Festschrift

LNCS 5760

Susanne Albers
Helmut Alt
Stefan Näher (Eds.)

Efficient Algorithms

Essays Dedicated to Kurt Mehlhorn on the Occasion of His 60th Birthday

Springer

Lecture Notes in Computer Science 5760

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Susanne Albers Helmut Alt
Stefan Näher (Eds.)

Efficient Algorithms

Essays Dedicated to Kurt Mehlhorn
on the Occasion of His 60th Birthday

Springer

Volume Editors

Susanne Albers
Humboldt-Universität zu Berlin, Institut für Informatik
10099 Berlin, Germany
E-mail: albers@informatik.hu-berlin.de

Helmut Alt
Freie Universität Berlin, Institut für Informatik
Takustr. 9, 14195 Berlin, Germany
E-mail: alt@mi.fu-berlin.de

Stefan Näher
Universität Trier, Fachbereich IV - Informatik
54286 Trier, Germany
E-mail: naeher@uni-trier.de

The illustration appearing on the cover of this book is the work of Daniel Rozenberg (DADARA)

Library of Congress Control Number: 2009932882

CR Subject Classification (1998): F.2, G.1, G.2, I.1.2, G.4, E.5

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-642-03455-1 Springer Berlin Heidelberg New York
ISBN-13 978-3-642-03455-8 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12731985 06/3180 5 4 3 2 1 0

Kurt Mehlhorn - Born 1949

Preface

"Effiziente Algorithmen" was the title of the first book by Kurt Mehlhorn in 1977. It was meant as a text for graduate students and published in German by Teubner-Verlag.

We decided to adopt this title 32 years later for this Festschrift in honor of Kurt on the occasion of his 60th birthday. It contains contributions by his former PhD students, many of whom are now university teachers themselves, and colleagues with whom he cooperated closely within his career. It is our pleasure that even Kurt's former PhD advisor, Bob Constable from Cornell University, kindly agreed to contribute. Many of the contributions were presented at a colloquium held in Kurt's honor on August 27 and 28, 2009 in Saarbrücken, Germany.

This Festschrift shows clearly how the field of algorithmics has developed and matured in the decades since Kurt wrote his book with the same title.

The classic approach based on discrete mathematics and computability and complexity theory continues to be the foundation of the field with ever new and important challenges as the first chapters of this Festschrift show. Kurt has contributed significantly to classical algorithmics and gained worldwide reputation. Starting from research in computability theory in his PhD thesis he made major contributions to complexity theory, graph algorithms, data structures, and was one of the first to recognize the significance of computational geometry contributing one of the early textbooks on the subject.

In spite of the success of classical algorithmics, by the 1990s more and more researchers recognized that in order to have their results acknowledged in the scientific community as a whole and applied commercially it was necessary that they took care of the implementation of algorithms themselves. It turned out that this aspect of algorithmics created challenging new theoretic problems, in particular, concerning software engineering, a closer investigation of heuristic methods, the numerical robustness of algorithms, and as a possible solution to this problem, exact computation. Peter Sanders' contribution to the Festschrift, for example, describes this new field of algorithm engineering and Chee Yap's contribution is a convincing plea why numerical computing is of great importance to the field of algorithmics.

Kurt Mehlhorn was one of the initiators of this new development of algorithmics and became one of its leaders and driving forces worldwide. In particular, he and his group created the software library LEDA, a uniquely comprehensive collection containing implementations of all the classical algorithms and being used extensively in academia and industry. In addition, considerable research was done by them on the theoretical aspects of implementing algorithms especially concerning robustness and exactness of computation.

Kurt has been a leader not only in scientific research but also in scientific organizations, in particular during his years as a vice president of the Max Planck society. The administrative work never prevented him from being a productive researcher which he continues to be up to this day.

So let us honor this eminent scientist, whom to our privilege we have had as a teacher and a friend.

Happy Birthday, Kurt!

August 2009

Susanne Albers
Helmut Alt
Stefan Näher

Acknowledgements

We would like to thank everybody who contributed to this Festschrift: the authors for their interesting articles, the colleagues and PhD students who helped proofread the contributions, Marc Scherfenberg for collecting, editing, and compiling the files of all the authors, and Wolfgang J. Paul and his team for organizing the colloquium in Saarbrücken.

The Editors

Table of Contents

I Models of Computation and Complexity

II Sorting and Searching

III Combinatorial Optimization with Applications

IV Computational Geometry and Geometric Graphs

V Algorithm Engineering, Exactness, and Robustness

Part I

Models of Computation and Complexity

Kolmogorov-Complexity Based on Infinite Computations

Günter Hotz

Fakultät für Mathematik und Informatik
Universität des Saarlandes

Abstract. We base the theory of Kolmogorov complexity on programs running on a special universal machine M, which computes infinite binary sequences $x \in \{0,1\}^\infty$. The programs are infinite sequences $p \in \{0,1\}^* \cdot 1 \cdot 0^\infty$. As length $|p|$ we define the length of the longest prefix of p ending with 1. We measure the distance $d(x,y) = 2^{-n}$ of $x, y \in \{0,1\}^\infty$ by the length n of the longest common prefix of x and y. $\Delta_M(x, 2^{-n})$ is the length of a minimal program p computing a sequence y with $d(x,y) \leq 2^{-n}$. It holds $\Delta_M(x, 2^{-n}) \leq \Delta_M(x, 2^{-(n+1)}) \leq n+2$ for all n. We prove that the sets of sequences

$$K_{\Delta_M} := \bigcup_{c \in \mathcal{N}} \{x \in X^\infty : \Delta_M(x, 2^{-n}) > n - c \quad \text{for all n}\}$$

$$K^{o(n)}_{\Delta_M} := \{x \in X^\infty : n + 1 - \Delta_M(x, 2^{-n}) = o(n)\}$$

have the measure 1 for memoryless sources with equal probabilities for 0 and 1. The sequences in $K^{o(n)}_{\Delta_M}$ are Bernoulli sequences. The sequences in K_{Δ_M} define collectives in the sense of von Mises up to a set of measure 0 and the sequences in $K^{o(n)}_{\Delta_M}$ have this property in a certain resricted fom.

Keywords: monotone Kolmogorov complexity, infinite computations, collectives.

1 Introduction

Kolmogorov [1] based his concept to define binary random sequences on the prefix approximation of the infinite sequences and the size of shortest programs to compute the prefixes. An infinite sequence $x \in X^\infty$ is random in his sense if for all prefixes $x[n]$ of length n of x the difference $|n - |p_n||$ of n and the length $|p_n|$ of a shortest program p_n to compute the prefix on a universal Turing machine is bounded by a constant $c \in \mathcal{N}$ depending on x and the used universal machine. Martin-Löf [2] proved that such sequences x do not exist, but that the concept works, if one substitutes the $\forall n$ by $\forall^\infty n$. This set of random sequences has the measure 1. The reason for the discovered breakdowns of the prefix complexity is a relation between the length of the prefix and the content of the prefix.

This relation in our sense has nothing to do with the character of a sequence to be random but only with the interest on certain prefix lengths. So we change the

S. Albers, H. Alt, and S. Näher (Eds.): Festschrift Mehlhorn, LNCS 5760, pp. 55–73, 2009.

definition of Kolmogorov at another place as Martin-Löf did. We use the approximations of the random sequences by infinite computable sequences. This is close to the concept of Schnorr, who used only programs p, which compute infinite sequences $M(p)$. But in this case too came in the connection with the length of the prefix $x[i]$ of the sequence $x \in \{0,1\}^\infty$ by the condition $M(p)(i) = x[i]$. So if I am interested in a shortest program p with $M(p)(i) = x[i]$ the computer has as additional information i, i.e., $\log(i)$ bits, the reason for the non-monotone growing length of the shortest programs to describe prefixes of the length i.

In our definition we do not expect that the computer on an argument i prints out $x[i]$. We are looking for approximations of x by sequences $M(p)$ of a precision $d(x, M(p)) < \epsilon$ and the size of shortest programs $p_{x,\epsilon}$ with this property. The size of these programs fulfils the monotonicity relation $|p_{x,\epsilon}| \leq |p_{x,\epsilon^*}|$ for $\epsilon > \epsilon^*$. We prove that the set of our random sequences has measure 1.

Additionally, we give a weaker definition for randomness and prove that each sequence in this set is a Bernoulli sequence. This concept allows us, only based on our complexity concept, to classify the sequences with convergent limit probabilities generated by information sources $(\{0,1\}, \mu)$ with $\mu : \{0,1\} \to \mathcal{R}$ a probability distribution.

We additionally prove that a certain subset of these random sequences is invariant under each blind selection of subsequences by computers, this means that *Das Starke Gesetz der grossen Zahlen* for collectives of von Mises [8] holds. The subset for which we are not able to prove this property has measure 0. This we prove without any restriction for $x \in K_{\Delta_M}$ but in the case of $K_{\Delta_M}^{o(n)}$ under a restriction, which has to do with the density of the components of x we select.

We do not need assumptions such as prefix free codes for the programs as introduced by Chaitin [5] to get the original idea of Kolmogorov really working. Essential for the simplicity of the theory is the existence of a print command, the assumption of infinite computations and a special procedure technique. This is not constructive in the classical sense. But the existing theory is not constructive either. So why not use infinite computations if it simplifies the theory.

We use a special universal Turing machine M and because of proof technical reasons two different nets of three such machines, which mathematically are based on the scalar products of the machines and not on the machines simulated by the universal machines. So the state sets are independent from the programs running on the machines.

The books of Calude [7], Li and Vitányi [9], and Chaitin [6] give excellent overviews about the existing theory. This paper is based on ideas presented in [10] especially on the chapter $\Delta^* -$ Complexity. Our concept can easily be extended to more general spaces.

2 Basic Definitions

2.1 The Universal Machine

Our machine is a special universal Turing-machine with **three** tapes: An input tape, a computing tape and an output tape. Each tape uses the binary alphabet

$X := \{0,1\}$ and it is infinitely long to the right side. In the initial state each position on the computing tape and the output tape are equal to 0. X^* is the free monoid of finite words over the alphabet X and X^∞ the set of infinite sequences over X. We define $w \cdot v$ for $w \in X^*$ and $v \in X^* \cup X^\infty$ by concatenation. The inscription of the input tapes are elements of $X^* \cdot 0^\infty$. We define for $x \in X^* \cdot 0^\infty$ and $x \neq 0^\infty$

$$|x| := \mathbf{max}\{i \in \mathcal{N} : x(i) = 1\}$$

and

$$|x| := 1 \quad \text{for} \quad x = 0^\infty.$$

We consider only infinite converging computations. An infinite computation is converging iff the writing head visits each position of the output tape only once. We substitute the halting states of the traditional machine by the cycle state *move to the right*. This means that the terminating finite computations will be transformed into converging infinite computations.

We use different machine models. The machine M has three tapes: An input tape, which is the program tape. The machine has one computing tape and an output tape. The input tape is a read-only tape. The output tape is a write-only tape. Its writing head moves only to the right. It is special in the following sense: It allows a printing command, which we will define later precisely.

Our second machine $\widetilde{M}$ and the third machine $\overline{M}$ have three input tapes T_1, T_2, T_3, for three programs, which are read-only tapes two and three computing tapes, respectively, and one write-only output tape. The printing head of this tape moves only to the right side. The main program is on tape T_1. The tapes T_2 and T_3 are for procedures, which will be called by the main program. The program on tape T_3 of machine $\widetilde{M}$ is always a print instruction. The programs on tape T_2 may be print instructions or any other programs. The print instruction has the form $0p \in X^* \cdot 0^\infty$. The programs of the type $1p \in X^* \cdot 0^\infty$ will be interpreted as usual by the universal machine. The main program of $\widetilde{M}$ uses the computing tape C_1 and the program on tape T_2 uses the computing tape C_2.

The main program p_1 never writes on the output tape. It only calls and controls the activity of the programs p_2 on T_2 and p_3 on T_3. The programs p_2 and p_3 are never both active, but p_1 may be active together with each of the two other programs. If the activity of the programs p_2 and p_3 changes then the writing on the output tape will continue on the position it has been stopped. The machines $\widetilde{M}$ and $\overline{M}$ may be understood as a net of three machines of type M. One machine calls the others and controls them. But they have only one output tape. They are motivated by proof technical reasons. Based on these machines we get some lower bounds for the size of minimal programs of M. The constructions of these machines are mathematically based on standard cartesian products of the universal machines, not of the machines simulated on the universal machines. This is essential for the additivity of the complexities of the running programs.

All convergent computations of a program p of the machine M or of programs $p := p_1(p_2, p_3)$ of the machine $\widetilde{M}$ generate infinite sequences $x \in X^\infty$ on the output tape. We write in this case $x := M(p)$ or $x := \widetilde{M}(p)$, respectively. For the program $p := 0 \cdot w \cdot 0^\infty$, $w \in X^*$ on T_3 we write $print(w)$. It generates

as output $w \cdot 0^\infty$. We look at our programs as infinite sequences with a finite number of 1's. The length of the program is defined as the length of the longest prefix ending with 1.

2.2 Infinite Sequences

We define $\overline{0} := 1$ and $\overline{1} := 0$ and for $x \in X^\infty$

$$\overline{x} := (\overline{x}(1), \overline{x}(2), \overline{x}(3), \ldots).$$

We write $x_i \oplus y_i$ for the addition modulo 2 for $x_i, y_i \in \{0, 1\}$ and for $x, y \in X^\infty$ we understand $x \oplus y$ as the componentwise application of the operation.

We define for $x \in X^\infty$

$$\|x\| := \sum_{i=1}^{\infty} \frac{x(i)}{2^i}$$

and as distance of $x, y \in X^\infty$

$$d(x, y) := \sum_{i=1}^{\infty} \frac{x(i) \oplus y(i)}{2^i}.$$

For $w \in X^*$ and $x \in X^\infty$ we write

$$w \prec x \quad \text{for} \quad w = x[n] := x(1) \cdot x(2) \cdot \cdots \cdot x(n)$$

where n is the length of w. For $x, y \in X^*$ it holds

$$d(x, y) < 2^{-n} \Longrightarrow x[n] = y[n].$$

This means that x, y have a common prefix w of length n for $d(x, y) < 2^{-n}$. If x, y have w as common prefix, then it holds

$$d(x, y) \leq 2^{-n}.$$

We see that for $w \in X^n$ and $x \in X^\infty$ we get

$$d(w1x, w0x) = d(w1x, w1\overline{x}) = 2^{-n}.$$

Using the prefix distance we get for the expression on the right hand side as distance $2^{-[n+1]}$.

3 Kolmogorov - Complexity of Binary Sequences

3.1 Definitions and Simple Consequences

Given a sequence $x \in X^\infty$ we ask for computable sequences $y \in X^\infty$ such that $d(x, y) < \epsilon$ for a given $0 < \epsilon < 1$. We are interested in programs p of minimal length $|p|$, with $M(p) = y \in X^\infty$, which are approximations of a given precision of given $x \in X^\infty$. The program length of the program $p := p_1(p_2, p_3)$ with program p_i on tape T_i of $\widetilde{M}$ or $\overline{M}$ we define as

$$|p| := |p_1| + |p_2| + |p_3|.$$

Definition 1. *Let $x \in X^\infty$, $\epsilon > 0$ and M, $\widetilde{M}$, $\overline{M}$ our universal machines. We define*

$$\Delta_M(x,\epsilon) := \min\{|p| : d(x, M(p)) < \epsilon\},$$
$$\Delta_{\widetilde{M}}(x,\epsilon) := \min\{|p| : d(x, \widetilde{M}(p)) < \epsilon\},$$
$$\Delta_{\overline{M}}(x,\epsilon) := \min\{|p| : d(x, \overline{M}(p)) < \epsilon\}$$

Because each sequence $M(p)$, which is an approximation of precision $\epsilon_2 < \epsilon_1$ of x is also an approximation of precision ϵ_1, it follows

Lemma 1. *For $x \in X^\infty$ and $\epsilon_1 > \epsilon_2 > 0$ it follows*

$$\Delta_M(x,\epsilon_1) \leq \Delta_M(x,\epsilon_2),$$
$$\Delta_{\widetilde{M}}(x,\epsilon_1) \leq \Delta_{\widetilde{M}}(x,\epsilon_2) \quad \textit{and} \quad \Delta_{\overline{M}}(x,\epsilon_1) \leq \Delta_{\overline{M}}(x,\epsilon_2).$$

Remark: We may run on our machines $\widetilde{M}$ and $\overline{M}$ main programs p_1, which do nothing else as one call of the program p_2 or p_3. These programs may be programs for M which compute a best approximation for the given x and ϵ. If $c := |p_1|$, then we have

$$\Delta_M(x,\epsilon) < \Delta_{\widetilde{M}}(x,\epsilon) + c, \quad \Delta_{\overline{M}}(x,\epsilon) + c$$

holds for all $x \in X^\infty$ and $\epsilon > 0$.

We see that our variant of the original definition of Kolmogorov for $\epsilon := 2^{-n}$ leads to a monotone dependence of $\Delta_M(x, 2^{-n})$ from n.

Our *print*-operation guarantees a simple upper bound for the approximations, if we restrict to $\epsilon = 2^{-n}$.

Lemma 2. *Let be $x \in X^\infty$ and $\epsilon = 2^{-n}$ then it holds*

$$\Delta_M(x,\epsilon) < n+1 \quad \textit{and} \quad \Delta_{\widetilde{M}}(x,\epsilon) < n+1+c$$

with c independent from x and n.

3.2 (Δ_M, H)- Approximable Sequences

We define sets of sequences $x \in X^\infty$, which in a certain degree are approximable by computable sequences. Let H be the set of monotone, unbounded, and computable mappings $h : \mathcal{N} \to \mathcal{N}$.

Definition 2. *For $h \in H$ we define the sequence $x \in X^\infty$ approximable of degree h iff there exists $n_0 \in \mathcal{N}$ such that for all $n > n_0$*

$$n + 1 - \Delta(x, 2^{-n}) \geq h(n)$$

holds. We define

$$\Lambda_{\Delta_M}(h, n_0) := \{x \in X^\infty : \forall_{n>n_0}(n + 1 - \Delta_M(x, 2^{-n}) \geq h(n))\}$$

and

$$\Lambda_{\Delta_M}(H) := \bigcup_{h \in H, n_0 \in \mathcal{N}} \Lambda_{\Delta_M}(h, n_0)$$

$h : \mathcal{N} \to \mathcal{N}$. Λ_{Δ_M} is the set of the (Δ_M, H)-approximable sequences.

We first give a lower and upper bound for the average lengths of a minimal set of shortest programs $p \in X^n$, which compute sequences $M(p) \in X^\infty$ such that every word of X^n appears as prefix. An upper bound we get by $n+1$ the maximal length of *print*-commands, which compute the outputs $X^n \cdot 0^\infty$.

Counting the words $w1 \in X^k$ for $k = 1, 2, \ldots, n$ we get

$$\sum_{m=1}^{n-1} 2^m = 2^n - 1.$$

We did not count the *print*-command $print(0)$. So we have 2^n programs of length $\leq n$. By computing the average length of the 2^n shortest programs we get a lower bound for the average length

$$A_n := \sum_{w \in X^n} \frac{\Delta_M(w \cdot X^\infty, 2^{-n})}{2^n}$$

of the minimal programs we are interested in. There are 2^m of our programs of length $m+1$ available. So we get for the average length of the set of programs with length $\leq n-1$ not considering the trivial print command the expression $\frac{B_n}{2^n}$, where B_n is defined as follows:

$$B_n := \sum_{m=0}^{n-1} (m+1) \cdot 2^m = n \cdot 2^{n-1} + B_{n-1}.$$

We get as solution of this recursion

$$B_n = (n-1) \cdot 2^n + 1$$

Taking in account the trivial print command we get as a lower bound for A_n

$$A_n \geq \frac{(n-1) \cdot 2^n + 2}{2^n} = n + \frac{1}{2^{n-1}} - 1.$$

Using the existence of our *print*-commands we get $n+1$ as upper bound. So we proved the following lemma.

Lemma 3

$$n - 1 < \sum_{w \in X^n} \Delta_M(w \cdot X^\infty, 2^{-n}) \cdot 2^{-n} \leq n + 1$$

This is the base for the proof of the following lemma.

Lemma 4. *For the source (X, μ) with $\mu(1) = \mu(0) = 2^{-1}$ and each unbounded mapping $h : \mathcal{N} \to \mathcal{N}$ and each $n_0 \in \mathcal{N}$ it holds*

$$\mu(\Lambda_{\Delta_M}(h, n_0)) = 0$$

Proof: We define

$$\Lambda^n_{\Delta_M}(h) := \{x \in X^\infty : n+1-\Delta_M(x, 2^{-n}) \geq h(n)\}.$$

To decompose a following sum in two parts, we define

$$S_1 := \{w \in X^n : h(n) \leq n+1-\Delta_M(wX^\infty, 2^{-n})\}$$

and

$$S_2 := \{w \in X^n : h(n) > n+1-\Delta_M(wX^\infty, 2^{-n})\}$$

It follows from lemma 3

$$n-1 < \sum_{S_1} \frac{\Delta_M(wX^\infty, 2^{-n})}{2^n} + \sum_{S_2} \frac{\Delta_M(wX^\infty, 2^{-n})}{2^n} \leq n+1$$

and

$$n-1 < (n+1-h(n)) \cdot \sum_{S_1} 2^{-n} + (n+1) \cdot \sum_{S_2} 2^{-n}.$$

We use the abbreviation

$$\mu_n := \mu(\Lambda^n_{\Delta_M}(h))$$

and get

$$n-1 < (n+1-h(n)) \cdot \mu_n + (n+1) \cdot (1-\mu_n)$$

and

$$\mu_n < \frac{2}{h(n)}$$

$h(n)$ is unbounded. Therefore there exists a sequence $n_0 < n_1 < n_2, \ldots$ with $n_i \in \mathcal{N}$ such that $h(n_i) < h(n_{i+1})$ for all $i \in \mathcal{N}$. It follows $\mu_{n_i} \to 0$. Obviously we have

$$\Lambda_{\Delta_M}(h, n_0) \subset \bigcap_{i=0}^{\infty} \Lambda^{n_i}_{\Delta_M}(h)$$

and therefore

$$\mu(\Lambda_{\Delta_M})(h, n_0) = 0$$

So it follows

Theorem 1

$$\mu(\Lambda_{\Delta_M}(H)) = 0.$$

We may substitute the set H of computable functions by a subset of H or we may extend H by non computable monotone mappings or by constant functions. The question is, which influence this has on a theory of random sequences. The following three lemmas give a first answer on this question.

Lemma 5. *To each countable set $\widetilde{H} := \{h_1, h_2, h_3, \ldots\}$ of unbounded monotone mappings there exists an unbounded monotone mapping $f : \mathcal{N} \to \mathcal{N}$, which is an asymptotic lower bound for all $h \in \widetilde{H}$.*

Proof: We define $f(n) := h_1(1)$ for

$$n < n_1 := \min\{k : h_1(k) > h_1(1) + 1, h_2(k) > h_1(1) + 1\}$$

and

$$f(n_1) := \min\{h_1(n_1), h_2(n_1)\} - 1$$

Let n_i and f be defined for $n \leq n_i$, then we define

$$n_{i+1} := \min\{k > n_i : h_1(k), h_2(k), \ldots h_{i+1}(k) > f(n_i) + 1\},$$

$$f(n) := h(n_i) \quad \text{for} \quad n_i < n < n_{i+1}$$

and

$$f(n_{i+1}) := \min\{h_1(n_{i+1}), \ldots h_{i+1}(n_{i+1}))\} - 1$$

This defines f for all $n \in \mathcal{N}$ and it follows $f(n_i) < f(n_{i+1})$ and $f(n) < h_i(n)$ for $n \geq n_i$. h is asymptotically a lower bound for each $h \in \widetilde{H}$ and it is not in $\widetilde{H}$. This ends the proof of the lemma.

Obviously it holds

$$\Lambda_{\Delta_M}(h_i) \subset \Lambda_{\Delta_M}(f).$$

This result is true for each given counting of the set $\widetilde{H}$. We define

$$\Lambda_{\Delta_M}(\widetilde{H}) := \bigcup_{h \in \widetilde{H}} \Lambda_{\Delta_M}(h)$$

and get

Lemma 6

$$\Lambda_{\Delta_M}(\widetilde{H}) \subset \Lambda_{\Delta_M}(f)$$

The lemmas show that there is a gap between $\Lambda_{\Delta_M}(H)$ and $\Lambda_{\Delta_M}(\widetilde{H})$, if $\widetilde{H}$ includes not computable monotone mappings, which are lower bounds for all $h \in H$. We can close this gap by extending H by the constant functions as the following lemma shows.

Lemma 7. *To each monotone unbounded mapping $f : \mathcal{N} \to \mathcal{N}$ there exists an upward approximation by computable monotone and bounded functions*

$$a^f_{n_0}(n) := f(n) \quad \textit{for} \quad n < n_0 \quad \textit{and} \quad a^f_{n_0}(n) := f(n_0) \quad \textit{for} \quad n \geq n_0$$

such that

$$\Lambda_{\Delta_M}(f) = \bigcap_{n_0 \in \mathcal{N}} \Lambda_{\Delta_M}(a^f_{n_0})$$

The proof the lemma is obvious.

The lemma shows that we may restrict to monotone and computational mappings.

3.3 Closure Properties of Λ_{Δ_M}

We are interested in sets $\mathcal{H} \subset H$ such that $\Lambda_{\Delta_M}(\mathcal{H})$ has properties which are consistent with our intuition.

If $\Lambda_{\Delta_M}(\mathcal{H})$ is the set of algorithmically well approximable sequences then it should follow for $y, z \in \Lambda_{\Delta_M}(\mathcal{H})$ that $x \in \Lambda_{\Delta_M}(\mathcal{H})$ for the operations

$$x := y \vee z, \quad x := y \wedge z, \quad x := \overline{y}, \quad x := y \oplus z,$$

where $\oplus$ means the addition $(mod 2)$. In other words $(\Lambda_{\Delta_M}(H), \vee, \wedge, \bar{\cdot})$ should be a boolean algebra.

It seems reasonable to look for a even stronger closure property: For $y \in \Lambda_{\Delta_M}(\mathcal{H})$ and each program p, which is able to use y as parameter in a reasonably restricted way it should follow $x := M(p; y) \in \Lambda_{\Delta_M}(\mathcal{H})$. This should hold not only for one but for each finite set of parameters. For shortness we discuss only the case of two parameters $y, z \in \Lambda_{\Delta_M}(\mathcal{H})$.

We extend our machine by two parameter tapes y, z, which are only readable and we allow for the reading heads only moves from left to right. We restrict these moves by the following condition. Let $r_x(t)$ the position of the printing head on the output tape of M at time t and $r_y(t), r_z(t)$ the position of the reading heads on y and z. We bound the possible moves of the reading heads on y and z as described by the relation

$$r_y(t), r_z(t) \leq r_x(t),$$

which does not allow the reading heads on the parameter tapes to move faster to the right as the writing head on the printing tape x.

If the extension of M by the parameter tapes y, z with program q under these conditions computes the output x we write

$$x := M(q; y, z).$$

The idea is to approximate x by use of programs p_y^n and p_z^n, which compute approximations of y and z, and the program q. We define

$$\mathcal{H} := \{h \in H : n + 1 - h(n) = o(n)\}.$$

$x \in \Lambda_M(\mathcal{H})$ is equivalent to the condition: There exists a computable function g(n)=o(n) such that

$$\Delta_M(x, 2^{-n}) < g(n) \quad \text{for all} \quad n \in \mathcal{N}$$

For $y, z \in \Lambda_M(\mathcal{H})$ there exist $h_y, h_z \in \mathcal{H}$ such that for $g_y(n) := n + 1 - h_y(n)$ and $g_z(n) := n + 1 - h_z(n)$ there exist programs p_y^n, p_z^n with

$$|p_y^n| \geq g_y(n) = o(n), \quad d(M(p_y^n), y) < 2^{-n},$$

and

$$|p_z^n| \geq g_z(n) = o(n), \quad d(M(p_z^n), z) < 2^{-n}.$$

We use the programs q, p_y^n and p_z^n as procedures for a program p, which simulates the behavior of q to compute an approximation of x with the precision 2^{-n}. This can be done in the following way: p calls the program p_y^n, writes the results of the program on its computing tape, counts its write commands and stops it after n steps. Then it calls p_z and proceeds as before in the case of p_y. Having finished this process it calls q to use both arrays virtually extended by 0^∞ instead of the tapes y, z to compute a sequence $\widetilde{x}$. We write for the result of this procedure

$$\widetilde{x} := M(p(q, p_y^n, p_z^n)).$$

It approximates $x := M(q : y, z)$ with precision 2^{-n}.

We are able to realize this procedure technique with a program p^n with a length $o(n)$. It depends on n only that far as it has to count the mentioned n steps. So it is sufficient to show

$$|q| + |p_y^n| + |p_z^n| = o(n).$$

$|q|$ is constant and our assumption about y, z guarantees the existence of $h \in \mathcal{H}$ such that $g(n) := n + 1 - h(n)$ is an upper bound for $|p_y^n|$ and $|p_z^n|$. So it follows that there exists $h_1 \in \mathcal{H}$ such that $|p(q, p_y^n, p_z^n)| < n + 1 - h_1(n)$. This proves the following

Theorem 2

$$\Lambda_{\Delta_M}(\mathcal{H}) :=$$
$$\{x \in X^\infty : \exists_{h\in\mathcal{H}} \exists_{n_0\in\mathcal{N}} (n + 1 - \Delta_M(x, 2^{-n}) \geq h(n) \quad \textit{for all} \quad n > n_0)\}$$

is closed under the operation $x := M(p; y, z)$ under the assumption of no preview on y and z. The closure under the boolean operations $y \vee z, y \wedge z, y \oplus z, \bar{y}$ are special cases of the first statement. The Operation $\oplus$ means the addition mod 2.

It follows that $(\Lambda_{\Delta_M}(\mathcal{H}), \vee, \wedge, \bar{\cdot})$ is an infinite boolean algebra and $(\Lambda_{\Delta_M}(\mathcal{H}), \oplus)$ is an infinite abelian subgroup of the abelian group $(X^\infty, \oplus)$. We are interested in the quotient group $X^\infty / \Lambda_{\Delta_M}(\mathcal{H})$ because the elements $y, z \in X^\infty, y \equiv z (mod\, \Lambda_{\Delta_M}(\mathcal{H}))$ are related under the aspect of the efficient computational approximation. If we consider y to be random then we may consider z as random, too.

Before we switch to the discussion of the random sequences we clarify the size of the set $\Lambda_{\Delta_M}(\mathcal{H})$ and that a restriction concerning the size of a preview is necessary.

There exists a simple algorithm to map X^∞ injectively to $\Lambda_{\Delta_M}(\mathcal{H})$.

Let $x \in X^\infty$ be a given sequence then we define $\delta : X^\infty \to \Lambda_{\Delta_M}(\mathcal{H})$ as follows: We first define for $i \in \mathcal{N}$ and $x_i \in \{0, 1\}$

$$\bar{\delta}_i(x_i) := x_i^{i+1}.$$

Inductively we define under the assumption that we have defined $y[n]$

$$y[n + 1] := y[n] \cdot 1 \cdot \bar{\delta}(n + 1, x_{n+1}) \quad \text{for} \quad x_{n+1} = x_n = 0,$$
$$y[n + 1] := y[n] \cdot 0 \cdot \bar{\delta}(n + 1, x_{n+1}) \quad \text{for} \quad x_{n+1} = x_n = 1,$$

$$y[n+1] := y[n] \cdot 10 \cdot \bar{\delta}(n+1, x_{n+1}) \quad \text{for} \quad x_n = 0, x_{n+1} = 1,$$
$$y[n+1] := y[n] \cdot 01 \cdot \bar{\delta}(n+1, x_{n+1}) \quad \text{for} \quad x_n = 1, x_{n+1} = 0.$$

This defines δ uniquely. It is obvious, that δ is injective.

We give an example: For $x[6] := 110100$ we get

$$11 \cdot 0 \cdot 111 \cdot 01 \cdot 0000 \cdot 10 \cdot 11111 \cdot 01 \cdot 000000 \cdot 1 \cdot 0000000$$

This sequence of length 35 is uniquely defined by the original sequence of length 6 and a short program p, which is the same for all n. In general we have a description of length $n + |p| = o(n^2)$ for a sequence of a length $\geq n^2$. It follows $\delta(x) \in \Lambda_{\Delta_M}(\mathcal{H})$. The cardinality of X^∞ and $\Lambda_{\Delta_M}(\mathcal{H})$ are equal. The not total inverse mapping δ^{-1} is computable, but it needs a preview on y of size $r(n) = O(n^2)$.

Observation: We are able by application of a very simple partial algorithm on $\Lambda_{\Delta_M}(\mathcal{H})$ by compressing sequences to generate the whole set X^∞.

3.4 Random Sequences

We define for $c \in \mathcal{N}$ motivated by Lemma 7 and following the idea of Kolmogorov

$$\Lambda_{\Delta_M}(c) := \{x \in X^\infty : \exists_{n_0 \in \mathcal{N}}(n + 1 - \Delta_M(x, 2^{-n}) \geq c \quad \text{for} \quad n > n_0)\}.$$

It follows in the notation used in the proof of Lemma 4 and on base of the same arguments

$$\mu(\Lambda_{\Delta_M})(c) < \frac{2}{c}.$$

We define

$$\Lambda_{\Delta_M}(\mathcal{N}) := \bigcap_{c \in \mathcal{N}} \Lambda_{\Delta_M}(c)$$

and get

$$\mu(\Lambda_{\Delta_M}(\mathcal{N})) = 0.$$

It follows for $K_{\Delta_M}(\mathcal{N}) := X^\infty - \Lambda_{\Delta_M}(\mathcal{N})$

$$\mu(K_{\Delta_M}(\mathcal{N})) = 1.$$

$K_{\Delta_M}(\mathcal{N})$ is the set of all sequences $x \in X^\infty$, for which exists a constant $c \in \mathcal{N}$ such that infinitely often $n + 1 - \Delta_M(x, 2^{-n}) < c$ holds.

The question is if we should consider the sequences $x \in K_{\Delta_M}(\mathcal{N})$ as random sequences. The following sections will give answers to this question. Our first question is how do the sequences behave globally over $\mathcal{N}$. This behavior is described by the function

$$f_x(n) := \max\{n + 1 - \Delta_M(x, 2^{-i}) : i \leq n\} \quad \text{for} \quad x \in K_{\Delta_M}(\mathcal{N}).$$

This function is not computable and we do not need to compute this function. An assumption about the behavior of this function has as consequence the existence

of certain programs. In this sense we will use the function. We study our problem by applying our machines of type $\widetilde{M}$.

Let us assume that there exists $c_{n_0} \in \mathcal{N}$ such that

$$n + 1 - \Delta_M(n, 2^{-n}) < c_{n_0}$$

infinitely often and that $f_x(n)$ is unbounded. Then there exists for each $c \in \mathcal{N}$ a minimal $n_c \in \mathcal{N}$ such that $f_x(n_c) \geq c$. We define a program $\widetilde{p}^k := p_1(p_2, p_3^k)$ for $\widetilde{M}$ as follows: Let be $p_2 := p$, where $M(p)$ approximates x such that

$$n_c + 1 - \Delta_M(x, 2^{-n_c}) = f_x(n_c)$$

holds. The program $\widetilde{p}^k$ depends on n_c, too, but we construct for each n_c an infinite sequence of programs $\widetilde{p}^k$. The dependence on k comes in by the print program. We define the print programs by $p_3 := \mathbf{print}x[n_c : n_c + k]$ for $k = 2, 3, 4 \ldots$. Remember that the print programm generates the infinite sequence $x[n_c : n_c + k] \cdot 0^\infty$. But if we use n_c as information for p_1, then our proof will not work. So we try to use $f_x(n_c)$ as information for p_1 to stop p_2 and to call p_3^k. The final definition will be given later.

The program p_1 first calls p_2 and counts the number n_t of moves of the printing head on the output tape depending on the number t of computing steps. It counts additionally the moves of the reading head of tape T_2 and computes the maximum $\max(t)$ of its positions after time t. If $\max(t) - n(t) = f_x(n_c)$, then p_1 stops the program p_2 and starts the program $p_3^k := \mathbf{print}x[\max(t) + 1 : \max(t) + k]$.

It may be that $\max(t) < n_c$ but this does not matter because for $k_0 := n_c - \max(t)$, $\widetilde{M}(\widetilde{p}^{k_0})$ approximates the sequence x with the same precision as $M(p)$ does. So instead of the program $p_2 := p$ we may choose the program $p_2 := p[1 : \max(t)]$ because the rest of the program p will not be used in $\widetilde{p}^k$. This means that we may assume $\max(t) = |p_2|$ and $m(t) = n_c$.

p_1 does not depend on k. So we have

$$\begin{aligned} |p_1| &= \widetilde{c} + \log(f_x(n_c)) \\ f_x(n_c) &= n_c + 1 - |p_2| \\ n &= n_c + k \end{aligned}$$

where $\widetilde{c}$ is independent from k and c. It follows

$$\begin{aligned} n + 1 - \Delta_{\widetilde{M}}(x, 2^{-(n_c+k)}) &\geq n + 1 - |\widetilde{p}^k| \\ &= n + 1 - (|p_1^k| + |p_2| + |p_3^k|) \\ &= n_c + k + 1 - (\widetilde{c} + \log(f_x(n_c)) + |p_2| + k + 1) \\ &= n_c + k + 1 - \widetilde{c} - \log(f_x(n_c)) - |p_2| - k - 1 \\ &= n_c + 1 - |p_2| - 1 - \widetilde{c} - \log(f_x(n_c)) \\ &= f_x(n_c) - \log(f_x(n_c)) - (\widetilde{c} + 1) \end{aligned}$$

For $c \to \infty$ it follows

$$n + 1 - \Delta_{\widetilde{M}}(x, 2^{-n}) \to \infty.$$

This contradicts our assumption: There exists a constant c_0 such that $n+1-\Delta_{\widetilde{M}}(x,2^{-n}) < c$ infinitely often and it holds

$$\Delta_M(x,2^{-n}) \geq \Delta_{\widetilde{M}}(x,2^{-n}) - \bar{c}.$$

$\bar{c}$ is a constant defined by the length of a main program p_1 of $\widetilde{M}$ that calls a best approximation program p of M as $p_2 := p$. This program p_1 does not depend on n. It follows

Theorem 3. *For each $x \in K_{\Delta_M}$ there exists a constant c such that*

$$n+1-\Delta_M(x,2^{-n}) < c.$$

for all $n \in \mathcal{N}$.

3.5 Other Classes of Random Sequences

We will discuss here a weaker condition for randomness. The class is of interest because we are able to prove that the sequences of this class are Bernoulli sequences. And the class is invariant under some restricted blind selections of subsequences by computers. These properties can be proved without any assumption concerning the existence of the limit behavior of mean values related to x.

We define

$$K^{o(n)}_{\Delta_M} := \{x \in X^\infty : n+1-\Delta_M(x,2^{-n}) = o(n)\}$$

We prove first some elementary relations between the two sets of random sequences we are interested in.

Lemma 8

$$K^{o(n)}_{\Delta_M} = K^{0(n)}_{\Delta_M} \oplus \Lambda_{\Delta_M}(\mathcal{H})$$

Proof: From $0^\infty \in \Lambda_{\Delta_m}(\mathcal{H})$ it follows

$$K^{o(n)}_{\Delta_M} \subset K^{0(n)}_{\Delta_M} \oplus \Lambda_{\Delta_M}(\mathcal{H})$$

For the proof of the inclusion in the opposite direction we choose $x = y \oplus z$ with $x \in K^{o(n)}_{\Delta_M}$ and $z \in \Lambda_{\Delta_M}(\mathcal{H})$. It follows $x \oplus z = y$. Using a variant of the concept of the machine $\widetilde{M}$ to compute the approximation of y on base of minimal programs p_2 and p_3 to approximate x respective z with $c := |p_1|$ we get

$$n+1-\Delta_M(y,2^{-n}) \geq n+1-\Delta_M(x,2^{-n}) - \Delta_M(z,2^{-n}) - c.$$

We divide the inequality by n and apply our assumptions about y and z. So it follows for $n \to \infty$

$$n+1-\Delta_M(x,2^{-n}) = o(n)$$

this means $x \in K^{o(n)}_{\Delta_M}$ as claimed by the lemma.

It follows

$$K_{\Delta_M} \subset K^{o(n)}_{\Delta_M} = K^{o(n)}_{\Delta_M} \oplus \Lambda_{\Delta_M}(\mathcal{H})$$

and

Lemma 9

$$(K_{\Delta_M}(\mathcal{N}) \oplus \Lambda_{\Delta_M}(\mathcal{H})) \subset K^{o(n)}_{\Delta_M}$$

It is open if the lemma remains true if we substitute $\subset$ by $=$.

Bernoulli Sequences. In a first step we prove our results under some assumptions about the limit behavior about $x \in K^o_{\Delta_M}$. To formulate these conditions we define a characteristic function $\chi : A \times A \to \{0,1\}$

$$\chi(v,w) = 1 \Leftrightarrow v = w.$$

and $x[l:m] := x_l \cdot \ldots \cdot x_m$ for $l < m$.

Theorem 4. *Under the assumption of the existence of the following limits we define*

$$p_w := \lim_{k\to\infty} \frac{\sum_{i=0}^{k-1} \chi(w, x[i\cdot n+1 : (i+1)\cdot n])}{k} \quad \textit{for} \quad w \in A^n.$$

We claim

$$x \in K^{o(n)}_{\Delta_M} \Rightarrow p_w = \frac{1}{2^n}$$

for all n and all $w \in A := X^n$

Proof: Let be $k, n \in \mathcal{N}$ and $A := X^n$. We consider the prefixes $x[n\cdot k] \in A^k$ of x and define for $w \in A$

$$n_w := \sum_{i=0}^{k-1} \chi(w, x[i\cdot n+1 : (i+1)\cdot n])$$

and

$$p_{n,k}(w) := \frac{n_w}{k}$$

It follows

$$\sum_{w\in A} p_{n,k}(w) = 1 \quad \text{and} \quad \lim_{k\to\infty} p_{n,k}(w) = p_w.$$

We use the well known construction of minimal prefix free codes based on the Kraft inequality in the special case

$$c : A^* \to \{0,1\}^*.$$

We define $i : A \to \mathcal{N}$ uniquely by

$$-\log p_{n,k}(w) \le i(w) < -\log p_{n,k}(w) + 1.$$

This is equivalent to

$$p_{n,k}(w) \ge 2^{-i(w)} > \frac{1}{2} \cdot p_{n,k}(w).$$

It follows

$$1 = \sum_{w \in A} p_{n,k}(w) \geq \sum_{w \in A} 2^{-i(w)} > \frac{1}{2}.$$

As we know from the coding theorem we can find a prefix free code c such that

$$-\log p_{n,k}(w) \leq |c(w)| = i(w) < -\log p_{n,k}(w) + 1 \quad \text{for} \quad w \in A. \tag{1}$$

Using the identy

$$c(x[n \cdot k]) = \prod_{l=0}^{k-1} c(x[l \cdot n + 1] : (l+1) \cdot n)$$

we get

$$|c(x[n \cdot k])| = \sum_{l=0}^{k-1} |c(x[l \cdot n + 1] : (l+1) \cdot n)| = \sum_{w \in A} |c(w)| \cdot n_w.$$

We define the entropy

$$H(p_{n,k}) := -\sum_{w \in A} p_{n,k}(w) \cdot \log p_{n,k}(w)$$

an get by summing up (1)

$$H(p_{n,k}) \leq \frac{1}{k} \cdot |c(x[n \cdot k])| = \frac{1}{k} \cdot \sum_{w \in A} n_w \cdot |c(w)| \tag{2}$$

$$< -\sum_{w \in A} (\frac{n_w}{k} \log \frac{n_w}{k} - \frac{n_w}{k}) = H(p_{n,k}) + 1 \tag{3}$$

We define $p_n := \lim_{k \to \infty} p_{n,k}$ and get

$$\lim_{k \to \infty} H(p_{n,k}) = H(p_n).$$

From (2) it follows

$$|c(x[n \cdot k])| \leq k \cdot H(p_{n,k})$$

and

$$\lim_{k \to \infty} \frac{|c(x[n \cdot k])|}{k} \leq H(p_n)$$

If $H(p_n) < n$, then it follows $n \cdot k - |c(x[n \cdot k])| \neq o(n \cdot k)$. c depends on k, but it can be computed and the application $c(w)$ is computable. So it follows that $n \cdot k - \Delta_M(x, 2 - n \cdot k) \neq o(n \cdot k)$ and x is not in $K^o_{\Delta_M}$.

This theorem can be generalized:

Theorem 5

$$x \in K^{o(n)}_{\Delta_M} \Rightarrow \{x \quad \textit{is a Bernoulli sequence}\}$$

This means that we do not need the assumption about the existence of the limit.

Proof: If one of the limits we assumed to exist in the theorem above does not exist, then there exist at least two limit points. To each such limit point exists a subsequence $n_1 < n_2 < n_3 < ... \in \mathcal{N}$, for which a unique limit exists. Applying the construction our proof is based on, we get a compression of x by a factor $0 < \alpha \leq 1$. For at least one of the limit points we get a compression by a factor $\alpha < 1$. This contradicts the assumption $x \in K^{o(n)}_{\Delta_M}$ because the oscillations of $n + 1 - \Delta_M(x, 2^{-n})$ are bounded by $o(n)$. End of the proof.

This result can be generalized to

Theorem 6. *Let be $\alpha, \epsilon \in [0, 1]$, $\alpha = H(\epsilon, 1 - \epsilon)$ the entropy and*

$$K^{[\alpha]}_{\Delta_M} := \{x \in X^\infty : \alpha \cdot n - \Delta_M(x, 2^{-n}) = o(n)\}.$$

For

$$x \in K^{[\alpha]}_{\Delta_M}$$

and each sequence

$$x_1 < x_2 < x_3 < \ldots \in \mathcal{N}$$

it holds

$$p_w := \lim_{l \to \infty} \frac{\sum_{i=0}^{k-1} \chi(w, x[i \cdot |w| + 1 : (i + 1) \cdot |w|])}{k_l} = \epsilon^{m_1} \cdot (1 - \epsilon)^{|w| - m_1},$$

where m_1, k_l are defined by

$$m_1 := \sum_{i=1}^{|w|} \chi(1, w_i) \quad \textit{and} \quad k_l := \lfloor \frac{n_l}{|w|} \rfloor$$

Proof: If there exist two different sequences $n_1 < n_2 < n_3 < \ldots$ with two different limit points α_1, α_2 then there exist oscillations of $\Delta_M(x, 2^{-n})$ of the size $|\alpha_1 - \alpha_2|$. This contradicts the restriction $\alpha \cdot n - \Delta_M(x, 2^{-n}) = o(n)$ for these oscillations.

Invariance Properties of K_{Δ_M} and $K^{o(n)}_{\Delta_M}$. A subsequence of a random sequence generated by an algorithm, which blindly selects the elements for the subsequence should be again a random sequence [8]. This should be even the case if the algorithm knows the prefix of the random sequence up to the position before the position i it has to decide "select or not select" x_i. We will prove here this invariance property only for the special case under additional assumptions. In the case of K_{Δ_M} we prove this not for each sequence, but only for sequences of the subset $K_{\Delta_{\overline{M}}} \subset K_{\Delta_M}$. In the case of $K^{o(n)}_{\Delta_M}$ we restrict the selection procedures by a density condition.

Let $i : \mathcal{N} \to \mathcal{N}$ and $j : \mathcal{N} \to \mathcal{N}$ be strictly monotone and computable mappings with the following properties:

$$i(\mathcal{N}) \cup j(\mathcal{N}) = \mathcal{N} \quad \text{and} \quad i(\mathcal{N}) \cap j(\mathcal{N}) = \emptyset,$$

We define

$$i(x) := (x_{i(1)}, x_{i(2)}, x_{i(3)}, \ldots) \quad \text{and} \quad j(x) := (x_{j(1)}, x_{j(2),x_{j(3),\ldots}})$$

for $x \in X^\infty$ and

$$n_1 := \sharp\{k \in \mathcal{N} : i(k) \leq n\} \quad \text{and} \quad n_2 := \sharp\{k \in \mathcal{N} : j(k) \leq n\}.$$

If S is a finite set, then $\sharp S$ means the number of elements in S. It follows

$$n_1 + n_2 = n.$$

Theorem 7. *Under the assumption $x \in K_{\Delta_{\bar{M}}}$, where $K_{\Delta_{\bar{M}}} \subset K_{\Delta_M}$ for a net $\bar{M}$ of three machines it holds*

$$x \in K_{\Delta_M} \Rightarrow i(x), j(x) \in K_{\Delta_M}$$

and under the condition $n_1, n_2 = \Omega(n)$ it holds

$$x \in K^{o(n)}_{\Delta_M} \Rightarrow i(x), j(x) \in K^{o(n)}_{\Delta_M}$$

Proof: We get approximations of the precision 2^{-n} by computing approximations of $y := i(x)$ and $z := j(x)$ of precision 2^{-n_1} and 2^{-n_2}, respectively. Let p_y and p_z be programs, which compute infinite sequences to approximate y and z with precision 2^{-n_1} and 2^{-n_2}, respectively. p_1 and p_2 compute the mappings i and j, respectively. We describe a program p that uses variants of the programs p_1, p_2, p_y, p_z as procedures to compute an approximation $\widetilde{x}$ of x with the precision 2^{-n}.

We substitute the programs p_y, p_z by programs *print*(p_y), *print*(p_z), which we will define later. the program p we define as follows.

$$\begin{array}{l} \textit{repeat infinitely often}\{\ i := 1\ ; \quad j := 1; \quad k := 1 \\ \qquad \textit{if} \quad p_1(i) = k \ \ \textit{then} \ \ \textit{print}(p_y); \quad i := i + 1; \quad k := k + 1 \\ \qquad\qquad\qquad \textit{else} \ \ \textit{print}(p_z) \quad j := j + 1; \quad k := k + 1\} \end{array}$$

We modify p_y as follows.

The call *print*(p_y) starts the program in its last state when it has been stopped.

The program state of p_y after the printing on the output tape will substituted by a stop state.

The procedure *print*(p_z) we define analogously.

We see that the sizes of the programs p, p_1, p_2 are independent from n. The size of the programs p_y, p_z has not been changed. So we get for the size $|P_n|$ of $P_n := p(p_1, p_y, p_2, p_z)$ relative to the universal machine $\overline{M}$ consisting of two components of type M to compute p_y and p_z and a component to compute p, p_1, p_2 the relation

$$\Delta_{\overline{M}}(x, 2^{-n}) \leq |P| = \Delta_M(y, 2^{-n_1}) + \Delta_M(z, 2^{-n_2}) + c_p$$

with $c_p := |p| + |p_1| + |p_2|$. Using the relation $n = n_1 + n_2$ we get

$$n+1-\Delta_M(x,2^{-n}) \geq (n_1+1-\Delta_M(y,2^{-n_1}))+(n_2+1-\Delta_M(z,2^{-n_2}))-(c_p+1)$$

Under the assumption $x \in K_{\Delta_M}$ it follows, that there exists a $c \in \mathcal{N}$ such that $n+1-\Delta_M(x,2^{-n}) < c$ for all n. It follows

$$n_1+1-(\Delta_M(y,2^{n_1}) < c+c_p+1 \quad \text{and} \quad n_2-\Delta_M(z,2^{-n_2}) < c+c_p+1$$

for all n_1 and all n_2. This means that $y, z \in K_{\Delta_M}$ as claimed by the first part of the theorem.

It remains to discuss the case $x \in K^{o(n)}_{\Delta_M}$. In this case it follows

$$n_1 - \Delta_M(y,2^{-n_1}) = o(n) \quad \text{and} \quad n_2 - \Delta_M(z,2^{-n_2}) = o(n)$$

For the proof of the theorem we need $o(n_1)$ and $o(n_2)$, respectively, on the right hand side of our equations. This follows for $n_1, n_2 = \Omega(n)$ as assumed in our theorem.

Lemma 10

$$\mu(K_{\Delta_M} - K_{\Delta_{\bar{M}}}) = 0$$

Proof: Analogously to the proof of $\mu(K_{\Delta_M}) = 1$, one proves $\mu(K_{\Delta_{\overline{M}}}) = 1$. It holds $K_{\Delta_{\bar{M}}} \subset K_{\Delta_M}$ because the machine $\bar{M}$ can simulate M by using the identical program p_x on one of the two submachines controlled by a program p independent from p_x. So the lengths of the two programs on M and $\overline{M}$ differ only by the constant $|p|$. It may be that an even shorter program configuration on $\bar{M}$ exists to compute an approximation of x of the same precision. But it cannot happen that a sequence relative to $\overline{M}$ is random and is not relative to M. The claim of the lemma follows.

4 Concluding Remarks

We have seen that the use of infinite computations and generalizations of the prefix approximation simplifies the theory. In no step we did really use assumptions on computability of the mappings in $h \in H$ or assumptions concerning the complexity of runtime of programs p to define Δ_M. Complexity aspects come in only in connection with the application of procedure technics or in programs to compute approximations on base of the coding theorem, which we applied in the proof of our last theorems. But this complexities are all on a very low level. Complexity hierarchies may play an important role in connection with the definition of Δ_M as C. P. Schnorr has proved, [3, 4]. Hierarchies may be generated too by networks of machines as we considered in a special case. But it is open if $K_{\Delta_M} \neq K_{\Delta_{\overline{M}}}$.

References

1. Kolmogorov, A.N.: Drei Zugänge zur Definition des Begriffs Informationsgehalt. Probl. Peredaci Inform. 1, 3–11 (1965) (in Russian)
2. Martin Löf, P.: The Definition of Random Sequences. Information and Control 8, 602–619 (1966)
3. Schnorr, C.-P.: Zufälligkeit und Wahrscheinlichkeit, eine algorithmische Begründung der Wahrscheinlichkeitstheorie. Lecture Notes in Mathematics, vol. 212, pp. 1–212. Springer, Heidelberg (1971)
4. Schnorr, C.-P.: Eine neue Charakterisierung der Zufälligkeit von Folgen, Habilitationsschrift zur Erlangung der Venia Legendi im Fach Mathematik der Universität des Saarlandes (1969)
5. Chaitin, G.I.: On the Length of Programs to Compute Finite Binary Sequences. J. Assoc. Comp. Machin. 13, 547–569 (1969)
6. Chaitin, G.I.: Algorithmic Information Theory. Cambridge University Press, Cambridge (1987)
7. Calude, C.: Theories of Computational Complexity. North-Holland, Amsterdam (1988)
8. von Mises, R.: Grundlagen der Wahrscheinlichkeitstheorie. Math. Zeitschrift 5, 5–99 (1910)
9. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and its Applications, pp. 1–546. Springer, Heidelberg (1993)
10. Hotz, G., Gamkrelidze, A., Gärtner, T.: Approximation of Arbitary Sequences by Computable Sequences - A new Approach to Chaitin-Kolmogorov-Complexity, 1–18 (unpublished, 2007); Gärtner T., Hotz, G.: Approximation von Folgen durch berechenbare Folgen - Eine neue Variante der Chaitin-Kolmogorov-Komplexität, Technischer Bericht A 01/02, März 2002, Fakultät für Mathematik und Informatik der Universität des Saarlandes, pp. 1–19
11. Chadzelek, T., Hotz, G.: Analytic Machines. Theoretical Computer Science 219, 151–167 (1999)
12. Hotz, G.: Algorithmische Informationstheorie, pp. 1–142. Teubner Texte zur Informatik, B. G. Teubner Verlag (1997)

Author Index