MATHEMATISCHES FORSCHUNGSINSTITUT OBERWOLFACH

Tagungsbericht 46/1972

Algorithmen und Komplexitätstheorie

26.11. bis 30.11.1972

Unter Leitung von Professor C.P.Schnorr (Frankfurt) und Professor A.Schönhage (Tübingen) fand in diesem Jahr erstmals eine Tagung über Algorithmen und Komplexitätstheorie im Mathematischen Forschungsinstitut Oberwolfach statt. Es war gelungen, einen internationalen Kreis von Teilnehmern zu dieser Tagung zu gewinnen. Die sich hieraus ergebende Gelegenheit zu einem eingehenden und fruchtbaren Gedankenaustausch auf diesem Gebiet wurde lebhaft begrüßt.
Die Vorträge beschäftigten sich u.a. mit schnellen Algorithmen, unteren Schranken für die Rechenzeit von Algorithmen (insbesondere von Entscheidungsverfahren in der Logik), abstrakter Komplexitätstheorie rekursiver Funktionen, Programm-Komplexität und Zufälligkeit von Folgen, sowie Programm-Schemata.

Teilnehmer

D.A.Alton, Iowa City
G.Ausiello, Rom
H.Becker, Darmstadt
J.Běcvář, Prag
M.Blum, Berkeley
van Emde Boas, Amsterdam
W.Böge, Heidelberg
H.Brakhage, Kaiserslautern
B.Buchberger, Innsbruck
A.B.Cremers, Karlsruhe
R.Daley, Chicago
M.Fischer, Cambridge
K.Ecker, Bonn
P.Flajolet, Rocquencourt
P.Fuchs, Frankfurt
H.Jürgensen, Kiel
L.Kalmar, Szeged
A.Martelli, Pisa
O.Mayer, Karlsruhe
A.R.Meyer, Cambridge
E.Minicozzi, Neapel
B.Monien, Hamburg
H.Müller, Erlangen
M.Nivat, Paris
M.S.Paterson, Coventry
C.P.Schnorr, Frankfurt
G.Schmidt, München
A.Schönhage, Tübingen
H.Schwichtenberg, Münster
D.Siefkes, Heidelberg
M.Sieveking, Zürich
J.Spieß, Göttingen
J.M.Steyart, Rocquencourt
H.J.Stoß, Konstanz
G.Stumpf, Frankfurt
G.Trautteur, Neapel
K.Weihrauch, Bonn
H.Walter, Darmstadt
R.Weicker, Hamburg
E.Zachos, Zürich

Vortragsauszüge

Alton, D.A.(Iowa City: Embeddability and Diversity of Speed-ups in Computational Complexity

Let Φ be an arbitrary measure of computational complexity. Let $P(\Phi)$ be the collection of all complexity classes of Φ, partially ordered under set inclusion. E.McCreight and A.R. Meyer have proven that every countable partial order can be embedded in $P(\Phi)$. I generalize this result in two ways, first by showing that the run-times which form the bounds on the complexity class can be separated by a total effective operator (Theorem 1) and second by showing that r-speed-upable functions can be responsible for the embedding (Theorem 2). I then show (Theorem 3) that any recursive ordinal can be embedded in $P(\Phi)$ in such a way that any speed-upable function in the largest complexity class in the embedding is also in the smallest. Theorem 1 was proved independently by R. Moll. A result of Prof. Dr. Schnorr shows that Theorem 3 is "best possible" in the sense that any descending sequence of run-times (corresponding to an effort to embed ω^*, the reversal of ω, into $P(\Phi)$) is the complexity sequence of a function.

Ausiello, Giorgio IAC - CNR, Rom: Considerations on the approximation of functions via resource bounded computations

Between different approaches to the problem of relating the complexity and the semantics of a program (such as structured complexity or the complexity properties of translations) a particular attention is deserved by the study of chains of functions defined by bounds on computations. -

The concept of semantics of programs defined in the Dana Scott lattice theoretics approach is formulated on an extensional environment where the basic objects are functions and not programs. The way programs are dealt with is through sequences of functions: we associate to any program a recursive operator ψ and then, via Kleene's theorem, we characterize the function being computed as the least fixed point of ψ and we characterize the program through the sequence of functions $\{\psi^n[\bot]\}$ where $\bot$ is the everywhere undefined function.

Starting from the notions of computational complexity, given any function f, given any program φ_j for f, given any measure of complexity Φ and given one r.e. sequence of total resource bounds $\{g_n\}$ we can define the sequence $\{f_n^j\}$ where

$$f_n^j(x) = \begin{cases} f(x) \text{ if } \Phi_j(x) \leq g_n(x) \\ \text{"error" otherwise} \end{cases}$$

The semantically based sequence and the complexity based sequence can be embedded in Scott's lattice of function and an interesting research field is to find relation between them.

Bečvař, Jiří, Prag: Programs and Complexity measures

The property of a function f(x,y) to be an acceptable Gödel-numbering of all partial recursive (p.r.) functions is not invariant with respect to the change of arguments. This fact, though not yet fully understood, has a connection to two (dual) concepts of reducibility between numberings of p.r.

functions. Results concerning the structure of the corresponding quasi-orderings are presented, especially the theorem (M. Chytil): mutual reducibility between numberings of arbitrary sets of p.r. functions implies their isomorphism. One of the two types of reducibility can be interpreted as a change of the encoding of arguments. The effect of this change in connection with various measures of computational complexity is discussed and a new axiom for composition due to M. Chytil is presented.

Blum, Manuel, Berkeley: Inductive inference

An inductive inference machine is a Turing machine that is designed to accept the graph of a recursive function, bit by bit, and to conjecture algorithms for computing that function. While being fed by the graph of a function f, such a machine may change its mind an infinite number of times, in which case it does not infer f. However, if for each enumeration of the graph of f there is a point in time at which the machine conjectures a correct algorithm for f and never again changes its mind, then we say that the machine can identify f.

Our goal has been to characterize the classes of recursive functions that can be identified by inductive inference machines. Our characterization is in terms of the complexity of the identifiable functions: we have designed a machine M that can identify the class of compressed functions, i.e. all those functions having optimal or near-optimal algorithms. The compressed functions include all real-time computable functions, all (more-or-less) quickly computable functions (such as the primitive recursive functions), and some arbitrarily difficult to compute 0-1 valued functions.

van Emde Boas, P., Amsterdam: Abstract Resource bound Classes

Comparing the properties of complexity and honesty classes yields two differences. Firstly, the hierarchy of honesty classes has at its base not the bad recursive properties shown by the complexity classes; secondly, there exists no honesty procedure to rename honesty classes by a measured set. An abstract framework to explain these differences is developed. Givena measured set of "generalized runtimes" one defines abstract resource bound classes in two ways, called strong and weak classes, depending on the acceptance or rejection of an "infinite runtime". A formalization eliminating this "infinite runtime" is given by the concept an acceptance relation.
By selection of a suitable acceptance relation one can represent complexity classes,honesty classes (typical examples of strong c.q. weak classes) complexity classes modulo sets of exceptional points, and several types of summed complexity classes. The part of complexity theory independent of the first Blum axiom (gap, union, and naming theorem) is developed in this framework; moreover these theorems are generalized in several directions, yielding new applications for complexity classes as well.

Buchberger, Bruno, Innsbruck: Programmänderung zur Exekutionszeit

Wir geben eine rekursionstheoretische Formulierung des Konzepts eines "programmierten Automaten, der zur Exekutionszeit seine Programme ändert" und eines Automaten mit "festen Programmen". Wir zeigen, wie die Zustände eines beliebigen universellen Automaten so "dekomponiert" werden können, daß eine Komponente als ein zur Exekutionszeit unverändertes Programm aufgefaßt werden kann.

Cremers, A.B., Karlsruhe: Komplexität formaler Ausdrücke

Mit Hilfe der für formale Sprachen erklärten Symboliteration werden contextfreie Ausdrücke eingeführt und eine Erweiterung der Theorie formaler Ausdrücke angegeben.
Es werden mehrere Kriterien der "Konstruktionskomplexität" definiert, darunter ein der "star-height" entsprechendes Kriterium für context-freie Ausdrücke. Diese Kriterien messen zunächst die Komplexität formaler Ausdrücke. Durch $K(L) := \min \{K(E) \mid L = \langle E \rangle\}$ definiert man dann die Komplexität einer formalen Sprache L bezüglich eines Kriteriums K.
Ein Kriterium K heißt zusammenhängend über einem Alphabet T, falls zu jeder nichtnegativen ganzen Zahl n eine Sprache L existiert mit K(L) = n.
Alle eingeführten Kriterien werden als zusammenhängend herausgestellt: man erhält jeweils unendliche Hierarchien von Komplexitätsklassen.
Im Anschluß daran werden Entscheidbarkeitsprobleme zur Komplexitätstheorie formaler Ausdrücke erörtert und die Anwendung der Theorie zur Klassifikation von Listenstrukturen erläutert.

Daley, Robert, Chicago: Program Size and Computation Time: A Combined Analysis

The minimal-program complexity of binary sequences with a priori computation time restrictions and the resulting hierarchy is discussed. In contrast to the Blum complexity hierarchy, one can construct sequences of any desired effectively specified "time bounded" minimal-program complexity. Some rather surprising variations of this construction are mentioned. Examples of nonrecursive sequences with both extremely slow growing program size requirements and computation time requirements are presented. Trade-offs between program size and computation time for recursive sequences are also discussed.

Fischer, Michael J., Cambridge: On-line pattern-matching Algorithms

Pattern-matching in strings can be regarded as a kind of "multiplication", allowing a general technique for converting off-line multiplication algorithms to on-line operation to be applied. For the problem of exact pattern match, we obtain a time n log n on-line multitape Turing machine by showing how to implement the Pratt-Knuth method on a Turing machine without time loss and then converting it to on-line operation. We show the more general problem in which the strings are allowed to contain single-symbol "don't care" markers is time-equivalent to "and-or" multiplication, and the latter problem can be reduced to ordinary integer product, yielding a time $n(\log n)^3(\log \log n)$ on-line Turing machine method.

Fuchs, Peter, Frankfurt: Some new results on the complexity approach to random sequences

Martin-Löf proved: $\forall f \in \mathbb{R}_1^1$: $[\sum_n 2^{-f(n)} = \infty] \Rightarrow [\exists g \in \mathbb{R}_1^1$: $\forall z \in X^\infty$: $\overline{\lim_n}(n-K_{A,\mathfrak{A}}(z(n),g(n))-f(n)) > 0]$. Here $\mathbb{R}_1^1$ denotes the class of recursive functions λ: $N \to N$ and $K_{A\mathfrak{A}}$ denotes the effective program-complexity as introduced by Schnorr.

We are going to characterize both Martin-Löf and Schnorr randomness in terms of effective program-complexity.

Let z be an infinite binary sequence (i.e.: $z \in X^\infty$).

z is not Martin-Löf random iff

$\exists g,f \in \mathbb{R}_1^1$: $\sum_n 2^{-f(n)} < \infty \wedge \overline{\lim_n}(n-K_{A,\mathfrak{A}}(z(n),h(n))-f(n)) > 0$

z is not Schnorr random iff

$\exists g,f \in \mathbb{R}_1^1$: $\sum_n 2^{-f(n)}$ conv. constr. $\wedge\ \overline{\lim_n}(n-K_{A,\mathfrak{A}}(z(n),g(n))-f(n)) > 0$

($\sum_n 2^{-f(n)}$ converges constructively iff there is a regulator of convergence $r \in \mathbb{R}_1^1$ such that: $\forall i$: $\sum_{n \geq r(i)} 2^{-f(n)} \leq 2^{-i}$).

This work is co-authored by C.P. Schnorr.

Meyer, Albert R., Cambridge: Inherently difficult decidable Problems in logic and automata theory

Just as one proves that many familiar mathematical problems are recursively undecidable, it is possible to prove that many <u>decidable</u> problems are inherently hard to decide. The method is a reducibility argument in which one efficiently codes Turing machines which run for large amounts of time, but we do eventually halt, into a problem of interest.

Let A be a set of words and $t: \mathbb{N} \to \mathbb{N}$ be a function.

Def. Time (A) $\leq$ t almost everywhere $\Leftrightarrow$ There exists a Turing machine which accepts the set A and which halts in $\leq t(\text{length}(x))$

steps for all but finitely many input words x.

Theorem Let A = any of the sets numbered below; then

$$(\exists c > 0)[\mathrm{Time}\ (A) \leq \lambda n.[2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}\Big\}c\cdot n] \text{ almost everywhere}],$$

but it is false that

$$(\forall \epsilon > 0)[\mathrm{Time}\ (A) \leq \lambda n.[2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}}\Big\}\epsilon.\log_2 n \text{ almost everywhere}].$$

The set A may be chosen to be

the valid sentences of the first order theory of

(1) linear order, (2) a single monadic function,

(3) discrete order with a single monadic predicate,

(4) dense order with a single monadic predicate,

the true sentences of the

(5) weak monadic second-order theory of the structure $\langle N, \text{successor} \rangle$,

(6) Presburger arithmetic with the additional predicate $\lambda x.y.$[x is a power of two and x divides y],

(7) the first order theory of finite binary words with the two successor functions $S_o(x) = xo, S(x) = x1$ and the predicate $\lambda x.y$[x is a prefix of y]

(8) theory of pure finite types

and finally

(9) the set of star-free expressions (familiar in automata theory) which define the empty language.

One can also obtain similar results, but with different time bounds for many other decidable problems.

Results (2) and (8) were discovered jointly with M.J. Fischer; results (9) is due to L. Stockmeyer.

Monien, B., Hamburg: Beziehungen zwischen zeitbeschränkten Turingmaschinen und Kellerautomaten

S.A. Cook bewies, daß die Klasse der Sprachen, die von deterministischen Turingmaschinen in Polynomzeit erkennbar sind, und die Klasse der Sprachen, die von Mehrkopf-Zweiweg-Kellerautomaten erkennbar sind, übereinstimmen.
Es wird der folgende Satz bewiesen:
Jede deterministische Turingmaschine, die mit der Zeitbeschränkung $T(u) = C\ n^{p}$ arbeitet, kann von einem deterministischen p-Kopf Zweiweg Kellerautomaten, der einen zusätzlichen "Zähler der Länge $(\log_2 n)$" besitzt, simuliert werden.

Müller, Horst, Erlangen: A recursive program scheme to which there exists no equivalent non-recursive program scheme

Scott's notion of (simple) program schemes is extended to recursive program schemes which allow instructions of typ L: do P go to L';(where L,L' are Labels, P is a procedure-name). The semantics of such program schemes is given by a machine which can interpret them. The recursive program scheme π

P: L_o: if B then goto L_1 else goto L_2;

L_1: do F_o goto L_3;

L_3: do P goto L_4;

L_4: do F_1 goto L_2;

L_2: halt;

computes on the machine $\mathfrak{M}$ with storageset $\{0,1,2,\ldots\}$, which interpretes B by "Test on 0" ("true for $x \neq 0$"), F_o by $\mathfrak{M}_{F_o} = \langle x \rightarrow x \dot{-} 1\rangle$, F_1 by $\mathfrak{M}_{F_1} = \langle x \rightarrow 2x+1\rangle$, the function $\mathfrak{M}_{\pi} = \langle x \rightarrow 2^{x}-1\rangle$.

There exist no simple program scheme equivalent to π, because all functions $\mathfrak{M}_\pi$ for simple program schemes π are bounded by a linear function $\langle x \to k(x+1)\rangle$ (for some k).

Nivat, Maurice, Paris: Some problems in the theory of program schemes

Some results established by various authors concerning the translatability of certain classes of recursive monadic schemes into flowcharts have been given intuitive rather than formal proofs. The aim of this talk is to give a formalism in which such proofs can be written easily and, one hopes, some new results obtained.

N.B. This work is coauthored by Melle I.Guersarian, CNRS Paris.

Paterson, M.S., Coventry: Complexity of On-line Multiplication of Integers and Reals

A function on strings is computed "on-line" in time complexity T if, for all n,at time T(n) the n^{th} output symbol has been given but the $(n+1)^{st}$ input symbol has not yet been read. Integers (or p-adic integers) may be multiplied on-line if the digits are presented from right to left. The principal results presented are lower bounds on the time complexity of on-line multiplication. For the two classes of machines, "uniform" and "polynomially-bounded", defined by Cook and Anderaa the lower bound has been improved to O(n log n/log log n), while for multi-tape Turing machines and for "oblivious" algorithms on a very general class of machines the bound is O (n log n). (Oblivious

algorithms are such that the sequence of storage locations accessed is independent of the digits of the inputs].
The "oblivious" bound is shown to be optimal for the class.

Schnorr, C.P., Frankfurt: Analysis of Turing machine computations

We use the Kolmogorov concept of program complexity (information content) of finite binary sequences and the idea of crossing sequences to derive lower bounds on the product of space-time requirements for certain decision problems on Multitape Turingmachines. Hereby we count the total space requirements of the working tapes only, i.e. we do not count the space requirements of the input tape. Using the concept of program complexity we generalise and trivialise some applications of the idea of crossing sequences which are due to Cobham. The product of space-time requirements deciding a certain relation R(u,v) on subwords u,v is bounded from below by the square of $\min_{w \in X^*}$ (Com(u,w) | R(u,w) holds) and a constant factor which depends on the machine only.
Hereby Com(u,w) is the "common information content" of the words u and w which is defined by

Com(u,w) = K(u) + K(w) - K(uw) where K is the Kolmogorov program complexity.

Schönhage, A., Tübingen: Unitäre Transformationen großer Matrizen

Nach V. Strassen kostet die Multiplikation von (N,N)-Matrizen höchstens $O(N^{\beta})$ viele Grundoperationen, wobei $2^{\beta} = 7$, also $\beta < 3$. Hier werden entsprechend unitäre Transformationen komplexer Matrizen behandelt und folgende Ergebnisse erzielt:
Wenn die Multiplikation beliebiger (K,K)-Matrizen in $O(K^{\alpha})$ Schritten möglich ist $(2 < \alpha < 3)$, dann kostet auch die QR-Zerlegung (Orthogonalisierung) beliebiger (N,N)-Matrizen höchstens $O(N^{\alpha})$, die unitäre Tridiagonalisierung hermitescher Matrizen höchstens $O(N^{\alpha})$ und die unitäre Transformation auf obere Hessenbergform höchsten $O(N^{\alpha} \lg N)$ Schritte.

Schwichtenberg, Helmut, Münster: Eine Klassifikation der ϵ_0-rekursiven Funktionen

Eine Funktion heißt ϵ_0-rekursiv, wenn sie aus den Ausgangsfunktionen $0, S, I_n^i$ mit Einsetzen und (geschachtelten) $\omega\alpha$-Rekursionen ($\alpha < \epsilon_0$ beliebig) definierbar ist. Jedem Faden im Definitionsbaum eines ϵ_0-rekursiven Funktion wird als Rekursionszahl $\alpha_1 + \ldots + \alpha_n$ zugeordnet, falls die auf dem Faden liegenden Rekursionen nacheinander die Ordnungstypen $\omega\alpha_1, \ldots, \omega\alpha_n$ besitzen. Die Rekursionszahl einer Definition einer ϵ_0-rekursiven Funktion sei das Maximum der Rekursionszahlen aller Fäden im Definitionsbaum. Sei $\mathcal{R}_\alpha := \{f \mid f$ definierbar mit Rekursionszahl $\leq \alpha\}$. Es werden mehrere Charakterisierungen der $\mathcal{R}_\alpha$ angegeben, mit dem Ziel, die enge Beziehung zwischen den Funktionen in $\mathcal{R}_\alpha$ und der Ordinalzahl α zu verdeutlichen; s. Zeitschr. math. Logik Grundl. Math. Bd. 17 (1971), pp. 61-74.-

Eine dort nicht angegebene Charakterisierung: Unendliche Terme endlichen Typs seien definiert wie in Tait, "Infinitely long terms of transfinite type" (in: Formal systems and recursive functions, Amsterdam 1965, pp. 176 - 185).
Man kann endliche Bezeichnungen für unendliche Terme einführen, indem man zur Bezeichnung einer Folge (u.a.) verwendet (1) einen pr. rek. Index einer die Folgenglieder aufzählenden Funktion und (2) eine Standard-Bezeichnung einer Ordinalzahl $< \epsilon_0$ als Tiefenschranke. Ein Term t heiße prädikativ, wenn kein Teilterm eine höhere Tiefenstufe besitzt als t selbst. Sei $\mathcal{P}_\alpha := \{f \mid f$ def.bar durch eine prädikative Termbezeichnung mit Tiefenschranke $< \omega(\alpha+1)\}$. Dann gilt $\mathcal{R}_\alpha = \mathcal{P}_\alpha$ für $\omega \leq \alpha < \epsilon_0$.

Schwichtenberg, Helmut, Münster: Bemerkungen über elementare und subelementare Funktionen

---

1. Es gibt eine subelementare Funktion U (d.h. $U \in \mathcal{E}^2$ nach Grzegorczyk), so daß jede rekursive Funktion f sich darstellen läßt in der Form $f(x) = U(p,x,y)$ für alle $y \geq Q(x,s_f(x))$ mit einem Polynom Q; s_f ist eine Schrittzahlfunktion von f bzgl. Registermaschinen (s. Rödding: "Klassen rekursiver Funktionen" in: Lecture Notes in Math. Bd.70)

2. Es werden verschiedene naheliegende Definitionen von subelementaren Funktionen über binären Bäumen angegeben und die entstehenden Funktionenklassen miteinander verglichen. Die Verhältnisse sind komplizierter als bei den natürlichen Zahlen. (s. E. Cohors-Fresenborg, Dissertation Münster 1971)

3. $\mathcal{R}^n$ sei die Klasse der primitiv rekursiven Funktionen, die mit $\leq n$ übereinanderliegenden primitiven Rekursionen definierbar sind. $\mathcal{E}^n$ sei die n-te Grzegorczyk-Klasse. Das bekannte Resultat $\mathcal{R}^n = \mathcal{E}^{n+1}$ für $n \geq 3$ (s. Archiv math. Logik Grundl. Bd. 12 (1969), pp. 85 - 97) läßt sich auf $n = 2$ verschärfen (Helmut Müller).

Stoß, H.-J., Konstanz: Der Mindestaufwand beim Permutieren von Daten

Problem 1: Given a tape with n entries

$$* * * * * * a_1 a_2 a_3 \ldots\ldots a_n * * * * * *$$

and given a permutation $\Pi \in \mathcal{S}_n$

Question: Which is the minimum number $L(\Pi)$ of steps which are needed to rearrange this data to

$$* * * * * * a_{\Pi(1)} a_{\Pi(2)} \cdots a_{\Pi(n)} * * * * * *$$

under the assumption that there are k (≥ 2) heads on the tape and that in one step you can permute the data under the heads and shift all heads at most one square?

Problem 2: Let a storage be given which contains its information partitioned into blocks of atmost r elements.

Question: Which is the minimum number L(E,F) of steps which are needed to rearrange a given set E of blocks to a new set F of blocks, if in one step you can rearrange the data contained in k(≥ 2)blocks, only?

We introduce the conception of complexity measure and show that each compl. meas. Ψ is a lower bound for L. Solving problem 2 we define a concrete Ψ from the entropy of probability measures induced by the numbers $\#(e \cap f)(e \in E, f \in F)$.

Solving problem 2 we divide the strings $a_1 a_2 a_3 \ldots a_n$ and $a_{\Pi(1)} a_{\Pi(2)} \cdots a_{\Pi(n)}$ into blocks. Now we can apply arguments similar to these used solving problem 2.

Stumpf, Gerd, Frankfurt: A characterization of complexity sequences

Let $(\varphi_i)_{i \in N}$ be a Gödel numbering of the partial recursive functions and $(\Phi_i)_{i \in N}$ a time measure for $(\varphi_i)_{i \in N}$ (with respect to Turing machines). A complexity sequence for a partial recursive function f is a sequence of functions $(p_i)_{i \in N}$ such that

$\forall i: D(p_i) = D(f)$

$\forall i: (\varphi_i = f): \exists j: \Phi_i \geq p_j$ (almost everywhere)

$\forall j: \exists k(\varphi_k = f): p_j \geq \Phi_k$ (almost everywhere)

Let φ_μ be a partial recursive function ($< \;\; >: N \times N \to N$ recursive, 1-1, onto) such that

(a) $\forall n: \forall x: \varphi_\mu \langle n,x \rangle \geq \lceil {}^2\log(x+1) \rceil$

(b) $\exists c: \forall n: \forall x: \Phi_\mu \langle n,x \rangle \leq c\, \varphi_\mu \langle n,x \rangle$

(c) $\forall n: D(\varphi_\mu \langle n, \cdot \rangle) = D(\varphi_\mu \langle o, \cdot \rangle)$

(d) $\forall n: \lim\limits_{x} \dfrac{\varphi_\mu \langle n,x \rangle}{\varphi_\mu \langle n+1,x \rangle} = \infty$

then $(\varphi_\mu \langle n, \cdot \rangle)_{n \in N}$ is a complexity sequence for a 0-1 valued partial recursive function.

This work is co-authored by C.P. Schnorr.

P. Fuchs (Frankfurt)