973.

[22] R. V. Freivald, "Minimal Gödel numbers and their identification in the limit," in *Mathematical Foundations of Computer Science 1975, 4th Symposium, Mariánské Lázně, September 1-5, 1975* (J. Bečvář, ed.), vol. 32 of *Lecture Notes in Computer Science*, (Berlin), pp. 219–225, Springer-Verlag, 1975.

[23] Е.Б.. Кинбер, "О предельном синтезе почти минимальных геделевских номеров," in Теория Алгоритмов и Программ (J. Bārzdiņš, ed.), vol. I, pp. 212–223, Latvian State University, 1974.

[24] K.-J. Chen, "Tradeoffs in the inductive inference of nearly minimal size programs," *Inform. Control*, vol. 52, no. 1, pp. 68–86, 1982.

[25] R. Freivalds and T. Zeugmann, "On the amount of nonconstructivity in learning recursive functions," in *Theory and Applications of Models of Computation, 8th Annual Conference, TAMC 2011, Tokyo, Japan, May 23-25, 2011, Proceedings* (M. Ogihara and J. Tarui, eds.), vol. 6648 of *Lecture Notes in Computer Science*, pp. 332–343, Springer, 2011.

[26] R. McNaughton, "The theory of automata, a survey," *Advances in Computers*, vol. 2, pp. 379–421, 1961.

[27] B. A. Trakhtenbrot, "On the frequency computation of functions," *Algebra i Logika*, vol. 2, no. 1, pp. 25–32, 1964. (in Russian).

[28] E. B. Kinber, "Frequency calculations of general recursive predicates and frequency enumerations of sets," *Soviet Mathematics*, vol. 13, pp. 873–876, 1972.

[29] E. B. Kinber, "Frequency computations in finite automata," *Cybernetics and Systems Analysis*, vol. 12, no. 2, pp. 179–187, 1976.

[30] H. Austinat, V. Diekert, U. Hertrampf, and H. Petersen, "Regular frequency computations," *Theoret. Comput. Sci.*, vol. 330, no. 1, pp. 15–21, 2005.

[31] M. Hinrichs and G. Wechsung, "Time bounded frequency computations," *Inform. Comput.*, vol. 139, no. 2, pp. 234–257, 1997.

[32] M. Kummer, "A proof of Beigel's cardinality conjecture," *J. Symbolic Logic*, vol. 57, no. 2, pp. 677–681, 1992.

[33] V. Harizanov, M. Kummer, and J. Owings, "Frequency computations and the cardinality theorem," *J. Symbolic Logic*, vol. 57, no. 2, 1992.

[34] R. Beigel, W. Gasarch, and E. Kinber, "Frequency computation and bounded queries,"

Theoret. Comput. Sci., vol. 163, no. 1-2, pp. 177–192, 1996.

[35] J. Case, S. Kaufmann, E. B. Kinber, and M. Kummer, "Learning recursive functions from approximations," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 183–196, 1997.

[36] E. Kinber, C. H. Smith, M. Velauthapillai, and R. Wiehagen, "On learning multiple concepts in parallel," *J. of Comput. Syst. Sci.*, vol. 50, no. 1, pp. 41–52, 1995.

[37] R. Freivalds, "Finite identification of general recursive functions by probabilistic strategies," in *Fundamentals of computation theory: FCT '79, Proceedings of the Conference on Algebraic, Arithmetic, and Categorial Methods in Computation Theory held in Berlin/Wendisch-Rietz (GDR), Sept. 17-21, 1979* (L. Budach, ed.), (Berlin), pp. 138–145, Akademie-Verlag, 1979.

[38] A. Ambainis, "Probabilistic inductive inference: a survey," *Theoret. Comput. Sci.*, vol. 264, no. 1, pp. 155–167, 2001.

[39] A. Ambainis, K. Apsītis, R. Freivalds, and C. H. Smith, "Hierarchies of probabilistic and team *FIN*-learning," *Theoret. Comput. Sci.*, vol. 261, no. 1, pp. 91–117, 2001.

[40] L. Pitt, "Probabilistic inductive inference," *J. ACM*, vol. 36, no. 2, pp. 383–433, 1989.

[41] J. Pepis, "Ein Verfahren der mathematischen Logik," *J. Symbolic Logic*, vol. 3, no. 2, pp. 61–76, 1938.

[42] L. Kalmár, "On the reduction of the decision problem. First Paper. Ackermann prefix, a single binary predicate," *J. Symbolic Logic*, vol. 4, no. 1, pp. 1–9, 1939.

[43] R. Freivalds, "Ultrametric automata and Turing machines," in *Turing-100* (A. Voronkov, ed.), vol. 10 of *EPiC Series*, pp. 98–112, EasyChair, 2012.

[44] R. Freivalds and T. Zeugmann, "Active learning of recursive functions by ultrametric algorithms," in *SOFSEM 2014: Theory and Practice of Computer Science, 40th International Conference on Current Trends in Theory and Practice of Computer Science, Nový Smokovec, Slovakia, January 26-29, 2014, Proceedings* (V. Geffert, B. Preneel, B. Rovan, J. Štuller, and A. M. Tjoa, eds.), vol. 8327 of *Lecture Notes in Computer Science*, (Cham, Heidelberg, New York, Dordrecht, London), pp. 246–257, Springer International Publishing Switzerland, 2014.