MATHEMATISCHES FORSCHUNGSINSTITUT OBERWOLFACH

T a g u n g s b e r i c h t 41|1977

Algorithmen und Komplexitätstheorie
16.10. bis 22.10.1977

Die dritte Tagung über Algorithmen und Komplexitätstheorie stand wieder unter der Leitung von C.P. Schnorr (Frankfurt), A. Schönhage (Tübingen) und V. Strassen (Zürich). Von den insgesamt 49 Teilnehmern aus 9 Ländern kamen 17 von ausserhalb Europa.

Mit 40 Vorträgen war das Programm wesentlich dichter gedrängt als früher. Schwerpunkte bildeten algebraische Komplexitätstheorie, insbesondere die der endlich dimensionalen linearen Algebren, sowie Sprachen- und Automatentheorie, insbesondere Fragen, die mit der Cookschen Hypothese zusammenhängen. Eine grössere Anzahl von Beiträgen befasste sich mit kombinatorischen Algorithmen, deren Datenstrukturen und Komplexität. Ferner wurde über so verschiedenartige Gegenstände wie Mustererkennung, Komplexität von Spielen, Komplexität logischer Theorien, Boolesche Komplexität und Kolmogorovsche Komplexität berichtet.

Teilnehmer

H. Alt, Saarbrücken
P. van Emde Boas, Amsterdam
A. Borodin, Toronto (Canada)
H. Brakhage, Kaiserslautern
H. Bremer, Frankfurt a. M.
P. Brucker, Oldenburg
G. E. Collins, Madison (USA)
St. A. Cook, Toronto (Canada)
W. Erni, Heidelberg
F. Fiala, Ottawa (Canada)
M. J. Fischer, Seattle (USA)
M. Fuerer, Zürich (Schweiz)
P. Gács, Frankfurt a. M.
Z. Galtl, Tel-Aviv (Israel)
H. F. de Groote, Tübingen
J. Heintz, Frankfurt a. M.
G. Hotz, St. Ingbert
D. E. Knuth, Stanford (USA)
B. Korte, Bonn
R. E. Ladner, Seattle (USA)
J. C. Lafon, Strasbourg (Frankreich)
J. van Leeuwen, Utrecht (Niederlande)
R. Loos, Karlsruhe
A. R. Meyer, Cambridge (USA)
M. Mignotte, Strasbourg (Frankreich)
B. Monien, Paderborn
J. Morgenstern, Nice (Frankreich)
M. Nivat, Paris
M. S. Paterson, Coventry (England)
W. J. Paul, Bielefeld
A. Paz, Haifa (Israel)
J. de Pillis, Riverside (USA)
V. R. Pratt, Cambridge (USA)
Ch. Rackoff, Toronto (Canada)
J. E. Savage, Providence (USA)
A. Schoenhage, Tübingen
C. P. Schnorr, Frankfurt a. M.
P. Schuster, Zürich (Schweiz)
D. Siefkes, Berlin
M. Sieveking, Frankfurt a. M.
H. J. Stoss, Konstanz
V. Strassen, Zürich (Schweiz)
J. F. Traub, Pittsburgh (USA)
L. G. Valiant, Edinburgh (Großbritannien)
K. Weihrauch, Aachen
R. Weicker, Erlangen
M. Wietlisbach, Zürich (Schweiz)
S. Winograd, Yorktown Heights (USA)
A. C. Yao, Stanford (USA)

Vortragsauszüge

Lower bounds on formula size for symmetric Boolean functions

Mike Paterson, Coventry

Two lower bound results are described, each a corollary of a more general theorem for arbitrary functions. The first theorem, due to Krapchenko, is given a new, more elegant, proof by induction on formula size. The second theorem is new and is due to M. Fischer, A. Meyer and M. Paterson.

Let B_n denote the set of n-argument Boolean functions, and $S_n (\subset B_n)$ the subset of symmetric functions. Each $f \in S_n$ can be defined by a Boolean vector $\langle f_0,\ldots,f_n\rangle$ giving the value of f(x) for $\sum_i x_i$ ranging from 0 through n. $L_B(f)$ denotes the number of occurrences of variables in the smallest formula for f, with connections from the basis B. Let B_2 denote the set of all 16 2-argument Boolean functions, and $U = B_2 - \{\oplus, \equiv\}$.

RESULT A (Corollary of Krapchenko's theorem)

If $f \in S_n$ and $f_{k-1} \neq f_k$ then $L_U(f) \geq (n-k+1)k$

RESULT B (Corollary of FMP theorem below)

If $f \in S_n$ and $f_{k-1} \neq f_{k+1}$ then $L_{B2}(f) \geq cn \cdot \log k$ (c constant)

To state our new theorem, some further definitions are needed. $f \in B_n$ is affine if $f(x) = a_0 + (a_1 \wedge x_1) + \ldots + (a_n \wedge x_n)$ where $a_i \in \{0,1\}$. A central subfunction of f is obtained by giving some equal numbers of arguments of f to 0 and to 1.

FMP-THEOREM (Fischer, Meyer, Paterson)

If $f \in B_n$ and $L_{B2}(f) \leq rn$ then f has an affine, central subfunction of at least n/k^r arguments where k is independent of n,r.

A new complexity measure for languages

Maurice Nivat, Paris
(in collaboration with Luc Boasson and Bruno Courcelle)

Call Rat_n the family of rational languages which can be recognized by a non deterministic finite automaton with a number of states less than or equal to n.

Call $\delta_{K,L}$ the minimal length of a word in $K \cap L$. We consider the function g_L, attached to a given language L and called the rational index of L:

$$g_L(n) = \max\{\delta_{K,L} \mid K \in Rat_n\}$$

On the contrary of what happens for most complexity measures which have been proposed up to now for languages, the rate of growth of g_L is a complexity measure which behaves "well" when the standard operations of language theory are performed on L. We write, for all pairs of functions f, g of $\mathbb{N}$ into $\mathbb{N}$ $f<g$ iff for infinitely many n $f(n) \leq g(n)$

Results $g_{L \cup L'} < \max(g_L, g_{L'})$

$g_{LL'} < g_L + g_{L'}$

$g_{L^*} < \lambda n\, n g_L(n)$

If τ is a rational transduction which maps L onto L'

$g_{L'} < \lambda n\, n g_L(pn)$ for some integer p.

If $L \circ L'$ denotes the marked substitution of L' into L

$g_{L \circ L'} < \lambda n\, g_L(n)(1+g_{L'}(n))$

From these lemmas the two following theorems follow easily

Th 1 Call Less(Pol) the family of languages L such that

$g_L < \lambda n\, n^k$ for some $k \in \mathbb{N}$

Then Less(Pol) is a substitution closed rational cone (or AFL)

Th 2 Call More(Exp) the family of languages such that

$\lambda n\, 2^{\alpha n} < g_L$ for some $\alpha \in \mathbb{R}$, $\alpha > 0$.

Then More(Exp) is an anticone which means that $L \in$ More(Exp) and there exists a rational transduction of L' onto L imply $L' \in$ More(Exp).

We then remark that if d_2' is the language generated by the grammar $\xi = a_1 \xi \xi \bar{a}_1 + a_2 \bar{a}_2$ then $\lambda n\, 2^n < g_{d_2'}$.

We also remark that if L is a linear algebraic language, or a one counter language $L \in$ Les(Pol)
The following inclusions follow

Gen$\subset$ More(Exp) where Gen is the family of generators of Alg, as a rational cone.

Gre$\subset$ Les(Pol) where Gre is the least substitution closed rational cone containing the linear and one counter languages (Gre is the family of Greibach languages)

We can show that the last inclusion is proper by looking at at the rational index of Δ_2, the language proved by L. Boasson to be in Alg$\setminus$(Gre$\cup$Gen)
We end with the two conjectures

Conjecture 1 More(Exp)$\cap$ Alg = Gen

Conjecture 2 Alg = (Alg$\cap$ Les(Pol))$\cup$(Alg$\cap$ More(Exp))

These two conjectures are closely linked to long standing conjectures of S. Greibach and the authors.

The Complexity of Counting Problems

L.G. Valiant, Edinburgh

A completeness class intermediate between the NP-complete and the PSPACE-complete problems is established. Typical numbers of this class (called the #P-complete problems) are the problems of counting the number of solutions for NP-complete problems. The importance of the class follows from the discovery that for numerous problems for which the detection of a solution is polynomial time computable, the counting problem is still #P-complete. Examples of such problems are (i) counting perfect matchings in bipastite graphs, or equivalently the permanent of (0-1)-matrices, (ii) counting trees in directed graphs, (iii) the probability that in a given graph with edge probability a half, two given nodes are connected, (iv) counting maximal cliques, and (v) counting the number 1 satisfying assignments for monotone Boolean formulae in 2-conjunctive normal form.

Complexity of Scheduling Problems

Peter Brucker, Oldenburg

A survey on complexity of scheduling problems is given. Scheduling problems on single, different and identical machines are classified and the influence of various parameters on the complexity is studied. The problems for which a polynomial-bounded algorithm is available are listed. NP-completeness is established for a large number of other scheduling problems leaving only a small class of problems for which it is unsettled wether they are polynomial-bounded or NP-complete.

Proofs of optimality for bilinear forms computation

Jean Claude Lafon, Strasbourg

We show how to obtain lower bound for the minimum number of general multiplications necessary to compute p bilinear forms $x^t B_i y$ $(i=1,\ldots,p)$, $B_i \in \mathcal{M}_{m,n}(K)$ (K being a field of characteristic different of two), $x^t = (x_1,\ldots,x_m)$, $y^t = (y_1,\ldots,y_n)$, $z^t = (z_1,\ldots,z_p)$.

If we take $x^t B_k y = \sum_{i,j,k}^{m,n,p} b_{ijk}\, x_i y_j z_k$ then we have to compute the rank of the tensor (b_{ijk}) of $K^m \times K^n \times K^p$, that is the least integer q such that:

$$(b_{ijk}) = \sum_{j=1}^{q} u_j \otimes v_j \otimes w_j \quad u_j \in K^m,\ v_j \in K^n,\ w_j \in K^p.$$

Equivalently we have:

(1) $x^t B(z) y = \sum_{j=1}^{q} (u_j^t x)(v_j^t y)(w_j^t z)$ and is the tensorial rank of this bilinear form.

To obtain a lower bound for this tensorial rank, we describe two kinds of methods. In the first, we use linear substitution of the form $x \longrightarrow Tx$ (for example), to make some elements of (1) be equal to zero, and then we have to study the remaining bilinear forms. In the second kind of methods, we use invariant transformations to do the same thing:

An invariant transformation is a transformation $x \to T_1 x$, $y \to T_2 y$, $z \to T_3 z$ (T_1, T_2, T_3 being regular) such that:

$x^t B(z) y \equiv x^t T_1^t B(T_3 z) T_2 y$ (the bilinear form is invariant).

With these techniques, we can obtain a lower bound of $2n-1$ for the product of two nxn matrices and results of optimality for the following computation: vectorial product-polynomial product-convolution of two vectors - quaternion multiplication.

On varieties of optimal algorithms for the computation of a bilinear mapping

Hans F. de Groote, Tübingen

Let K be a field, $\Phi: K^\ell \times K^m \to K^n$ a bilinear mapping with corresponding tensor $t \in K^\ell \otimes K^m \otimes K^n$. The notion "equivalence of algorithms for the computation of Φ" was discussed in the non-commutative model of computation. The scaling equivalence classes of length-R-algorithms for Φ are in one-to-one correspondence of the set

$$\mathcal{D}_R(\Phi) = \{(u_1 \otimes v_1 \otimes w_1, \ldots, u_R \otimes v_R \otimes w_R) / \sum_{r=1}^{R} u_r \otimes v_r \otimes w_r = t\}$$

of ordered decompositions of t into R tensors of rank one. The group Γ of automorphisms φ of $K^\ell \otimes K^m \otimes K^n$ that are of the form $\pi o(A \otimes B \otimes C)$ (π a suitable permutational mapping) and leave the tensor t fixed operates on the variety of length-R-algorithms for Φ. Γ_Φ is called the isotopy group Φ.

$$\Gamma_\Phi^o := \{A \otimes B \otimes C / (A \otimes B \otimes C)t = t\}$$

is a normal subgroup of Γ_Φ.

The group γ_R of permutations of $\{1, \ldots, R\}$ acts on $\mathcal{D}_R$ and since the actions of γ_R and Γ_Φ commute on $\mathcal{D}_R$,

$G_\Phi := \Gamma_\Phi \cdot \gamma_R$ is a group, too. G_Φ is called the extended isotopy group of Φ.

Def. Two (scalingequivalence classes of) algorithms for Φ are called equivalent iff they are in the same G_Φ-orbit.

For some important cases, isotopy groups were determined, e.g.

Prop. Let A be a finite dimensional unital central simple algebra over K, Φ the multiplication of A. Then $A \otimes B \otimes C$ is an element of Γ_A^o iff there are units a, b, $c \in A$ such that

$$A^T = L_{a^{-1}} R_b,\ B^T = L_{b^{-1}} R_c,\ C = L_a R_{c^{-1}}$$

where L, R are the operators of left and right multiplications respectively. (This was proved independently also by V.Strassen.)

The following results on concrete algorithm varieties were presented:

Theorem A Let K be a field, $\mathbb{M}_2(K)$ the algebra of 2×2-matrices over K, Φ the multiplication of $\mathbb{M}_2(K)$. Then every optimal algorithm for Φ is equivalent to Strassen's algorithm. Precisely $G_\Phi^o := \Gamma_\Phi^o$. $\mathfrak{S}_7$ acts transitively on the variety of scaling equivalence classes of optimal algorithms for Φ

Theorem B Let $\Phi: \mathbb{M}_2(K) \times \mathbb{M}_2(K) \longrightarrow \mathbb{M}_2(K) \times \mathbb{M}_2(K)$

$$(x,y) \longrightarrow (xy, yx)$$

(i) If the characteristic of K is different from two, then optimal algorithms have length nine. Moreover there are several different equivalence classes of optimal algorithms for Φ.

(ii) If K is the Galois field GF(2), then optimal algorithms for Φ have length ten.

For both cases optimal algorithms were presented.

Multiplicative Complexity of Product of Polynomials

Shmuel Winograd, Yorktown Heights

Fourier transform leads to the problems of the computational complexity of the product of two polynomials with indeterminate coefficients modulo a third polynomial with constant coefficients. As a first step all (multiplicatively) minimal algorithms for computing the coefficients of $(\sum_{i=0}^{m} x_i u^i)(\sum_{i=0}^{n} y_i u^i)$ (i.e. all the algorithms using m+n+1 multiplications) were classified. This result

was used to classify all (multiplicatively) minimal algorithms for computing the coefficients of $(\sum_{i=0}^{n-1} x_i u^i)(\sum_{i=0}^{n-1} y_i u^i)$ mod $P_n(u)$, where $P_n(u)$ is an irreducible polynomial. The number of multiplication in this case is 2^n-1. By the Chinese Remainder Theorem the problem of computing the coefficients of $(\sum_{i=0}^{n-1} x_i u^i)(\sum_{i=0}^{n-1} y_i u^i)$ mod $P_n(u)$, where $P_n(u) = \prod_{i=1}^{k} P_i^{e_i}(u)$, is reduced to the k independent problem of multiplication modulo $P_i^{e_i}(u)$. It is shown that the only (multiplicatively) minimal algorithms are those which compute each of the k problems separately, and therefore require 2n-k multiplications. These results led to new algorithms for computing the Discrete Fourier Transform. The problem of computing the coefficients of the product of two polynomials in several variables modulo polynomials in each of the variables modulo polynomials in each of the variables was shown to be equivalent to several independent problems of this kind in one variable. This latter result leads to new algorithms for computing the multidimensional Discrete Fourier Transform.

Additive Complexity

Jacques Morgenstern, Nice

When one wants to compute a set of linear forms by a linear algorithm one can save additions if certain relations occur between the coefficients of the given forms. A knowledge of these relations would be of interest to find lower bounds.

In the simple case of three forms on three variables, the possible configurations in the projective plane $P_2(k)$ are of the following forms:

. 3 points are on the same line

. there exists a homology (with a center and cross ratio k) which transform some given lineas into other given lines.

The relations could be viewed as conditions for roots of a certain matrix to be multiple.

In general the conditions are given by polynomials which are homogeneous with respect to each row and each column that appears in the polynomial.

Playing "Twenty Questions" Against a Liar

Albert R. Meyer, Cambridge

Searching for an unknown $x \in \{1,\ldots,n\}$ by comparing x to constants can be carried out with

$$\log_2 n + k \cdot \log_2 \log_2 n + O(k)$$

comparisons in the worst case, even when the answers to as many as k of the comparisons may be erroneous. For k fixed independent of n, this bound is within an additive constant of optimal, even if arbitrary "Yes-No" questions about x are allowed.

The problem of determining an interval of size $\leq \frac{1}{2}$ containing an unknown $y \in [0,1]$ is essentially equivalent, and the above results carry over to this continuous version of the identification problem.

The results were obtained jointly with D.J. Kleitman of M.I.T., R.L. Rivest of M.I.T. and J. Spencer Suny, Stony Brook, U.S.A.

Complexity of guessing-games

Vaughan R. Pratt, Cambridge

We consider the problem of guessing a number n by asking only questions of the form "$x \leq n$?" for various x. In place of the usual complexity measure for this problem (number of questions asked) we count the total time required to formulate all the questions, in terms of the number of applications of functions. Here an interesting choice of constants and functions is 0, successor, and doubling. For the nondeterministic case we easily prove a $\Theta(\ell)$ bound where $\ell = \log_2 n$ (i.e. upper and lower bounds both linear in ℓ). For the deterministic case we prove a $\Theta(\ell^2)$ bound. The upper bound ($O(\ell^2)$) is obtained with a

binary search algorithm which computes each of the ℓ bits of n at cost $O(\ell)$. The lower bound ($\Omega(\ell^2)$) is proved by arranging $\mathbb{N}$ as a spiral in the plane, so that successor takes us around the spiral while doubling takes us radially outwards. An adversary argument identifies $\Omega(\ell)$ questions that a given algorithm must use such that each question can be charged with at least $\ell/2$ function applications, giving $\Omega(\ell^2)$ function applications altogether.

Higher order network complexities

C.P. Schnorr, Frankfurt

We encode a Boolean network with $\leq 2^r$ nodes as a Boolean function $g: B^r \to B^r \times B^r \times B^4 \times B$.

Let $x \in B^r$ be a node with $g(x) = (g_1(x), g_2(x), g_3(x), g_4(x))$ then $g_1(x), g_2(x)$ are the first and second predecessor of x, $g_3(x)$ encodes the operation at node x and $g_4(x) = 1$ iff x is a terminal node. The k-order network complexity of the Boolean function f is defined as

$C_k(f)$ = minimal size of a network

$k-1$ { that computes the encoding of a network
that computes the encoding of a network
$\vdots$
that computes the encoding of a network
that computes f

Thm 1: $C_K(f) \leq C_{k-1}(f)(1+O(n)/n)$ for all $f: B^n \to B$

Thm 2: For any Turing program P for $f: B^n \to B^m$

$$C_k(f) \leq \text{const}_k\ (||P|| + \log^{k-1}(T_P))$$

where $||P||$ is the size of the program, T_P its maximal running time. Here we only consider programs that also determine the length n of the input of the computed function f.

Why Should A Table Be Sorted?

Andrew C. Yao, Stanford

To retrieve an item from a table of size n, it is known that $\lfloor \lg(n+1) \rfloor$ probes are necessary and sufficient in the worst case, if the table is sorted. Do there exist entirely different scheme of arranging items in a table (such as hashing) that improves on this bound? We prove that, for any table arrangements, $\lfloor \lg(n+1) \rfloor$ probes are required in the worst case, provided that the key space is large. For smaller key space, more efficient schemes may exist. For example, we show that a single probe can determine whether an item is in the table, if and only if the key space contains fewer than $2n-1$ keys ($n \geq 3$). We also show that, with one additional location, an item can be retrieved in two probes provided that the key space is large. These results are obtained by combinatorial reasoning rather then the usual information-theoretical argument.

Relativized Complexity Classes

Charles Rackoff, Toronto

Let $R \subseteq NP$ be the set of languages L such that for some polynomial time predicate Q and number k,

$L = \{x \mid \exists\, y, |y| = |x|^k, Q(x,y)\}$

$= \{x \mid \exists$ at least $2^{|x|^k - 1}$ values of y, $|y| = |x|^k, Q(x,y)\}$.

Define $U \subseteq NP$ to be the set of languages L such that for some polynomial time predicate Q and number k,

$L = \{x \mid \exists\, y, |y| = |x|^k, Q(x,y)\}$

$= \{x \mid \exists$ a unique y, $|y| = |x|^k, Q(x,y)\}$.

For a set A, let P^A be the class P defined with respect to machines using A for an oracle; define NP^A, R^A, etc. similarly.

Theorem: There exist oracles A B such that

$U^A = R^A = CO\text{-}NP^A = NP^A \neq P^A$ and

$NP^A \neq P^A = U^A = R^A = (NP^A \cap CO\text{-}NP^A)$.

Worst-case-behaviour and lower bounds of combinatorial algorithms

Bernhard Korte, Bonn

Let us consider a subclusive set system $(E,\mathcal{S})$, i.e. $|E|<\infty$ and $\mathcal{S} \subset \mathcal{P}(E)$ such that $S_1 \subset S_2 \in \mathcal{S} \Longrightarrow S_1 \in \mathcal{S}$ and an optimization problem over it: $\max\{c(S) \mid S \in \mathcal{S}\}$ with $c: \mathcal{S} \to \mathbb{R}$ separable on E (submodular on $\mathcal{S}$). We first analyze the worst-case-behaviour of the most common algorithmic approach to this general problem, namely the GREEDY-algorithm. Let $c(S_G)$ be the solution of the greedy algorithm and $c(S_O)$ the optimal solution. Then (theorem):

$$\forall c \qquad 1 \geq \frac{c(S_G)}{c(S_O)} \geq \min_{F \subseteq E} \frac{\ell_r(F)}{ur(F)}$$

whereas $ur(F) := \max\{|S \cap F| \mid S \in \mathcal{S}\}$ and
$\ell_r(F) := \min\{|S| \mid S \subseteq F, S \in \mathcal{S}, \forall e \notin F \setminus S: S \cup \{e\} \in \mathcal{S}\}$.
A natural question which might be asked: Is there any polynomial approximative algorithm for general subclusive set systems (or general 0-1 integer programming problems) which might have a better guaranteed performance than the greedy. The answer is no, i.e. (theorem) any oracle algorithms which check wether $S \in \mathcal{S}$ or not by an arbitrary oracle needs $O(2^{|E|})$ calls on its oracle in order to have a better worst-case-behaviour than the greedy.

On Kolmogorov's complexity

Peter Gàcs, Budapest

The logarithm of the maximal semicomputable measure (the pror probability) and the variant of Kolmogorov's complexity defined by Levin, K(x), are known to coincide up to an additive constant. The proof of this theorem gives rise to a storage allocation problem which we solve within some degree of accuracy. We also show that K(K(x)x) is not always less than log n - loglog n, where n is the length of x, although the prior probability of the trouble-making x's is small.

Simulation of multi-dimensional Turing machines

Martin Fürer, Zürich

T steps of a Turing machine M with several heads on multi-dimensional tapes can be simulated in time $O(T^2)$ with a Turing machine M' with one head on one linear tape. (Straight forward simulations need time $O(T^2 \log T)$.) With a similar method we can get the improvement of M.J. Fischer and N.J. Pippenger ($O(T^{2-1/d})$) of the result of H.J. Stoss ($O(T^{2-1/d} \log T)$) for the simulation with two tapes.

On the tape of M' we write a chronological description of the symbols written and the moves executed by M. On other tracks of the tape we write decimal coordinates of some squares visited by M. The length of the coordinates determines the number of squares without coordinates between two squares with coordinates. The origins of the coordinate systems are the positions of the heads of M, they are always changing. So all interesting squares have small coordinates. In each simulation step we change all coordinates (by one), change the dinstance between two coordinates (in linear time) if necessary and look for squares with coordinates 0.

Schemes for fast matrix multiplication

John de Pillis, Riverside

Given real m×n matrix $A = (a_{ij})$, and
g×r matrix $B = (b_{k\ell})$, then
two products between A and B are defined, viz

(1) $A \otimes B$ (the tensor product, or Kronecker product)

In matrix form,
$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & \vdots & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix} \quad \updownarrow mq$$
$$\longleftarrow nr \longrightarrow$$

In operator form, $A \otimes B: C \longrightarrow ACB^t$ for appropriately dimensioned matrices C.

(2) $(A][B)$ (the dyad product)

In matrix form
$$(A][B) = \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & \cdots & a_{11}b_{qr} \\ a_{12}b_{11} & a_{12}b_{12} & \cdots & a_{12}b_{qr} \\ \vdots & \vdots & & \vdots \\ a_{mn}b_{11} & a_{mn}b_{12} & \cdots & a_{mn}b_{qr} \end{pmatrix} \quad \updownarrow mn$$
$$\longleftarrow qr \longrightarrow$$

Question: How Do "$\otimes$" and "][" Relate to Matrix Multiplication?

Answer: Given the product of A m×n with B n×q
$A \cdot B = C = (c_{ik})$; $i=1,2,\ldots,m$, $k=1,2,\ldots,q$, then
$c_{ik} = \sum_{j=1}^{n} a_{ij} \cdot b_{jk}$ ' $c_{ij}(A,B)$, a bilinear functional in A, B. Thus (Reisz reprs'n thm., eq)
$c_{ik}(A,B) = \langle A \otimes B, \mathcal{R}_{ik} \rangle$
for some unique mn×np matrix $\mathcal{R}_{ik}$. (Here $\langle \cdot , \cdot \rangle$ denotes the inner product.) In fact,
$$\mathcal{R}_{ik} = \sum_{j=1}^{n} E_{ij} \otimes E_{jk}, \text{ where}$$
E_{rs} is the appropriately dimensioned matrix with 1 in the rs-th place and zeros elsewhere.

Note, the mnp element set
$$\mathcal{E} = \{E_{ij} \otimes E_{jk} : i=1,2,\ldots,m, j=1,2,\ldots,n, k=1,2,\ldots,p\}$$

spans all mp-element set

$$\mathcal{R}' = \{\mathcal{R}_{ik}: i=1,2,\ldots,m,\ k=1,2,\ldots,p\}.$$

Since each $\otimes$ in the mnp-element set corresponds to a scalar multiplication in producing A·B, we ask:
Is there a P-element set, P<mnp, of tensors

$$\mathcal{Z}' = \{X_r \otimes Y_r: r=1,2,\ldots P\}$$

which (by ±1,0 co-efficients) spans $\mathcal{R}'$ as well as $\mathcal{E}$?

Thm Existence of set ' above is equivalent to P-element set

$$\mathcal{Z} = \{[X_r][Y_r]: r=1,2,\ldots,P\}$$

of dyads, which span (using ±1,0 coefficients) the set

$$\mathcal{R} = \{E_{ik} \otimes \mathcal{U}_n: i=1,2,\ldots,m,\ k=1,2,\ldots,p\},$$

where E_{ik} is always an m×p matrix, and
$\mathcal{U}_n$ is the n×n identity matrix.

This formulation, using dyads (or rank-one matrices) of set $\mathcal{Z}$ to generate single tensors $E_{ik}\otimes\mathcal{U}_n$ of set $\mathcal{R}$ leads to recapture and extensions as follows:

Thm [Hopcroft-Musinski,Probert] If an (m,n,p) set-up (an m×n multiplying an n×p matrix) can be effected using P(m,n,p) scalar products. Then the same number of scalar products suffice for all permuted set-ups, i.e., (m,p,n), (p,n,m), etc.

Extension Consider the k-fold product $A_1 \cdot A_2 \ldots A_k$, i.e., a (m_0,m_1,m_2,m_k) set up (A_r is $m_{r-1}\times m_r$, r=1,2,...,k), Suppose the product entries are generated as follows (using k=3, for the product A·B·C for simplicity)

$$c_{i\ell} = \sum_{j,k} a_{ij}\cdot b_{jk}\cdot c_{k\ell} \longleftarrow \text{standard algorithm}$$

$$= \sum_{r=1}^{P} \alpha_r^{i\ell}\cdot a_r^{i\ell}\cdot b_r^{i\ell}\cdot c_r^{i\ell} \longleftarrow \text{new scheme}$$

where $a_r^{i\ell}$ is a ±1,0 linear combination of entries from A,
$b_r^{i\ell}$ is a ±1,0 " " " " " B,
$c_r^{i\ell}$ is a ±1,0 " " " " " C.
Then, as 2P scalar mult's are required ($2P < m_0m_1m_2m_3$) for the (m_0,m_1,m_2,m_3) set-up, so only 2P scalars suffice for any dihedral permutation, i.e. generated by shift

$(m_0,m_1,m_2,m_3) \longrightarrow (m_1,m_2,m_3,m_0)$, and reverse

$(m_0,m_1,m_2,m_3) \longrightarrow (m_3,m_2,m_1,m_0)$.

Thm [Hopcroft-Kerr] There is a scheme H-K on a (6,2,6) set-up where the number of scalar multns, H-K (6,2,6)=57.

Extension There is a scheme Q where Q(6,2,6) = 56
There is a scheme $\hat{Q}$ where (72,72,72) = $7^3 8^3$.

On languages recognizable by deterministic Turing machines within polynomial time

W. Erni, Heidelberg

It is well known that for the family of context-free languages L_{CF} one can prove $L_{CF} \subset L_{PTIME}$, the family of languages acceptable by deterministic multitape Turing acceptors in polynomial time. Furthermore we know that $L_{DSPACE(\log^2(n))}$, the family of languages acceptable by deterministic multitape Turing acceptors in space $\log^2(n)$. Now one may ask whether for "moderate" extensions of L_{CF}, e.g. those language families L generated by regulated rewriting systems, we may prove

(1*) $L \subset L_{PTIME}$ or (2*) $L \subset L_{DSPACE(\log^2(n))}$.

First we survey recent results. Then we use log(n)-space bounded nondeterministic auxiliary pushdown acceptors as a tool for proving property (1*) or (2*). We discuss the techniques if L is the family of languages accepted in linear time by pushdown acceptors with counters, and if L is the family of languages generated by nonterminal bounded context-free grammars with context-free control sets, and if L is the family of languages generated by deterministic L_{FIN}-iteration grammars with context-free control sets, and if L is the dhyper-algebraic extension of L_{CF}.

Analysis of two equivalence algorithms

Donald Knuth, Stanford

The algorithms "Quick Find" and "Quick Find Weighted" are considered with random input of three kinds. In the first model, all spanning trees of the graph are equally likely; in the second model, each pair of distinct equivalence classes is equally likely; in the third model, each pair of vertices in distinct equivalence classes is equally likely. The algorithms have qualitatively different behaviors in the different models, as shown in the following table:

	Model 1	Model 2	Model 3
Quick Find	$\sim \sqrt{\frac{\pi}{8}}\, n^{3/2}$	$\sim n \ln n$	$\sim \frac{1}{8} n^2$
Quick Find Weighted	$\sim \frac{1}{\pi} n \ln n$	$\sim \frac{1}{2} n \ln n$	$\sim \Theta(n)$

The analysis of Model 3 is of greatest interest, as it requires deep study of the connectivity of random graphs. This work was done jointly with Arnold Schönhage.

The analysis of some alternative path compression techniques*

Jan van Leeuwen, Utrecht

In the study of set-manipulation algorithms there has been considerable attention for the analysis of UNION-FIND programs. If the collapsing rule is used for FINDs and the weighted union rule for UNIONs then the time for n-1 UNIONs and m FINDs is bounded by $\mathcal{O}(m\alpha(m,n))$, where $\alpha(m,n)$ is an extremely slowly growing function (Tarjan, 1972). In the collapsing rule one must first climb up a tree to locate the root and then traverse the very same path a second time for attaching each of its nodes to the root. In his recent book on programming, Dijkstra independently presented the same technique, but he noted that the need for traversing a FIND-path twice makes it a rather unelegant method from the programmer's point of view. In an

attempt to find a more elegant implementation, we shall consider the problem of how the "second pass" can be eliminated without losing much of the original efficiency. We consider several alternative tree compression methods, including Rem's algorithm as presented by Dijkstra. We show that in worst case the algorithms are not as good as expected. We propose a simpler one-pass technique called "path-halving", which performs well in both the unbalanced and balanced cases. We show that the time needed for $\mathcal{O}(n)$ UNIONs and FINDs is bounded by $\mathcal{O}(n\log n)$ when only path halving is used. If the weighted union rule is used for UNIONs and path halving for FINDs, then the cost for $\mathcal{O}(n)$ UNIONs and FINDs reduced to $\mathcal{O}(n\log^* n)$. We conclude that path halving is a viable alternative to the original collapsing rule for all practical purposes. We note that a precise analysis like Tarjan's has failed to produce a better than $n\log^* n$ bound, and we currently conjecture that the bound is best possible for UNION-FIND programs using balancing and path-halving (perhaps even for all "one-pass" techniques).

* Footnote. This work was carried out jointly with Th. van der Weide, Department of Applied Mathematics, University of Leiden, Leiden, the Netherlands.

Space Bounds for Maze Threadability on Restricted Models

Stephen Cook (speaker) and Charles Rackoff, Toronto

Our motivation is to show NSPACE(logn)≠ DSPACE(logn) by showing the maze threadability problem is not DSPACE(logn). We prove the latter, but only for a restricted model of Turing machine. A d-maze M is a triple <G,s,g>, where G is a directed graph such that the edges leading out of each node have distinct labels from {1,2,...,d}, and s and g are nodes of G. M is threadable iff there is a directed path from s to g. A d-JAG J (Jumping Automaton for Graphs) is a machine with a deterministic finite state control with

q states which operates on an input maze ⟨G,s,g⟩ by moving p pebbles (or pointers) around the nodes of G. In one move, J either moves some pebbles i along some edge j, $1 \le j \le p$, $1 \le j \le d$, or jumps pebble i to the node on which pebble k is located, $1 \le i,k \le p$. The move depends on the current state, the coincidence partition on the pebbles, and which pebbles scan s or g. Initially pebble 1 is on g and all others are on s. We say J is valid for N if J correctly determines whether an arbitrary input d-maze of N or fewer nodes is threadable. We say the storage used by J is S=plogN+logq.

Theorem 1. $\forall N \exists J$ valid for N with storage $O(\log N)^2$)

Theorem 2. For $d \ge 2$, any J valid for N has storage $\Omega(\frac{(\log N)^2}{\log\log N})$

Theorem 3. For $d \ge 3$, no d-JAG correctly determines threadability on all undirected d-mazes (G,s,g) is undirected iff for every edge there is an edge in the opposite direction between the same two nodes).

Saving space in fast string matching

Zvi Galil, Tel Aviv

(Results are together with J. Seifezas at Penn State)

The string matching problem is to find all occurences of a pattern string x in a text string y. The naive algorithm has worst case time complexity proportional to $|x| \cdot |y|$ while the fast algorithm of Knuth-Morris and Pratt (KMP) requires only $O(|x|+|y|)$ time. On the other hand the naive algorithm requires only two additional memory location, but the KMP requires $|x|$ additional memory location in every case.

We improve both algorithms as follows: We derive a linear time algorithm that requires only $O(\log|x|)$ additional memory locations in the worst case (in the average four additional memory locations suffice). For every $\varepsilon > 0$ we derive a constant space algorithm that requires only $O(|x|\varepsilon \cdot |y|)$ in the worst case.

Complexity of the Satisfiability Problem for Propositional Dynamic Logic

Michael J. Fischer, Seattle

The propositional dynamic logic of regular programs, PDL, is a formal logical system derived from modal logic for reasoning about program schemes. The syntax consists of two sets: Σ, a set of programs and Φ, a set of formulas. Let Σ_0 be a set of uninterpreted statement symbols and Φ_0 a set of uninterpreted predicate symbols (propositional variables). Σ and Φ are defined inductively:

(i) $\Sigma_0 \cup \{\theta\} \subset \Sigma$, $\Phi_0 \cup \{\text{true},\text{false}\} \subset \Phi$.

(ii) If $a,b \in \Sigma$, $p,q \in \Phi$, then $a;b$, $a \cup b$, a^*, $p? \in \Sigma$ and $p \vee q$, $\neg p$, $\langle a \rangle p \in \Phi$.

Relational semantics are given by a structure $\mathfrak{A} = (W^{\mathfrak{A}}, \pi^{\mathfrak{A}}, p^{\mathfrak{A}})$, where $W^{\mathfrak{A}}$ is a set of worlds or states. $\pi^{\mathfrak{A}}: \Phi_0 \to 2^{W^{\mathfrak{A}}}$, and $p^{\mathfrak{A}}: \Sigma_0 \to 2^{W^{\mathfrak{A}} \times W^{\mathfrak{A}}}$. We extend p to Σ and π to Φ:

$$
\begin{array}{llll}
p(\theta) = \emptyset & & \pi(\text{true}) = W & \\
p(a;b) = p(a) \circ p(b) & & \pi(\text{false}) = \emptyset & \\
p(a \cup b) = p(a) \cup p(b) & & \pi(p \vee q) = \pi(p) \cup \pi(q) & \\
p(a^*) = p(a)^* & & \pi(\neg p) = W - \pi(p) & \\
p(p?) = \{(w,w) \mid w \in \pi(p)\} & & & \\
 & & \pi(\langle a \rangle p) = \{w \mid \exists v\ (w,v) \in p^{\mathfrak{A}}(a) \,\&\, v \in \pi^{\mathfrak{A}}(p)\}. &
\end{array}
$$

$(u,v) \in p^{\mathfrak{A}}(a)$ means some execution of a takes state u to state v. $w \in \pi^{\mathfrak{A}}(p)$ means p holds (is true) in state w. We write $\mathfrak{A},w \models p$ iff $w \in \pi^{\mathfrak{A}}(p)$. p is satisfiable if $\exists \mathfrak{A},w$ such that $\mathfrak{A},w \models p$.

Theorem 1 (Small model). If $p \in \Phi$ is satisfiable, then $\exists \mathfrak{A},w$ such that $\mathfrak{A},w \models p$ and size $\mathfrak{A} \leq 2^{\text{size } p}$.

Theorem 2 (Upper bound). The satisfiability problem for PDL is in NTIME(c^n) for some constant c.

Theorem 3 (Lower bound). $\exists\, c > 1$ such that the satisfiability problem for PDL is not DTIME(c^n).

Nondeterminism on Parallel Machines

Hanke Bremer, Frankfurt

I consider processors, which can compute one boolean function and which are connected like a tree. This tree-machine has a simple and fixed structure and has polynomial the same power than any other parallel-machine. If I allow to sach sub-automata (processor or node in the tree) a non-deterministic choice, then I get a more powerful and natural nondetermination, which does some effect even for parallel-machines. And in the corresponding class NP of the tree-automata there is a problem, which is complete and in which it is asked, whether there exist functions, which satisfy a given expression.

$$\exists f_1,\ldots,f_n\colon A(f_1,\ldots,f_n).$$

It is not to be seen, how to solve this problem even on a deterministic parallel-machine in polynomial time.

Two General Paradigms for Obtaining Lower and Upper Complexity Bounds

J.F. Traub, Pittsburgh

1. The lower bound paradigm is based on the pre-image of an "information" operator. It can be applied to any problem characterized by

A. The problem cannot be solved exactly with finite complexity.
B. Only certain "information" about a problem element is available.

Examples of such problems include optimization, differential equations, integration, and nonlinear equations, in a finite or infinite number of dimensions.

For such problems we construct an "information based theory" which permits the rational synthesis of algorithms. These algorithms are optimal or close to optimal with respect to error and complexity.

Theorems may be found in "General Theory of Optimal Error Algorithms and Analytic Complexity-- Part A: General Information Model" by J.F. Traub and H. Wozniakowski, Carnegie-Mellon University Report, Sept. 1977.

2. The upper bound paradigm is based on Newton iteration for "algebraic problems". Earlier Kung and Traub ("All Algebraic Functions Can be Computed Fast" CMU Report 1976) showed that with Newton iteration the first q terms of any algebraic function could be computed with complexity O(M(g)). Newton iteration has been extended to the multivariate non-commutative case over abstract rings. An application is generation of context-free languages from programs. Newton iteration is also widely applicable to develop power series solutions of nonlinear operator equations.

Some Results in Algebraic Complexity

Peter Schuster, Zürich

For some matrices rigidity in the sense of L.G. Valiant is computed (results of Bernhard Griesser). Some negative results of Alex Alder on autarchy for multiplicative complexity in polynomial rings are presented. The multiplicative complexity of the product, composition, continued fraction and quotient of polynomials with general coefficients is computed (results of Werner Hartmann). The multiplicative complexity of evaluating the d-th derivative of an interpolation-polynomial at the interpolation points is shown to have order of magnitude $n \cdot \log n$ (generalizing a result of V. Strassen). The complexity of a set of rational functions symmetric under a finite substitution-group G of the variables is estimated from below by $\log|G|$. The fixgroup of the structure-tensor of the k-algebra $k[x]/_f$, where $f \in k[X]$, is completely described (result of Peter Ritzmann).

A graph-theoretic property of computations

H.-J. Stoss, Konstanz

We investigate the computational complexity of functions $f = (f_1,\ldots,f_n): K^m \to K^n$ where K is an arbitrary set, Φ a set of operations in K and computations are defined as networks over Φ. From the usual definition of "f_j depends on the i-th variable x_i" follows that in each network for f at least

$$\gamma_f(o) := \#\{(i,j) \mid f_j \text{ depends on } x_i\}$$

pairs of inputs and outputs are connected.

We generalize this concept defining quantities $\gamma_f(k)$ $(k=0,1,..)$ s.t. in each network for f we can eliminate any k nodes and still $\gamma_f(k)$ pairs of inputs and outputs are connected. To define $\gamma_f(k)$ we have to investigate the restriction of f to subsets $V \subset K^m$ that belong to a suitable set $\mathfrak{V}_k \subset \text{pot } K^m$.

In some more special situations (linear-, boolean functions) is for almost all functions $\gamma_f(\frac{n}{2}) \geq \frac{1}{5} n^2$ and that shows using a theorem of Valiant that these functions have a complexity $\geq \varepsilon n \cdot \lg\lg n$.

Compromising between finite and infinite complexity

Dirk Siefkes, Berlin

Let WS1S be the monadic second order theory of $\langle \omega, < \rangle$.

Theorem: If R is elementary recursive and $k \in \mathbb{N}$, then $R \restriction t_k n$ is $O(m^k \cdot n)$-definable in WS1S. (Here $t_0 n := n$, $t_{k+1} n := 2^{t_k n}$.)

Thus for defining R in a given interval in an optimal way, the choice of k is crucial. A similar observation on programs, e.g. for multiplying two integers, leads to the following

Definition: A sequence $(A_k)_{k\in\Sigma^*}$ of algorithms computes a function f piecewise iff the following holds:

(i) Each A_k computes f.

(ii) There is a sequence $q_0 = \sigma < q_1 < \ldots$ s.t. A_k is optimal in the interval $[q_k, q_{k+1})$ among the A_j.

(iii) Both A_k and q_k are computable from k.

Measuring the complexity of functions using piecewise computations compromises quite promisingly between the drawbacks of finite and infinite complexity.

Data representation and computational complexity

Klaus Weihrauch, Aachen
(and Rutger Verbeek)

A reasonable representation of a set M by eg. numbers should satisfy some natural requirements. It is studied for several requirements whether they are satisfyable or not. Given any complexity class K, there is some function on M which is computable for some bijective representation, but which is not computable in K for any bijective representation. This is no longer true for numberings of M, however, a powerful numbering must have a difficult equivalence problem. A similar statement holds for gödelizations. There are two functions, each of which can be computed easily using appropriate gödelizations, but for no gödelization both of them become easily computable. Finally it is shown that for no pair γ, γ' of gödelizations the γ'-primitive-recursive functions properly include the γ-primitive-recursive functions.

Hierarchy results for unary languages

Burkhard Monien, Paderborn

We show that for the classes of languages over a one-letter alphabet

DH(k), NH(k), DC(k), NC(k), DR(k), NR(k)

defined by deterministic or nondeterministic k-head automata, linear bounded k-counter machines and linear bounded k-register machines the following hierarchy results hold:

$$DH(k) \subsetneq DH(k+1),\quad NH(k) \subsetneq NH(k+1)$$

$$DC(k) \subsetneq DC(k+1),\quad NC(k) \subsetneq NC(k+2)$$

$$DR(k) \subsetneq DR(k+2),\quad DR(k) \subsetneq DR(k+2)\ .$$

Pebbling the FFT Graph and the Integer Multiplication Function

John E. Savage, Providence
(and Sourmitri Swamy)

The performance of the FFT algorithms is examined under limitations on computational space and time. It is shown that if the algorithms with n inputs, n a power of 2, is implemented with S temporary storage locations where $S = O(n/\log n)$, then the computation time T grows faster than n log n. Furthermore, T can grow as fast as n^2 if $S = S_{min}+O(1)$ where $S_{min} = 1+\log_2 n$, the minimum storage necessary. These results are obtained by deriving tight bounds on T versus S.

While the above results are derived for a particular algorithm for the discrete Fourier Transform, we show that any straight-line algorithm for the multiplication of two binary numbers with the result in binary must satisfy $ST \geq \Omega(n^{3/2})$. This is derived using a method introduced by Grigoryev.

More on Pebbles

Allan Borodin, Toronto

We consider the pebbling game on arithmetic circuits (i.e. the gates are +, -, *, scalar multiplication) viewed as directed graphs. The rules are that a pebble may be placed on a node if all the operands are presently pebbled

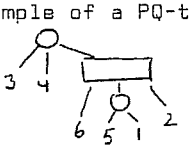
(in particular, an input may always be pebbled), and that any pebble may be removed from the graph. The goal is to have pebbled all outputs in any order. We measure the "space" S, the maximum number of pebbles on the graph during the game, and the "time" T, the total number of pebble placements. It is shown that any n-super concentrator graph require $T \geq n^2/s^2$ (roughly speaking). Consequently, discrete Fourier transform of prime order n, require $T = \Omega(n^2/s^2)$. It is also shown that any (f(s),n,n) grate (in the sense of Valiant) require $T \geq n^2/s$, when $f(s)=(n-s)^2$ or $f(s)=n(n-s)$. Consequently "almost all" linear functions (defined by $A_{n\times n} \vec{x}_{n\times 1}$) require $T = \Omega(n^2/s)$ and moreover, the FFT graph of order $n = 2^r$ require $T = \Omega(n^2/s)$.

Implementation of PQ-Tree Algorithms

Richard E. Ladner, Seattle
(with Michael J. Fischer and Sarah M. Young)

PQ-trees originally defined by Booth and Lucker are data structures which they used to solve in linear time the consecutive ordering problem, given $S_1,\ldots,S_k \subseteq V$, find a linear ordering of V in which the elements of each S_i form an interval.

An example of a PQ-tree



P-node: Permutations of children allowed

Q-node: Reversal of children allowed.

We give a new design for the main PQ-tree algorithm. The algorithm is linear time and has a form which is easy to prove correct. We provide details of our PASCAL implementation of the algorithm.

On the Modulo p Factorizations of an Integral Polynomial

George E. Collins, Madison

The degrees of the irreducible factors of a polynomial A of degree n constitute a partition of the integer n, denoted by part (A). Let Φ_p be the homomorphism from Z onto the prime finite field GF(p). Let p_i be the $i^{\underline{th}}$ prime, A an integral polynomial of degree n, π_i = part(Φ_{p_i}(A)). For any partition π of n, let P(π) be the known probability than the lengths of the disjoint cycles of an n-permutation constitute the partition π. Let P(π,A) be the limit probability, if it exists that a term of the sequence $(\pi_1,\pi_2,\ldots)$ is π. A is normal in case, for every n-partition π, P(π,A) = P(π).

It is conjectured that almost all integral polynomial of degree n, for each fixed n, are normal. This means that if S_d is the set of all polynomials of degree n with coefficients in [-d,d] then the probability that an element of d is normal approaches 1 as $d\to\infty$. The conjecture is proved for n = 2. Some theorems and computations support the conjecture for n=3 and n=4, In fact, we are led to two stronger conjectures: (I) If the degree of A is n and the order of the Galois group of A is n! then A is normal; (II) Almost all integral polynomials of degree n have Galois groups of order n! Applications of normality to computing part (A), and thereby the complete factorization of A, are discussed.

Fast Quantifier Elimination for Algebraically Closed Fields

Joos Heintz, Zürich
together with Ruedi Wüthrich, Zürich

Theorem: Quantifier elimination of prenex formulas in first order theories of algebraically closed fields of arbitrary characteristic is possible in time $d^{(cn)^n}$ where

d = sum of the degrees of the polynomials appearing in the formula

n = total number of variables in the formula

c>o some constant.

This implies, that a formula without free variables is decidable in time $2^{2^{cL}}$, where L is the length of the formula.

In connection with this theorem the following question arises: What is the largest cardiality of a finite subset of an affine space definable by a formula of given length? We have the following

Theorem: Let k be an algebraically closed field of arbitrary characteristic and $Y \subset k^n$ finite, definable by a first order formula of length L. Then

$$\# Y \leq 2^{2^{cL}}.$$

Complexity of algebraic numbers

Maurice Mignotte, Strasbourg

We try to give a reasonalbe definition of the complexity of a given algebraic number. We study the different conditions to be satisfied by such a notion. We show the link between the complexity of an algebraic number and the cost necessary to obtain an approximate value of this number. Different possible definitions are studied and discussed.

Specialization of the multiplicative complexity function

M. Sieveking, Frankfurt

Let $L_K(\underline{z}) = L_K(*/,\{f_1(\underline{z}),\ldots,f_r(\underline{z})\})$ be the number of multiplications/divisions required to evaluate (compute) functions f_i of the form

$$f_i(\underline{z}) = \frac{a_0(\underline{z})+\ldots+a_n(\underline{z})x^n}{1+b_1(\underline{z})x+\ldots+b_n(\underline{z})x^n} \qquad 1 \leq i \leq r$$

where $\underline{z} = (z_1,\ldots,z_\ell)$, $a_i,b_j \in k_o[z_1,\ldots,z_\ell]$, z_i algebraically independent over k_o and K is an algebraically closed field containing a field $k(z_1,\ldots,z_\ell)$ and k_o is the prime field of k. The computation may use the operations +,-,*,/ and constants from $x \cup k$. x is supposed to be algebraically independent over K. Similarly we define

$$L_k(\underline{c}) = L_k(*/,\{f_1(\underline{c}),\ldots,f_r(\underline{c})\}) \qquad \underline{c} \in k^\ell$$

Theorem There is a natural number γ with the following property: If any polynomial $H \in k_o[z_1,\ldots,z_\ell]\setminus\{0\}$ with $H(\underline{c}) = 0$ has a degree $> \gamma$ then

$$L_K(\underline{z}) = L_k(\underline{c}) \qquad (\underline{c} \in k^\ell)$$

It suffices to take

$$\gamma = 2^{L_K(\underline{z})\cdot 2^{L_K(\underline{z})}} \cdot \deg \operatorname{graph} \varphi$$

where φ is a mapping $k^\ell \longrightarrow k^{r(2d+1)}$ which has the first 2d coefficients of the powerseries expansion of $f_1,\ldots,f_r$ as components and $d = \max\{\deg_x f_i \mid 1 \leq i \leq r\}$.

Remark If r=1 and $\underline{c} \in k^\ell$ is such that

$$L_k(\underline{c}) \leq L_k(*/,f_1(\underline{c})+x^{2d+1}g) \text{ for all } g \in k((x))$$

then a smaller γ is sufficient, namely

$$\gamma = 2^{L_K(\underline{z})(2d+1)} \cdot \deg \operatorname{graph} \varphi.$$

Non-Deterministic Polynomial Optimization Problems and their Approximation

A. Paz, Haifa

NP problems are considered as recognition problems. It is first shown that every NP problem can be represented as an optimization problem. We then develop a new theory of non-deterministic polynomial optimization problems (NPOP's).

NPOP's are classified and studied with regard to the possibility or impossibility of "reducing" certain types of NPOP's to other types, in a sense specified in the text. A "complete" NPOP is shown to exist thus generalizing Cook's theorem for NPOP's. Finally, approximations of NPOP's are studied. Necessary conditions and sufficient conditions for approximability are given and it is shown that the known approximability results fit within the general frame developed in this paper.

The Influence of the Machine Model on the Time Complexity of Context-Free Language Recognition

Reinhold Weicker, Erlangen

It is shown that the assumption of a RAM model with "unit cost criterion" in the case of context-free language recognition leads to time bounds significantly better than the well-known upper bounds of $O(n^3)$ or $O(n^{2.81})$. First, Earley's algorithm for general context-free language-recognition is modified in a way such that it makes use of the array structure of the recognition matrix. This leads to a formulation of the algorithm with steps "Union" (of non-disjoint sets) and "Get next element" as basic steps. Two models of a RAM are considered, a "BOOLRAM" with Boolean operations and a one-bit shift operation, and the normal "PLUSRAM" with addition and proper subtraction operations. It is shown that, if we assume a unit cost criterion, the algorithm can be implemented on a BOOLRAM with time bound $O(n^2)$ and on a PLUSRAM with time bound $O(n^2 \log n)$.

Progress on priority deques

P. van Emde Boas, Amsterdam

In 1975 we presented at the FOCX 16 meeting in Berkeley a data structure for set-manipulation on a fixed universe $U = \{1,\ldots,n\}$ in which all instructions on a single ordered set, i.e. inserting elements, deleting elements, testing membership, computing least or largest elements, and computing predecessors and successors, all take time $O(\log\log n)$ for each element processed.

The structure as presented (see also [1]), uses space $O(n \log\log n)$ RAM-words. In the meantime the overhead factor $\log\log n$ in the space requirements has been eliminated. Another problem is that the structure as described does not correspond to the structure obtained by unwinding the recursion from a recursive explanation of the basic idea. In the talk I will explain what the structure should look like in its most simplified form.

1. P. van Emde Boas, R. Kaas & E. Zijlstra, Design and implementation of an efficient priority queue, Math. Systems Theory 10(1977) 99-127.

2. P. van Emde Boas, Preserving order in a forest in less than logarithmic time and linear space, Information Processing Letters 6(1977) 80-82.

V. Strassen (Zürich)