MATHEMATISCHES FORSCHUNGSINSTITUT OBERWOLFACH

T A G U N G S B E R I C H T 7/81

Effiziente Algorithmen

1.2. bis 6.2.1981

Die Tagung fand statt unter Leitung von Herrn Mehlhorn (Saarbrücken) und Herr Walter (Darmstadt). Ein erster thematischer Schwerpunkt waren Probleme, die "VLSI-Design" betrafen und damit zusammenhängende "Rechteckprobleme" (Measure-, Intersectionproblem etc.). Ein zweites häufig behandeltes Thema war "Parallel Computation", dabei besonders Kommunikation und Synchronisation. Eine dritte wichtige Gruppe waren schließlich Datenstrukturen. Es wurden neue vorgestellt (k-dimensionale B-Bäume, "treap") und bekannte auf bestimmte Eigenschaften untersucht. Die übrigen Vorträge verteilten sich auf verschiedenste Themen über Effiziente Algorithmen, so daß insgesamt ein recht breites Spektrum behandelt wurde.

Vortragsauszüge (in zeitlicher Reihenfolge)

J. VAN LEEUWEN:

Stratified Balanced Trees

In the last few years the number of different techniques for balancing search trees has been steadily increasing. We present a new perspective on many of the known classes of balanced trees, showing that nearly all contain closed sub-classes of "stratified" trees which can be maintained by one master algorithm. The work, carried out with M.H. Overmars, shows yet another step towards a "calculus of data structures".

R. KEMP:

On the Average Oscillation of a Stack

The oscillation of a stack can be described by a function f where f(t) is the number of nodes in the stack after t units of time. The maxima (MAX-turns) and minima (MIN-turns) of the function f correspond to the points where the length of the stack is reversed.

First several new enumeration results concerning planted plane trees are derived, e.g. the number of all n-node trees with λ leaves and a root of degree m. Next, it is shown, that the average number of MAX-turns is n/2; the variance B given by n(n-2)/(8n-12). Finally, we prove that (for large n and fixed j) the average number $\Delta_n(j)$ ($\nabla_n(j)$) of nodes stored in the stack between the j-th MIN-turn and the (j+1)-th MAX-turn (are removed from the stack between the j-th MAX-turn and the j-th MIN-turn) is given by

$$\Delta_n(j) = 8/3 - 8/9\ \theta(j) + O(n^{-1/2}) \text{ and } \nabla_n(j) = 4/3 + 8/9\ \theta(j) + O(n^{-1/2})$$

where $\theta(j) = \sum_{1\leq\lambda\leq j-1} [\lambda(\lambda+1)]^{-1} 3^{-\lambda} P'_\lambda(5/3)$ and $P'_\lambda(5/3)$ is the first derivative of the λ-th Legendre polynomial taken in the point 5/3. This fact implies several properties of the average oscillation of a stack.

I. WEGENER:

Algorithms for the solution of search problems

An oblivious person looses again and again his glasses in his own house. In order to serach optimally, that means to minimize the expected time of search, he estimates the probability p(k), that he has lost the glasses in room k, the probability g(j,k) of finding the glasses exactly during the i-th search of room k, if the glasses are in this room, and the cost c(k,k) of this search. We discuss more important situations where this search model is adequate. Afterwards we present an efficient algorithm for the construction of good and optimal strategies. Furthermore, we discuss six more general search models where we discuss in detail the problem of maximizing the probability of success, if one is allowed to spend not more than effort L, and the search model where we consider also the cost of changing from k to k'.

D. WOOD:

Rectangle Problems

There has been a great deal of interest in rectangle problems in the last few years, one of which is the measure problem, namely compute the area covered by the set of rectangles. Others are the intersection problem, the connectivity problem, etc.. In each of these new data structures have been introduced to solve them in either a static or dynamic manner. The aim of our current work (with Gaston Connet and Ian Munro, University of Waterloo) is to re-examine these problems with two aims in mind. The first, for which no progress has yet been made, is that of finding lower bounds for these problems. The second, which we discussed, is to use simple well known data structures for these problems, if at all possible! We are able to solve the 2-dimensional measure problem in O(n log n) time and O(n) space using the sweeping line method. Our contribution is to reduce it to O(n) space and to use an AVL- or weight-balanced tree for the 1-dimensional dynamic measure problem that is obtained. Second, we present a f-dimensional dynamic solution to the point stubbing problem, taking O(log n) time for insert, delete and query and O(n) space. This leaves open other 1-dimensional dynamic problems; which of them can be solved by using standard balanced search trees ?

Chr. STRELEN:

A Minimal Property and its Application to Tries

We consider files with retrieval on secondary keys. The records are stored in buckets. Our concern is to keep the number of storage accesses low when retrieving. This is done by a proper arrangement of the records into the buckets. We give a minimal property for this problem. A trie scheme based on the property is proposed. It provides low storage overhead, and efficient updates. The trie can be used to retrieve the records. Here the needed effort for searching the records decreases when the number of specified keys grows.

Th. OTTMANN:

A comparison of iterative and defined classes of search trees

Classes of search trees can be defined in a purely static way by imposing constraints on the structure of trees belonging to them. On the other hand, we define a class of trees in a dynamic way as the smallest class which contains the empty tree and which is closed under insertions. We discuss the question whether or not these two definitions coinade for various classes of search trees and for top-down versus bottom-up insertion schemes. We, in particular, relate 2-3 trees and 2-3-4 trees and their update schemes to notations for natural numbers. Finally, we present purely top-down update schemes for brother trees which carry over to AVL trees, if an propriate routing scheme is used.

H.-P. KRIEGEL:

Multidimensional B-trees

A data structure, called k-dimensional B-tree of order d, $d \geq 1$, is presented which allows each of the operations k-dimensional exact match query (ACCESS), INSERT and DELETE to be performed in time $O(\log_{(d+1)} N+k)$ in the worst case, where N is the number of k-dimensional records in the file.

Above structure can be adapted to the case of time-varying access frequencies. In case k = 1, we suggest weighted 2B-trees of order d for weighted dynamic dictionaries with the following properties:

1. The operation ACCESS key x_i can be performed in time $O(\log_{d+1} W/w_i)$ i.e. the tree is always nearly optimal. Here w_i is the weight (number of accesses) of key x_i and $W = \sum_{i=1}^{N} w_i$.
2. The operation PROMOTE, DEMOTE and SPLIT can be performed in ACCESS time.
3. We can CONCATENATE two trees in time $O(\log_{d+1} W_1)$ when W_1 is the total weight of the heavier tree.

In case of k-dimensional keys a factor (k-1) has to be added to the time complexities.

C.P. SCHNORR:

Refined Analysis and Improvements on Some Factoring Algorithms

By combining the principles of factoring algorithms we obtain some improved algorithms which by heuristic arguments all have a time bound $O(\exp\sqrt{c \cdot \ln n \cdot \ln \ln n})$ for various constants $c \geq 3$. In particular, Miller's method of solving index equations and Shanks method of computing ambiguous quadratic forms with determinant -n can be modified in this way. We show how to speed up the factorization of n by using preprocessed lists of those numbers in [-u,u] and [n-u,n+u], $0 \ll u \ll n$ which only have small prime factors. These lists can be uniformly used for the factorization of all numbers in [n-u,n+u]. Given these lists, factorization takes $O(\exp[2(\ln n)^{1/3}(\ln \ln n)^{2/3}])$ steps. We slightly improve Dixon's rigorous analysis of his Monte Carlo factoring algorithm. We prove that this algorithm with probability 1/2 detects a proper factor of every composite n within $o(\exp\sqrt{\beta \ln n \ln \ln n})$ steps.

E. McCREIGHT:

Treap Husbandry

Treaps (tree-heap) are defined as a family of data structures that index a dynamic set of ordered pairs (x,y) by means of a search tree in x adjoined to an ordered heap in y. Treaps require space linear in the number of pairs.

In a simple radix (or static) treap containing n pairs, $O(\log|X| + \log n)$ time is required to insert a new pair or delete an old one. Given three numbers x_0^*, x_1^* and y_0^*, all the pairs (x,y) that satisfy the conditions $x_0^* \leq x \leq x_1^*$ and $y_0^* \leq y$ can be enumerated in time $O(\log|X| + \log n + s)$, where s is the number of enumerated pairs.

Using a treap, a dynamic set of linear intervals [x,y] can be maintained in linear space and logarithmic time. With this representation, those intervals from the set that totally contain a test interval [x*,y*] can be enumerated in logarithmic time beyond that which is required for enumeration. This is a new result.

There is also a more complicated comparative (dynamic) treap form based on 2-3-4 search trees (or perhaps on the recent search trees of Oliviê of Antwerp). With this form the $\log|X|$ term above disappears (in exchange for larger constant factors).

G. AUSIELLO:

Probabilistic Analysis of the Performance of Greedy-Like Algorithms for some Classes of Max-Subset Problems

Maximization problems over power sets are introduced with the aim of characterizing the approximability of combinatorial problems. The characterization of classes of combinatorial problems which can be fully approximated is studied both from the point of view of worst case behaviour and from the point of view of probabilistic analysis.

The probabilistic analysis of the behaviour of classical greedy algorithms for clique and other combinatorially similar problems, which can be found in the literature, is extended to a large class of non-fully approximable max-subset problems.

On the other side conditions for the full approximability are introduced and the behaviour of greedy algorithms over random instances is studied.

J.E. SAVAGE:

Planar Combinatorial Complexity and Tradeoffs for VLSI Algorithms for One-output and Multi-output Functions

The computation of functions using very large scale integrated circuits (VLSI) offers interesting tradeoffs between the area A of the chips and computation time T. In this talk we demonstrate that area and time satisfy $AT^2 = \Omega(C_p(f))$, where f is the function to be computed. We use this relation to prove that the one-output function which recognizes whether two n-bit binary numbers are reciprocals of one another or not requires $AT^2 = \Omega(n^2)$. We also comment on the realization of matrix multiplication through VLSI.

Th.-D. HUYNH:

Complexity of the Equivalence Problem for Commutative Grammars

In this paper we define the notion of commutative grammars and consider the complexity of two basic decision problems : the uniform word problem and the equivalence problem for various commutative grammar classes.

W. PAUL:

On Parallel Computers

We consider the problem how to interconnect many processors in order to get an efficient general purpose parallel computer and show:

i) every computation can be spelled up asymptotically by massive use of parallelism.

ii) some interconnection patterns are good for some problems and very bad for others

iii) in a certain sense every interconnection pattern is bad for some problem

iv) there is an interconnection pattern which is reasonably good for all problems

L.G. VALIANT:

Is General Purpose Parallel Computing Possible ?

An idealistic parallel computation is one that can be realised on an N-processor machine with arbitrary simultaneous access to a shared common memory (but avoiding fetch or store conflicts). A realistic N-processor computer is one having one processor at each node of an N-node network of small degree (e.g. constant, or log N). We show that there is a realistic N-processor computer that can realize any idealistic N-processor computation with only a log N multiplicative factor loss in runtime. To do this we give a distributed parallel algorithm for achieving the following in O(log N) time : given one packet at each node of an appropriate low degree network (e.g. n-dimensional cube, d-way shuffle) with a distinct destination node written on each, the algorithm has to route each packet to its destination without two packets ever passing down any one wire at the same time.

N. BLUM:

An $\Omega(n^{4/3})$ lower bound on the monotone complexity of the convolution

Let $A = \{a_0,\ldots,a_{n-1}\}$, $B = \{b_0,\ldots,b_{n-1}\}$ be two sets of n variables. Then we define :

$$c_k = \bigvee_{\nu+\mu=k} a_\nu \wedge b_\mu \qquad \text{for } 0 \le k \le 2n-2$$

$$C_n = \{c_k \mid 0 \le k \le 2n-2\}$$

C_n is called the n-th degree convolution.
By applying a new method we prove an $\Omega(n^{4/3})$ lower bound on the monotone complexity of convolution. The best lower bound known so far was $\Omega(n \log n)$.

N.D. JONES:

Flow Analysis of Lambda Expressions

Program flow analysis consists of a static analysis of a program's text with the purpose of deducing its dynamic or runtime properties. The main application has been program optimization, but flow analysis may also be used in program verification, program certification and debugging.

Satisfactory methods have been developped for flow analysis of flowchart programs; these mostly involve mapping program points into approximation lattices which provide finite descriptions of the sets of data values the variables may assume whenever control reaches a given point. However, the flow analysis of programs with procedures has not been completely successful.

The purpose of this talk is to describe a new method for interprocedural analysis which is capable of handling programs with recursion, call by name or call by value and procedures both as arguments to procedures and as the values returned by them. The method is applied to the lambda calculus as a "worst-case" example but may also be used for more conventional imperative languages.

H. EDELSBRUNNER:

Recent results on geometric intersection searching

Problems dealing with intersections of geometric objects constitute one of the important branches in computational geometry. However, only recently searching problems involving such sets where the desired answer consists of all objects intersecting a query object have been considered. In the meantime, a number of remarkable results have been obtained:

The important special case dealing with axes-parallel objects only has been largely solved. Most recent research has pushed forward our knowledge about intersection searching in a more general setting: A set of simple polygons, each of which contains at most a constant number of edges, can be preprocessed to allow for a query to be answered in time proportional to the logarithm of the number of objects involved.

Ph. FLAJOLET:

The height of trees and other products

The average case analysis of an algorithm requires a detailed counting of its input configurations together with a weighting of costly/noncostly configurations. There are many cases where there are no closed form expressions for these counting problems. A possible solution is provided by a singularity analysis of generating functions. Several translation lemmas can be set up (having the flavour of the classical Darbour-Polya method) that relate local properties of generating functions around singularities to the asymptoties of their coefficients. A typical application is provided by the analysis of some simple recursive tree algorithms, with a diversity of statistics on the inputs. Examples include : the height of tries (corresponding to the analysis of tree traversal), the occurrence of patterns and the analysis of tree matching, formal differentiation algorithms, algorithms for embedding trees in trees. The framework makes it possible to derive some general observations that relate structural and computational properties of programming [based on joint work with A. Odlyzko (USA) and J.-M. Steyaert (F)].

B. COMMENTZ-WALTER

Efficient Algorithms in Software Development

For several algorithmic proclems there are wellknown algorithms described in literature which are asymptotically efficient. Unfortunately, in case of operating systems, the algorithmic real life problems are unlike these wellknown ones of literature. Hence it is not possible to use these algorithms, But, nevertheless, the ideas underlaing these algorithms are very useful. We illustrate this by some examples.

I. MUNRO:

On heaps

Heaps are a popular implementation of priority queues. Traditional insertion and deletemin algorithms take 2 log n comparisons. It is shown that log n + log*n + O(1) comparisons are necessary and sufficient.

M. FONTET:

Determination of Imprimitives of a permutation group

In recent years, several polynomial algorithms have been designed to solve graph isomorphism problems for special classes of graphs. One technique proposed by the author to design such algorithms consists in associating to a graph its combinatorial maps. Thus, the original problem is reduced to the determination of particular imprimitivities of certain permutation groups given by their generators. We show here how these results can be extended to other classes of imprimitivities to get efficient algorithms for determining them.

E. MEINECHE SCHMIDT:

Different broadcasting primitives in a parallel environment

Two different notions of broadcasting are introduced, one is based on Milner's CCS-formalism and the other on Hoare's CSP formalism. In CCS-broadcasting a set of processes, driven by a common clock, communicate on a number of "frequencies" in such a way that a sender in one step sends a signal to everybody who is listening on this frequency. In CSP-broadcasting, on the other hand, the handshake between sender and receivers is preserved, hence time becomes a derived concept which increases because things happen. Both notions of broadcasting are defined in terms of simple automata based models and their difference is illustrated by simple examples.

E. SHAMIR

Random Graphs Algorithms

We concentrate on the construction of a Hamiltonian path through all the vertices of a graph.
Let $G_{n,p}$ denote the probability space of random graphs over n labelled vertices, where the $\binom{n}{2}$ edge-occurrences are independent random variables with probability p for occurring.

We take $p = c\frac{\log n}{n}$ and study the limit probabilities of events as $n \to \infty$.

Posa proved that for $c \geq 3$ the graph is Hamiltonian with probability 1. Clearly $c > 1$ is necessary. What is the precise threshold ? We analyse and amplify Posa's argument using rotations of partial path and prove that $c = 1+\varepsilon$ assures a Hamilton path.

Algorithms to construct the path have random inputs, the adjacency lists of a random graph. They can proceed deterministically in several ways, based on scanning the lists, effecting path extension if possible, else rotation and FAIL if a list is exhausted.

We give a precise stochastic analysis of the histogram of the lists' lengths, showing that, say for $c > 2$, the algorithms does not fail before it scanned $(1+\varepsilon)n \cdot \log n$ independent edges, which suffice to construct a path with probability $1 - O(1)$.

J.L. BENTLEY:

A Case Study in Applied Algorithm Design : VLSI Artwork Analysis

The field of Applied Algorithm Design is concerned with applying the results and methods of Analysis of Algorithms to the real problems faced by practitioners of computing. In this lecture we will see how Applied Algorithm Design has been used in the construction of a VLSI design system. The primary algorithmic content of this talk deals with optimal worst-case algorithms, optimal expected-time algorithms , and NP-completeness results for a set of problems. We will also study such issues as the satistical inference of the distribution of inputs, the simplification from the real problem to the abstract problem, and the implementation of the algorithms in a real design system.

R. GOTTLER:

Generating optimal code for series-parallel graphs

The problem of generating optimal code for DAG's, that represent expressions, is easily solvable for trees but NP-complete in the general case. So it is reasonable to look for classes of DAG's more general than trees such that generation of optimal code can be done in polynomial time. It is shown here, that series-parallel graphs are such a class of DAG's. The process of code generation is just similar to that used by Sethi & Ullman for trees, i.e. first each node is labelled with the minimal number of registers needed to compute this node without store-instructions and then this information is used for code generation.

J. FRANCON:

Sequence of Operations Analysis

The purpose of the talk is to describe a rather general method for computing the average cost of sequences of insert, delete, and search operations in a dynamic data structure. This method is applicable to many of the interesting known implementations of data structures. It allows one to perform quantitative comparisons among various data organizations (i.e. data structures together with associated algorithms for operating on these structures).

C.H. PAPADIMITRIOU

Analysis of Greedy Heuristics for Planar Graphs

We present a technique for analyzing the performance of greedy heuristics when applied to planar graphs. The technique involves induction on the faces of a planar representation, and it appears to help derive both upper and lower bounds. Using these ideas we show that, for a planar graph with n nodes

(a) The greedy heuristic yields an independent set of size at least n/5.25

(b) The maximum independent set is at most 2n/3, for graphs of minimum degree three or more.

(c) The greedy heuristic yields an independent set at least 11/30 times the optimum.

These bounds are the best possible.

J. NIEVERGELT:

Generality of plane-sweep algorithms

Plane-sweep algorithms have emerged as an important tool in computational geometry. Their domain of application has gradually expanded from dealing with unstructured sets of straight-line segments, to sets of lines structured into objects such as rectangles, polygons, and graphs embedded in the plane; and from computing line intersections to answering many topological and geometric questions.

We show that a general purpose plane-sweep algorithm can answer all the following types of questions in time O((n+s)log n) and space O((n+s)), where n is the total number of line segments and s is the total number of line intersections:

Gefördert durch DFG Deutsche Forschungsgemeinschaft

Enclosure, adjacency, connectivity, winding numbers, maximal and minimal width, areas and perimeters of regions. Under special circumstances time can be compressed to $O(n \log n + s)$ and space to $O(n + s)$.

B. MONIEN:

Approximation algorithms for feedback vertex problems

We define approximation algorithms A_1 and A_2 to solve the feed back vertex set problem for undirected and for directed graphs, respectively. We show

1.) G undirected

$$\Rightarrow |F_{A_1}(G)| \leq 2|1+o(1))\cdot|F_{OPT}(G)|\cdot\log|F_{OPT}(G)|$$

2.) G directed

$$\Rightarrow |F_{A_2}(G)| \leq 2\sqrt{2|V|}\cdot|F_{OPT}(G)|$$

where $F_{A_i}(G)$, $i = 1,2$, is the set of nodes computed by algorithm A_i and $F_{OPT}(G)$ is the optimal solution.

(Co-authors: R. Schulz and K. May).

A.O. SLISENKO:

Context-free grammars as a tool for describing polynomial time subclasses of NP-complete problems

I would like to draw attention to context-free generation, which may serve as a tool for adequate mathematical treatment of practically interesting subclasses of "hard" problems subclasses that can be solved in polynomial time. The idea will be exposed for the Hamiltonian circuit problem in planar graphs which is known to be NP-complete. Let us consider context-free graph grammars with productions permitting only substitutions of planar graphs instead of whole "star", consisting of a node and all semi-edges adjacent to it. If a grammar G is fixed then the degree of graphs in L(G) is bounded by a constant. This restriction allows to apply standard parsing technique and get polynomial-time parsing algorithm.

Theorem: For any context-free graph grammar G of the described type the Hamiltonian circuit problem for graphs in L(G) can be solved in polynomial time.

P. van EMDE BOAS

Integer LP in bounded dimension is polynomial

Hendrik W. Lenstra, jr. from Amsterdam has shown that the following problem ILP_k is polynomially solvable for each fixed k :
given a convex polytope K in $\mathbb{R}^k$ defined by rational inequalities
question does K contain a point with integral coordinates ?

The result establishes that the complexity of the general Integer Linear Programming problem, which is known to be NP complete is rooted in the number of variables rather than the number of constaints or the size of the coefficients of the problem.

The algorithm uses a reduction algorithm for lattice bases in $\mathbb{R}^k$ which transforms an arbitrary distorted basis for a lattice into another one which is somewhat orthogonal. This algorithm is used upon the image of $\mathbb{Z}^k$ under a linear transformation τ designed in order to transform the given polytope into a "round" one, having the property that its boundary is included between two spheres whose radii differ by a factor $c_1(k)$ dependent of the dimension only. From the reduction algorithm one obtains that the lattice can be decomposed into parallel layers $L'+n.b_1$, such that the hyperplanes containing these layers have a distance β satisfying $r'/\beta \leq c_2(k)$ for some constant $c_2(k)$ depending on k only, where r' is the covering radius of the lattice L. Now either the radius of the small sphere is less than r' and we can reduce the problem to at most $c_1(k).c_2(k)$ lower-dimensional subproblems, or the small sphere contains certainly a lattice point and the algorithm answers "yes". Analysis give polynomial bounds which are very crude.

M. FISHER:

Using Clocks to Improve the Efficiency of Distributed Algorithms

Consider a system of m asynchronous processes and n lights, each controlled by a distinct process ($m \geq n$). Processes communicate via shared variables, each of which can be accessed by at most b processes. A session is any time interval during which every light flashes at least once. A system state is quiescent if no further light flashes occur. A round is any time interval during which every process takes at least one step. The time to quiescence is the largest number of rounds the system can run before reaching a quiescent state.

Definition: The (s,n)-session problem is to find an n-light system which produces at least s sessions and then becomes quiescent in every computation.

Theorem 1: Every solution to the (s,n)-session problem has time to quiescence at least $(s-1) \log_b n$.

Theorem 2: There is a solution to the (s,n)-session problem with time to quiescence at most $O(s \log n)$.

Theorem 3: If each process is augmented with a local clock whose rate of drift is bounded and known in advance, then there is a solution to the (s,n)-session problem with time to quiescence only $O(s + \log n)$.

(Co-authors: E. Arjomandi and N. Lynch)

F. MEYER AUF DER HEIDE:

Efficiency of universal parallel computers

We consider parallel computers (PC's) with a finite number of processors. The capability of communication is described by a graph with bounded degree.

For several kinds of simulations among PC's we prove that it is not possible to simulate all PC's with n processors by one (i.e. a universal PC) without a significant additional expense of time or of processors.

Then we present a universal PC whose graph is a "permuter", which needs $n^2\log(n)$ processors, and extends the time only by a factor log log(n). We prove that no universal PC whose graph is a permuter can be faster.

A. ZACHARIOU:

On certain number theoretic algorithms

A close study of Euclid's proof for the existence of infinitely many primes leads to Theorem 1: For any odd prime p there exist infinitely may odd primes q such that p is the least prime divisor of 2q+1. Corollary: There exist infinitely many primes q for which 2q+1 is composite. Such primes q can be effectively detected by Theorem 2 : Let $n \geq 2$ be integer. For each pair of integers (i,j) with $1 \leq i < j \leq n-1$, let r_{ij} be the remainder of the division of ni by j. Let t be the total number of such pairs (i,j) for which $j \leq 2r_{ij}$. Then: n is prime iff $4t = (n-1)^2$. Also: n is composite iff $4t < (n-1)^2$. In contradistinction to the twin prime conjecture we prove: Theorem 3 : There exist infinitely many primes p for which p+2 is composite. Theorem 4: There exist infinitely many primes q for which q-2 is composite. As an aside we obtain Theorem 5 : The Diophantine inequation $x-y \geq 4$ has infinitely many solutions in (successive) primes x and y. (Co-author: E. Zachariou)

A.B. CREMERS:

On hierarchic protocols

There are many reasons for process communication to be cast in a strict topology of access capabilities. For the special case of tree communication schemes an implementation of so-called "hierarchic protocols" is suggested and explicated in terms of a well known paradigm of concurrent programming control: critical regions. Two special aspects of the "efficiency" of hierarchic protocols are investigated: Regarding the amount of "bias"(inequitable distribution of waiting times among processes) introduced by a communication tree an elementary queueing-theoretic analysis relates this quantity to the overall traffic intensity in the tree. With respect to a machine-independent complexity measure it is shown that hierarchic protocols can win over "flat" ones.

K. MEHLHORN:

Cost Tradeoffs in graph embeddings, with applications

An embedding of the graph G in the graph H can be viewed as a one-to-one association of the vertices of G with the vertices of H. There are two natural measures of the cost of graph embedding : dilation cost (the maximum distance in H between the images of vertices that are adjacent in G) and expansion cost (the ratio of the size of H to the size of G). The main results of this paper illustrate three situations wherin one of the cost measures can be minimized only at the expense of a dramatic increase in the other cost. We also show that any binary tree with n nodes can be embedded into a complete binary tree of depth O(log n) with dilation cost O(1). The later result seems to be of general use in computational situations, where balanced trees are preferable to arbitrary tree structures.

M.S. PATERSON:

Implementations of priority queues using sequential storage

The usual algorithms implementing priority queues (INSERT, DELETE-MIN operations) make essential use of pointers, arrays or random-access storage. There are applications, particularly in large-scale data-processing, where a more restricted form of storage access is desirable. We show some simple priority queue algorithms which use only pushdown stacks or sequential tapes. These algorithms provide an efficient solution to the problems considered in our earlier joint work on the Mail-Carrier Problem.

We give here just our simplest algorithm which nevertheless runs within a constant factor of our proven lower bound of

$$\sum_{\text{insertions}} \log_2(1 + \text{size of queue at insertion time})$$

QUICKPRIOR( )   % S is a sorted list %

S := {S}; repeat S := $Q(\lfloor \log_2 |S| \rfloor, S)$ end-repeat

Q(k,S) % performs 2^k operations on queue S where

$2^k \leq |S| \leq 3.2^k$ %

if k = 0 then (do one operation in naive way since S is small)

else return $(merge(Q(k-1,Q(k-1,S_0)),S_1)$ where $S_0 \cdot S_1 = S$ and $S_0 = 2^k)$

M. KARPINSKI:

On a Tower Problem of Algebraic Integer and the Kannan-Lipton Orbit Result

Input: Given two algebraic integers λ,μ. Problem: Decide on whether there is any number $k \in w$ such that $\lambda^k = \mu$, if yes compute such a k. Following a general method of A. Schinzel we prove a composite upper Log bound for k. This gives an elementary proof of the claim of R. Kannan and R.J. Lipton on the Orbit Sets of rational matrices. Furthermore, using a standard recurring p-adic support theorem of T. Skolem we display an algorithm for checking equivalence of two towers $T(\lambda_0,\mu_0) = T(\lambda_1,\mu_1)$ for $T(\lambda,\mu) = \{k | \lambda^k = \mu\}$.

Berichterstatter: R. Güttler

Liste der Tagungsteilnehmer

Prof. Dr. G. Ausiello
Istituto di Automatica
v. Endossiana 18
I-oo185 Rom / Italien

Prof. J. Bentley
Dept. of Compter Science
Carnegie-Mellon-University
Pittsburgh, PA 15213
USA

Dipl.Inform. N. Blum
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Dr. Beate Commentz-Walter
IBM Böblingen
Bl 14 Abt. 3252 Entwicklung
und Forschung
Schöneicher First

7o3o Böblingen

Prof. Dr. A. Cremers
Universität Dortmund
Abt. Informatik
Postfach 5o o5 oo

46 Dortmund 5o

Prof. Dr. P. v. Emde Boas
Mathematisch Centrum
2e Boerhaavestraat 49
NL-Amsterdam 1oo5

Niederlande

Prof. Dr. M. Fisher
University of Washington
Department of Computer Science
FR 35
Seattle
Washington 98195
USA

Dr. Ph. Flajolet
IRIA-Laboria
B.P. 5
F-7815o Le Chesnay

Dr. J. Francon
Centre de Calcul du CNRS
B.P. 2o CRO

F-67o37 Strasbourg

R. Güttler
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Prof. Dr. G. Hotz
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Dr. Th. Huynh
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Prof. Neil Jones
Computer Science Department
University of Aarhus
Ny Munkegade
DK-8ooo Aarhus C

Dänemark

Ass. Prof. Dr. R. Kemp
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Prof. Dr. J. v. Leeuwen
Department of Computer Science
University of Utrecht
Utrecht 25o6

Holland

E. McCreight
Xerox Parc
Palo Alto
California

USA

Prof. Dr. K. Mehlhorn
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald

66 Saarbrücken

Prof. Dr. B. Monien
Gesamthochschule Paderborn
FB 17
Mathematik-Informatik
Postfach 1621

479o Paderborn

Prof. Dr. I. Munro
Department of Applied Analysis
University of Waterloo
Waterloo, Ontario N2L 3G1

Canada

Prof. Dr. J. Nievergelt
Institute for Informatics
ETH Zürich
Clausiusstr. 55

CH-8oo6 Zürich

Prof. Dr. H. Noltemeier
Lehrstuhl für Informatik III
RW Technische Hochschule
Büchel 29-31

51oo Aachen

Prof. Dr. T. Ottmann
Universität Karlsruhe
Lehrstuhl f. Angewandte Informatik I
Postfach 638o

75oo Karlsruhe 1

Prof. Dr. Papadimitriou
Massachusetts Institute of
Technology
MIT
545 Technology Square
Cambridge, Mass. o2139
USA

Prof. Dr. M. Paterson
Department of Computer Science
University of Warwick
Coventry, Warwickshire
CV4 7AL
Großbritannien

Prof. Dr. W.-J. Paul
Universität Bielefeld
Fakultät für Mathematik
Postfach 864o

48 Bielefeld

Prof. Dr. E. Savage
Division of Engineering
and Center of Computer and
Information Sciences
Brown University
Providence, Rhode Island
U S A

Prof. Dr. E.M. Schmidt
Department of Computer Science
University of Aarhus
DK-8000 Aarhus
Dänemark

A. Tsakalidis
Universität des Saarlandes
FB Angewandte Mathematik
und Informatik
Im Stadtwald
66 Saarbrücken

Prof. Dr. C.P. Schnorr
Universität Frankfurt
Fachbereich Mathematik
Auf der Morgenstelle 10
6000 Frankfurt a.M.

Prof. Dr. L. Valiant
University of Leeds
Department of Computer Sc.
Leeds
Großbritannien

E. Shamir
Dept. of Mathematics
Hebrew University
Jerusalem
ISRAEL

Prof. Dr. H. Waldschmidt
Techn. Hochschule Darmstadt
Fachbereich Informatik
Magdalenenstr. 11
6100 Darmstadt

Prof. Dr. G. Schlageter
Fachbereich Mathematik
Fernuniversität Hagen
Lützowstraße 125
5800 Hagen

Prof. Dr. H. Walter
Techn. Hochschule Darmstadt
Fachbereich Informatik
Magdalenenstr. 11
6100 Darmstadt

A.O. Slisenko
L O M I
Fontanka 27
Leningrad 191011
U D S S R

Prof. Dr. D. Wood
McMaster University
Dept. of Applied Math.
1280 Main Street West
Hamilton, Ontario
Canada, L8S 4K1

Dr. Chr. Strelen
Technische Hochschule Darmstadt, Fachbereich Informatik
Steubenplatz 12
6100 Darmstadt

Dr. I. Wegener
Fakultät Mathematik-Informatik, Universität Bielefeld
Postfach 8640
4800 Bielefeld

Gefördert durch
DFG Deutsche Forschungsgemeinschaft