MATHEMATISCHES FORSCHUNGSINSTITUT OBERWOLFACH

T a g u n g s b e r i c h t 43 /1974

Algorithmen und Komplexitätstheorie

27.10. bis 2.11.1974

Unter Leitung der Herren C. P. Schnorr (Frankfurt), A. Schönhage (Tübingen) und V. Straßen (Zürich) fand nun die zweite Fachtagung über Algorithmen und Komplexitätstheorie statt. Dank der neuen Räumlichkeiten war es möglich, eine größere Zahl von Interessenten einzuladen. Von den insgesamt 41 Teilnehmern kamen 12 aus dem Europäischen Ausland und 7 von Übersee.

Das 26 Vorträge umfassende Programm bot in breit gefächerter Auswahl reichlich Gelegenheit, sich über aktuelle Fragen auf diesem Gebiet zu informieren. Algebraische Komplexitätsprobleme bildeten einen thematischen Schwerpunkt, im einzelnen Polynomberechnungen, algebraische Zahlen und Matrix-Algorithmen betreffend. Weiter wurden Komplexitätsfragen der Analysis wie auch der Logik behandelt. Vergleichende Studien über verschiedene Maschinenmodelle stellten unter dem Gesichtspunkt der Komplexität Querverbindungen zur Automatentheorie und zur Theorie der formalen Sprachen her. Außerdem wurde über einige neue Algorithmen für Such- und Sortierprobleme berichtet.

Neben dem Vortragsprogramm wurden die vielfältigen Möglichkeiten zur Diskussion genutzt, zum Teil in kleinen informellen Sitzungen. Aus dem Rahmen solcher Aktivitäten ist insbesondere über eine bemerkenswerte Wette zu berichten, die am Ende dieses Berichts wiedergegeben wird.

Teilnehmer

O. Axelsson, Göteborg
J. Becvár, Prag
A. Borodin, Toronto
H. Brakhage, Kaiserslautern
B. Brosowski, Göttingen
P. Brucker, Regensburg
G. E. Collins, Kaiserslautern
St. A. Cook, Toronto
E. Engeler, Zürich
A. R. Meyer, Cambridge (Mass.)
B. Monien, Hamburg
J. Morgenstern, Nizza
H. Müller, Münster
M. Nivat, Paris
W. Oberschelp, Aachen
M. S. Paterson, Coventry
J. Perl, Berlin
Ch. Rackoff, Rocquencourt

W. J. Erni, Heidelberg
M. J. Fischer, Cambridge (Mass.)
K. Fleischmann, Berlin
P. Fuchs, Frankfurt
J. T. Gill, Stanford
H. F. de Groote, Tübingen
P. Henrici, Zürich
L. Hyafil, Rocquencourt
D. E. Knuth, Stanford
J. C. Lafon, Grenoble
R. Loos, Kaiserslautern
K. Mehlhorn, Saarbrücken
A. Schönhage, Tübingen
D. P. Schnorr, Frankfurt
D. Siefkes, Berlin
E. Specker, Zürich
J. Spieß, Göttingen
H.-J. Stoß, Konstanz
H.-G. Stork, Darmstadt
V. Straßen, Zürich
J. F. Traub, Pittsburgh
R. Weicker, Hamburg
K. Weihrauch, Bonn

Vortragsauszüge

V. Straßen, Zürich: Some Results on Algebraic Complexity

In this survey the order of magnitude of the computational complexity of those problems is given, to which the algebraic geometric method has been applied. These problems include elementary symmetric functions, sums of powers, evaluation of a polynomial at many points, interpolation, symbolic differentiation of an interpolation polynomial. We treat in more detail the expansion of a rational fraction into a continued fraction (Schönhage's algorithm has the optimal order of magnitude uniformly in inputs), and evaluation at many points and interpolation over a finite field. These last results are not yet published.

H. F. de Groote, Tübingen: On the Complexity of Quaternion-Multiplication

Let K be a real field and $H(K)$ the division-algebra of quaternions over K. Then both products uv and vu of the quaternions $u, v \in H(K)$,

$$u = x_1 + x_2 i + x_3 j + x_4 k \ , \quad v = x_5 + x_6 i + x_7 j + x_8 k \ ,$$

can be computed by an algorithm which contains only the

following ten products:

$p_1 = x_1x_5$, $p_2 = x_2x_6$, $p_3 = x_3x_7$, $p_4 = x_4x_8$,

$p_5 = (x_1+x_2)(x_5+x_6)$, $p_6 = (x_1+x_3)(x_5+x_7)$, $p_7 = (x_1+x_4)(x_5+x_8)$,

$p_8 = (x_3-x_4)(x_7+x_8)$, $p_9 = (x_2-x_4)(x_6+x_8)$, $p_{10} = (x_2-x_3)(x_6+x_7)$.

Furthermore this algorithm is shown to be optimal.

Theorem. Every algorithm which computes both products uv and vu of quaternions $u, v \in H(K)$ requires 10 essential multiplications.

J. C. Lafon, Grenoble: Optimal Computation of Certain Sets of Bilinear Forms

Let K be an infinite field $f_1,\ldots,f_p$ bilinear forms of the commutative ring $K[x_1,\ldots,x_m,y_1,\ldots,y_n]$. For certain sets of bilinear forms we determine the minimum number $C(f_1,\ldots,f_p)$ of non-scalar multiplications necessary to compute the p bilinear forms given $K \cup \{x_i\} \cup \{y_i\}$.

Let $Rt(V)$ denote the tensorial rank of the space V generated by the $m \times n$ matrices $B_1,\ldots,B_p$. We have

$$Rt(V)/2 \leq C(f_1,\ldots,f_p) \leq Rt(V) .$$

As in the non-commutative case, $C(f_1,\ldots,f_p) = Rt(V)$, if the space V has a tensorial base ($Rt(V) = \dim V$) . The space of Toeplitz and Hankel $n \times n$ matrices, and the space of cyclic $n \times n$ matrices over an algebraically closed field are shown to have tensorial bases. Consequently, the minimum number of non-scalar multiplications necessary to compute the coefficients of the product of two polynomials of degree n is shown to be $2n + 1$. For the convolution of two vectors of dimension n the unique optimal algorithm with n multiplications is given. The inversion of a triangular Toeplitz matrix requires less than $4n - 3$ multiplications / divisions, polynomial division (degree $2n$ / degree n) only $8n + 3$.

C. P. Schnorr, Frankfurt: Lower Bounds on the Complexity of Monotone Rational Computations

A computation of rational polynomials that only uses variables, positive rational numbers and the operations addition and multiplication is called a monotone, rational computation. We prove a general lower bound on the minimal number of additions in monotone rational computations. This lower bound implies that any monotone rational computation of the n-th degree convolution at least requires n^2-2n-1 additions. $\binom{n}{k}-1$ is the minimal number of additions in any monotone computation of the polynomial that is associated with the k-clique problem for graphs with n nodes.

A. Borodin, Toronto: On the Number of Additions and Arithmetic Complexity

We investigate the number, $p(k)$, of distinct real zeros of any polynomial over the reals which can be computed in k $(\pm)$ operations. Thus far, we can only show

$$3^k \leq p(k) \leq 2^{2^{\cdot^{\cdot^{\cdot^{2}}}}\Big\}k},$$

whereas we hope $p(k) < c^k$ for some c.

G. E. Collins, Kaiserslautern: High-Precision Calculation of Real Algebraic Numbers

Heindel (Jour. A.C.M., 1971) discusses the application of Sturm's theorem and exact arithmetic to isolating the real zeros of an integral polynomial into disjoint intervals. He further discusses the use of interval bisection for refining isolating intervals to arbitrary small length, thereby obtaining approximations of any specified accuracy to real algebraic numbers. Here we present an

alternative refinement algorithm, adapted from Newton's method, in which the interval length converges quadratically to zero. Exact arithmetic is used together with a scheme for adjusting the rational endpoints of the intervals to control the growth of number lengths. A theoretical analysis of the computing time of the algorithm is supplemented with empirically observed results from a SAC-1 implementation.

R. Loos, Kaiserslautern: The Complexity of Arbitrary Precision Arithmetic on Real Algebraic Numbers

A real algebraic number α is represented by a finite isolating rational interval I and a primitive squarefree polynomial over the integers $A(x)$. Using exact interval arithmetic, resultant algorithms, squarefree factorization, Sturm sequences, and interval bisection the maximum computing time to compute $\gamma=\alpha+\beta$ or $\gamma=\alpha\beta$ for real algebraic numbers $\alpha=(I,A)$, $\beta=(J,B)$, $\gamma=(K,C)$ is dominated by

$$n^{15}(n + L(d))^3.$$

The complexity of γ is given by

$$n_c \le n_A n_B,\ L(d_c) \le n^2 + nL(d),\ L_K \le n^4 + n^3 L(d).$$

($n = \max(n_A,n_B)$, $n_A = \deg(A)$, $d = \max(d_A,d_B)$, $d_A = \Sigma|a_i|$, $A(x) = \Sigma a_i x^i$, $L(d) = [\log(d + 1)]$, $L_K = \max(|i|,j,|k|,1)$, $K = (i/j,k/1)$.

The work is based on work of Collins, Heindel, Musser, and Rubald.

J. F. Traub, Pittsburgh: Lower Bound Theorems and Open Problems in Analytic Computational Complexity

A central problem in analytic complexity concerns the solution of the non-linear operator equation $f = o$. Here we deal only with the case where f is a non-linear scalar function. Let

α be a zero and approximate α by a sequence x_i generated by ϕ. The total cost, C_T, of approximating α to within ε is

$$C_T = \log \log \left(\frac{1}{\varepsilon}\right)\left(\frac{c}{\log p}\right)$$

where c is the cost of computing x_{i+1} from x_i and p is the order of ϕ. Let ε be fixed. To obtain lower bounds on cost we need upper bounds on p.

Kung and Traub [1973] have conjectured that for all multipoint iterations based on n evaluations, the order is bounded by 2^{n-1}. The conjecture has been affirmatively settled for $n = 2$ (Kung and Traub [1973], $n = 3$ (Wozniakowski [1974]), and for any iteration using Hermitian information (Wozniakowski [1974]). The conjecture is open for non-Hermitian information with $n \geq 3$.

The use of only restricted information and extended information ($\S f$) is discussed.

M. J. Fischer, Cambridge (Mass.): Combinational Complexity of Symmetric Transitive Closure

Theorem 1 (Fischer, Paterson). Let M be a symmetric $n \times n$ Boolean matrix. M^* can be computed by a logical network of size $O(n^2 \log^2 n)$.

The proof is by finding a Turing machine to compute M^* in time $O(n^2 \log n)$, and then converting it to a logical network using the general method of Fischer and Pippenger (reported at Oberwolfach, 11/73).

An important step in the Turing-machine construction uses a solution to the postman problem which we formulate as follows:

"Given n houses with addresses $1 - n$ to which mail is to be delivered. The postman visits the houses in turn. At each house, he delivers the mail for it and then picks up letters for later houses. He carries a fixed number of mailbags which he uses as pushdown stacks.

Problem: Deliver the letters properly and keep small the number of times letters are handled."

Theorem 2. There is a solution to the postman problem for n houses such that each letter is handled only $O(\log n)$ times.

M. S. Paterson, Warwick: Complexity of Monotone Networks for Boolean Matrix Products

Theorem. The obvious algorithm(i) for Boolean matrix product(ii) is uniquely(iii) optimal(iv) in monotone algorithms(v).

(i) Performs all IJK $\wedge$'s , and then $I(J-1)K$ $\vee$'s .

(ii) $C = A.B$ where

$$C_{ik} = \bigvee_{1 \le j \le J} a_{ij} \wedge b_{jk} \quad \text{for} \quad \begin{matrix} 1 \le i \le I \\ 1 \le k \le K \end{matrix} .$$

(iii) To within commutativity & associativity of $\wedge$ & $\vee$.

(iv) Uses IJK $\wedge$'s and $I(J-1)K$ $\vee$'s .

(v) Straight-line programs using only the monotone operations, $\wedge$, $\vee$.

K. Mehlhorn, Saarbrücken: Heuristics for Constructing Binary Search Trees

Given are n names $B_1,\ldots,B_n$ from a linearly ordered set and $2n + 1$ probabilities $\alpha_0,\beta_1,\alpha_1,\ldots,\beta_n,\alpha_n$. Here β_i is the probability of encountering name B_i , and α_j is the probability of encountering a name between B_j and B_{j+1} .

The problem is to find a tree which minimizes the average number of comparisons needed to locate an element.
Let P_{opt} be that number. Consider the following

Heuristics: Choose the root as to equalize the total weight of the left and right subtree as much as possible.

Let P_{heu} be the weighted path length of the tree yielded by the heuristics.

Theorem 1. $P_{heu} \leq 2 + 1.8\ P_{opt}$.

Theorem 2. $0.6\ H \leq P_{opt} \leq 1.2\ H + 2$

where H is the entropy of the frequency distribution.

A. Schönhage, Tübingen: A New Median Algorithm

The problem of finding the median of $n = 2m + 1$ elements of a totally ordered set is the problem of producing the partial order S_m^m which has a centre which is less than m other elements and greater than each of the m remaining elements. The best upper bound for this problem previously reported is $5.43\ n$. Here (as a result from joint work with Paterson and Pippenger) we present a new algorithm using only $\sim 3n$ comparisons. It is mainly based upon a cheap mass-production of many disjoint smaller S_k^k's ; each of these costs only $\sim 3.5\ k$, i.e. 1.75 per element. If we choose k of order $n^{1/4}$, then the centres of the S_k^k's can be ordered totally at small cost. Proper arrangement of these techniques gives the final result.

M. Nivat, Paris: Sur la classe de langages reconnus en temps linéaire par des machines de Turing à oscillation bornée (non deterministes)

Le present travail effectué en collaboration avec R. Book fait suite à un travail de R. Book, M. Paterson et moi-même.

Nous avons demontré que la classe des langages reconnus en temps linéaire par des machines de Turing à oscillation bornée

s'identifie à celle des images dans un morphisme continu de l'intersection de trois langages linéaires.

Je me propose de décrire plus complètement cette classe de langages dont on démontre qu'elle contient les langages quasi-rationnels et à compteur. De fait elle constitue un AFL clos par intersection donc par substitution.

La question est posée et non resolue de savoir si elle contient tous les algebriques, dont on démontre qu'ils peuvent être reconnus par une MT à oscillation bornée en temps polynomial. Plus, borner l'oscillation d'une MT non deterministe ne coûte qu'un facteur log. en général. On démontre aussi que P à oscillation bornée est different de NP a oscillation bornée.

S. A. Cook, Toronto: p-Verifiable Proof Systems for the Propositional Calculus

An equational proof system PV for number theory is presented in which the formulas are of the form $t = u$, where t, u are terms with free variables, constant symbols, and function symbols ranging over functions in Cobham's class $\mathcal{L}$ of functions computable in polynomial time. The axioms and rules are the usual ones for equality, together with substitution of terms for variables, "induction on notation", and introduction of new function symbols by p-limited recursion on notation. This system PV is the analog for $\mathcal{L}$ of Skolem's equational theory of primitive recursive functions. PV has the property that a proof of $f(x) = g(x)$ gives a uniform way of verifying in time bounded by a polynomial in n that $f(n) = g(n)$ for an arbitrary given n. A proof system for tautologies is defined abstractly as a function

$F: N \xrightarrow{\text{onto}} \{\text{codes for tautologies}\}$

with $F \in \mathfrak{P}$. We say F is <u>p-verifiable</u> iff

$\vdash_{PV} Tr(F(x),y) = 1$, where

$$Tr(x,y) = \begin{cases} 1 & \text{if } y \text{ codes a truth assignment satisfying formula coded by } x \text{ ,} \\ 0 & \text{if truth assignment } y \text{ fails to satisfy } x \text{ .} \end{cases}$$

System F_2 <u>p-verifiably simulates</u> system F_1 iff

$$\exists f \in \mathfrak{P} \quad \vdash_{PV} F_1(x) = F_2(f(x)) \text{ .}$$

<u>Theorem.</u> A system f is p-verifiable iff resolution with extension p-verifiably simulates F .
(The extension rule was suggested by Tsertin as a way of making resolution more powerful).

A. R. Meyer, Cambridge, Mass.: <u>Inherently Difficult Computational Problems: A Summary</u>

A large number of decidable problems in logic and automata theory have been shown to be inherently difficult in the sense that any decision algorithm requires a number of steps growing exponentially or more in the size of inputs to the algorithm. The proofs resemble classical proofs of undecidability. Moreover, implicit in the proofs are concrete constants which provide lower bounds on the size of computational networks for finite functions.

For example, the weak monadic second order theory of successor (WS1S) is decidable. Expressing sentences in the language of WS1S in an alphabet of 63 characters (standard logical connectives, decimal digits, parenthesis, etc.), and coding each character into a six bit binary word, the true sentences of length n correspond to a set of binary words of length $6n$. Let C_n be the smallest loop-free logical circuit (equivalently, straight-line

program) with $6n$ binary inputs and one output using two-argument Boolean operations as primitives to recognize the code words of true sentences.

Theorem (with L. Stockmeyer)
C_{616} contains at least 10^{123} primitve operations.
Remark: 10^{123} protons suffice to densely fill the known universe.

J. Perl, Berlin: On Finding All Solutions of Polynomial complete problems

Given a polynomial complete problem with input size n and the set of solutions, L_o. To find L_o in time $p(n)\text{card}(L_o)$, where p is a polynomial, is equivalent to the problem of finding any of these solutions in a time bounded by $p(n)$. Therefore the complexity $p(n)\text{card}(L_o)$ is possible iff the P-NP-problem has a positive answer.

Here especially for the partition problem an algorithm is presented which in special cases works proportionally to the number of solutions: $C = (n+1)\cdot\text{card}(L_o)$.

In the other cases the algorithm works in a similar way proportionally to the number of solutions and to the cardinalities of sets L_i which can be interpreted as sets of "quasi solutions", depending on the given problem structure:

$$C = \sum_{i=o}^{n} (n-i+1)\cdot\text{card}(\mathbf{L}_i)$$

B. Brosowski, Göttingen: Über eine untere Schranke für die Operationszeit bei Parallelverarbeitung

Mit Hilfe der Wortalgebra W_Δ über einem Typ Δ werden in Analogie zu Straßen: "Berechnung und Programm II" zur Darstellung von Berechnungen Δ-Mengen eingeführt. Die Parallelverarbeitung

durch ein Prozessorensystem $\left(P_k \mid k \in K \right)$ wird durch eine Zerlegung $W_\Delta = \cup W_k$ erfaßt, wobei angenommen wird, daß eine Operation ω eines Elementes $\omega \underline{w} \in W_k$ vom Prozessor P_k in der Zeit $t(\omega)$ ausgeführt werden kann. Es wird dann für eine Δ-Menge die maximale bzw. mittlere Operationszeit $\left(L_K \text{ bzw. } M_K \right)$ bzgl. $\cup W_k$ eingeführt und die Operationszeit einer endlichen Teilmenge F der Vereinigung der Trägermengen einer Δ - Palgebra definiert.

Die Funktionen L_K bzw. M_K genügen den Ungleichungen

$$M_K(F \cup \{\omega\ \underline{a}\}) \leq \frac{t(\omega)}{\#K} + M_K(F \cup \operatorname{Im} \underline{a}),\ \#K < \infty\ ,$$

$$L_K\Big(F \cup \bigcup_{j=1}^{J} \{\omega_j \underline{a}_j\}\Big) \leq \max_j t(\omega_j) + L_K(F \cup \bigcup_j \operatorname{Im} \underline{a}_j)\ ,$$

sofern die ω_i von verschiedenen Prozessoren ausgeführt werden können. Jede Funktion $\lambda_K : F \to [0,\infty]$, die der letzten Ungleichung genügt, ist unter einer Zusatzvoraussetzung eine untere Schranke für L_K.

J. Bečvář, Prague: Complexity of finite problems

Constructive information processing substantially involves transforming, shifting, copying, describing etc. of finite objects. By means of binary encoding a considerable portion of this activity can be converted into cumputation of boolean functions. Their complexity is thus substantial for estimating complexity of (large) finite or infinite problems. Several of the following measures of complexity of boolean functions are discussed: combinational c., syntactical c., finite automaton c., Turing machine c., Markov algorithm c., program c., decision algorithm c., generative c.

Combinational and program complexity of boolean functions have

appeared in inequalities involving also computational time and space. Continuing in this direction, one has to expect the development of putting quantitative computational characteristics into such relations which would be invariant within certain classes of situations; and, further, to link, if possible various such classes together in a way which would yield analogy of conservation laws for some quantitative characteristics.

P. Fuchs, Frankfurt: $\exists^\infty$-speed-up of Program Complexity

Let $A: X^* \times N \to X^*$ be any partial recursive function ("algorithm"). In $A(x,n)$ we consider x as a program which on input n (possibly) yields the output $A(x,n) \in X^*$. If $A(x,n)$ converges, we call x a description of $A(x,n)$. For any recursive function t and any finite binary sequence x we define $K^t(x)$ to be the length of a shortest program for x (with respect to the universal algorithm A) with running time $\leq t|x|$.

There are infinite binary sequences z with property
$\forall t \text{ rec} \exists t' \text{ rec } \forall q < 1 \; \exists^\infty n : K^{t'}(z(n)) + qn \leq K^t(z(n))$
(where $z(n)$ denotes the initial segment of lenth n).

We call an infinite binary sequence 'learnable' iff there is a computational resource bound t which is best, i.e. no linear $\exists^\infty$-speed-up is possible by taking a bigger resource bound. Learnable sequences are representative for some recursive distribution; the nonlearnable sequences form a set of measure 0 with respect to any recursive distribution. Let $\overline{\mu}$ be any computable product measure. If z violates no $\overline{\mu}$-law of exponential order, then we have

$$\exists t \text{ rec} : \lim K^t(z(n))/n = H(\mu) ,$$

and $\forall t' \text{ rec} : \underline{\lim}\, K^{t'}(z(n))/n \geq H(\mu)$.

J. T. Gill, Stanford: Probabilistic Turing Machines and Complexity of Computation

Probabilistic Turing machines are Turing machines with the ability to make decisions based on the outcomes of fair coin tosses. For any probabilistic Turing machine (PTM) M and any input x, we let $M(x)$ be the output of M, a random variable. We define the function computed by M to be the majority output function; that is, M computes f iff $P\{M(x) = f(x)\} > \frac{1}{2}$ for all x.

The error probability function for a PTM that computes a total function f is defined by $\epsilon(x) = P\{M(x) = f(x)\}$. Note that $\epsilon(x) < \frac{1}{2}$ for all x. To avoid pathologies, we shall consider only PTMs with error probability functions strictly bounded below $\frac{1}{2}$, that is $\epsilon(x) \leq \varepsilon < \frac{1}{2}$ for all x.

Proposition 1: Only partial recursive functions are computed by PTMs.

Proposition 2: If f is computed by a PTM with average running time T, then f can be computed by a deterministic TM in time $2^{O(T)}$.

Proposition 3: There is a language recognized faster by PTMs than by deterministic TMs. More specifically, there is a language L and a sublanguage $L_1 \subseteq L$ such that

(i) Every 1-tape deterministic TM that recognizes L requires time $\geq O(n^2)$ for infinitely many inputs in L_1 of length n.

(ii) There is a 1-tape PTM that recognizes L and runs in time $O(n \log n)$ for inputs in L_1.

Proposition 4: Every nondeterministic linear-bounded automaton can be simulated by a probabilistic linear-bounded automaton.

R. Weicker, Hamburg: Random Access Machines with Associative Memory Access

Such RAS is defined as a RAM (with +1, -1 as arithmetic operations) with additional instructions for an associative memory access:

$X_i \leftarrow X_j$,

$X_i \leftarrow R_{X_o}$; $R_{X_o} \leftarrow X_j$;

$X_i \leftarrow X_j + 1$, $X_i \leftarrow X_j \dot{-} 1$,

TRA on $X_j > 0$,

Read X_i , Print X_i ,

where X_i, X_j denotes the contents of directly addressable registers, R_i, R_j of indirectly addressable registers; the additional instructions of a RAS are

$$X_i \leftarrow ASSL_{X_j} \quad , \quad X_i \leftarrow ASSR_{X_j} \quad ,$$

where $ASSL_{X_j} = \max \{1 \mid 1 \leq X_o \text{ and } R_1 = X_j\}$,

$ASSR_{X_j} = \min \{1 \mid 1 \geq X_o \text{ and } R_1 = X_j\}$.

Theorem: Any RAS with time bound $T(n)$ can be simulated by a RAM with time bound $T(n) \left(\log T(n) \right)^2$.

A similar result (with an extra factor $\log T(n)$) holds, if we admit associative memory access operations controlled by the prefix of some register contents. The results also hold for machines with stronger arithmetic operations as long as the numbers computed in time t do not exceed t^c for some constant c .

B. Monien, Hamburg: Reducibility of Complexity Problems to Problems concerning Automata

The following relations are proved:

$$\begin{array}{ccc}
\exists_j\ P_1^{un} \subset D_j & & \exists_j\ N_S{}^{un} \subset D_j \\
\Updownarrow & & \Updownarrow \\
\text{TAPE}(n) = \text{TIME}(\exp(n)) & \Longrightarrow & \text{TAPE}(n) = \text{NTAPE}(n) \\
\Uparrow & & \Uparrow \\
\text{TAPE}(\log n) = \text{TIME}(\text{Pol}(n)) & \Longrightarrow & \text{TAPE}(\log n) = \text{NTAPE}(\log n) \\
\Updownarrow & & \Updownarrow \\
\exists_j\ P_1 \subset D_j & & \exists_j\ N_2 \subset D_j
\end{array}$$

where $\text{TAPE}(f(n))$, $\text{TIME}(f(n))$, $\text{NTAPE}(f(n))$, $\text{NTIME}(f(n))$ are the families of all languages acceptable by (nondeterministic) Turing machines with tape (time) bound $f(n)$. P_k (D_k, N_k) are the families of all languages acceptable by k-head 2-way deterministic pushdown automata (deterministic, nondeterministic finite automata). $P_k{}^{un}$, $N_k{}^{un}$ are the corresponding families with unary input.

H.-G. Stork, Darmstadt: Über ein mittels endlicher Automaten erklärtes Komplexitätsmaß für einfache Standardmengen

Programme werden als einfach strukturierte reguläre Ausdrücke über einem endlichen Alphabet $V = \{v_0, \ldots, v_{r-1}\}$ definiert. Ferner haben wir eine Klasse von Homomorphismen $h: V^* \to [r]^*$. Die Bilder von Elementen der durch Programme bezeichneten Mengen werden durch Automaten mit Kostenstruktur verarbeitet. Mittels

dieser Kostenstruktur wird ein Komplexitätsmaß für Programme definiert. Die Kosten der Verarbeitung eines Programms hängen ab vom gewählten Homomorphismus und dem jeweiligen Automaten. Scharfe untere Schranken werden hergeleitet für sehr einfache Teilklassen von Programmen. Die Begriffsbildungen sind motiviert durch automatische "overlay"-Verfahren in Betriebssystemen.

O. Axelsson, Göteborg: On the Computational Complexity of Some Matrix Iterative Algorithms

The importance of iterative methods in practical problems seems not to be appreciated enough. Such algorithms are simple in construction, can be interrupted when the demanded accuracy is reached, and for important classes of problems - as discretized selfadjoint elliptic partial differential equations - they are even faster than commonly used direct (factorization) methods. The demand of memory is kept to a minimum - in the simplest version, only the matrix coefficients or a procedure for generating them, is needed.

An upper bound of the computational complexity of such algorithms for the solution of the linear problem $Ax = f$, A, a Hermitian, positive definite matrix, is directly proportional to the square root of the spectral condition number and to the complexity of the product of the matrix and a vector.

J. Spieß, Göttingen: Some Practical Aspects of Matrix Multiplication

The improvements in operation time for multiplication of (m,n) by (n,p) matrices given by the algorithms of Winograd and Straßen are considered. If the condition $\frac{1}{5} > \frac{1}{m} + \frac{1}{n} + \frac{1}{p}$ holds, one "Straßen-step" reduces the number of arithmetic operations. A quite realistic computer model shows that even for small square

matrices of order ≥ 36 an improvement in operation speed can be obtained. It is shown that Straßen's algorithm for (n,n) square matrices needs $2\,(n/2)^2$ additional storage cells for one iterative step. Recursive application needs at most

$$2\left((n/2)^2 + (n/4)^2 + \ldots\right) = \frac{2}{3}\,n^2 .$$

P. Brucker, Regensburg: Theorie der Matrix Algorithmen

Im Jahre 1962 veröffentlichte Floyd einen Matrixalgorithmus zur Bestimmung der Längen aller kürzesten Wege in einem Netzwerk. Ändert man die dem Netzwerk zugrunde liegende algebraische Struktur, so läßt sich das gleiche Verfahren auf andere Probleme, wie z.B. Probleme der Bestimmung zuverlässigster Wege und Wege größter Kapazität in Netzwerken, Ganzzahlige Optimierungsprobleme, sowie Probleme der Bestimmung von Zusammenhangskomponenten in gerichteten Graphen, anwenden. Es wird eine allgemeine Netzwerktheorie entwickelt und untersucht, unter welchen Voraussetzungen der Matrixalgorithmus konvergiert. Außerdem wird der Rechenaufwand des Algorithmus für beliebige und speziell strukturierte Netzwerke analysiert und mit dem Aufwand bei anderen Algorithmen verglichen.

Auszug aus dem "Vortragsbuch"

Wette

Zwischen Ernst Specker und Volker Straßen, beide Zürich, wird eine Wette um folgende Aussage A abgeschlossen:

Die Menge der Primzahlen ist in $\mathcal{P}$

d.h. es gibt eine deterministische Turingmaschine im Sinne von Turing, welche für jede dezimal codierte Eingabe n, $n \in \mathbb{N}$, die Primheit von n in einer Schrittzahl entscheidet, welche

durch ein Polynom in $\log n$ nach oben beschränkt ist.

Volker Straßen gewinnt die Wette, falls bis zum 1. November 1984 ein Beweis für A publiziert ist, der im Prinzip in ZF durchführbar ist. Andernfalls ist Ernst Specker Gewinner.

Der Verlierer lädt den Gewinner alsbald zu einer Ballonfahrt ein oder entschädigt ihn durch fünfzig Gramm Gold.

Oberwolfach, den 31. Oktober 1974

gez.: Volker Straßen gez.: Ernst Specker

gez.: W. Oberschelp
(Protokollant)

A. Schönhage (Tübingen)