The Fourth Dagstuhl Seminar on Data Structures

March 2–6, 1998

organized by

Ian Munro, Stefan Näher, and Peter Widmayer

The design and analysis of algorithms is one of the fundamental areas in computer science. This also involves the development of suitable methods for structuring the data to be manipulated by these algorithms. In this way, algorithms and data structures form a unit and the right choice of algorithms and data structures is a crucial step in the solution of many problems. For this reason, the design, analysis and implementation of data structures form a classical field of computer science both in research and teaching.

The development of the research in this area has been influenced by new application fields such as CAD, Geographic Information Systems, Molecular Biology and Genetics. Not only new methods and paradigms, such as randomization or competitive analysis of algorithms, have been developed, but there is also some shift of interest away from theory, e.g., the classical analysis of asymptotic behavior of algorithms, to more practical issues, such as implementation problems and the usefulness of algorithms in practical applications. One can observe that more and more researchers in computer science also want to make their results available in form of programs or software packages. This trend is also reflected in important international conferences.

Results

The Fourth Dagstuhl Seminar on Data Structures was attended by 41 people from 11 countries. The workshop brought together researchers from all over the world working on different areas of the field of efficient data structures and algorithms. Quite a few interest groups used the workshop to discuss

and develop ongoing and new cooperations. Many interesting results and solutions for theoretical and practical problems were presented.

The explicit consideration of external memory is becoming a major concern. It is crucial for the practical value of algorithms and data structures, and it creates interesting theoretical problems. Accordingly, there have been several contributions of external memory algorithms and data structures for problems in Computational Geometry and Geographic Informations Systems.

On the theoretical side, there have been interesting contributions concerning search & graph structures. A particular result on hashing and several results on special-purpose data & search structures and added features (such as distributed, fault-tolerant, concurrent ...) extended recent insights and limits in the area. Another remarkable result showed a (nearly) exact bound for planar point location.

The geometric data structure section saw new algorithms concerning mobile robots and unknown terrains. An algorithm that extends by far the limits of known problem sizes has been presented for rectilinear Steiner trees.

An open problem session reviewed Knuth's 60 famous open problems. It turned out that the majority of the problems is still unsettled today.

Perspectives

The workshop helped to identify several tracks of research in data structures that play and will continue to play an inportant role in the area. The field of algorithms and data structures continues its expansion towards previously undiscovered fields. As computers influence more and more aspects of our lives, data structures are needed for an ever growing domain of applications, with questions of steadily increasing complexity. The general task of developing and analyzing simple and practical algorithms and data structures remains with us.

Thanks

The abstracts of the 35 talks are contained in this report, which was compiled by Nora Sleumer. The organizers would like to thank all participants and especially the team of Schloß Dagstuhl for helping to make the workshop a success. The warm atmosphere of the castle supported discussions and the informal exchange of recent results even beyond the personal perspective. This is exactly what "Schloß Dagstuhl" stands for in the community.

Contents

1 Simpler and Faster Dictionaries on the AC^0 RAM

Torben Hagerup
MPI für Informatik, Saarbrücken, Germany

We consider the static dictionary problem of using $O(n)$ w-bit words to store n w-bit keys for fast retrieval on a w-bit AC^0 RAM, i.e., on a RAM with a word length of w bits whose instruction set is arbitrary, except that each instruction must be realizable through an unbounded-fanin circuit of constant depth and $w^{O(1)}$ size, and that the instruction set must be finite and independent of the keys stored. We improve the best known upper bounds for moderate values of w relative to n. If $w/\log n = (\log\log n)^{O(1)}$, query time $(\log\log\log n)^{O(1)}$ is achieved, and if additionally $w/\log n \geq (\log\log n)^{1+\epsilon}$ for some fixed $\epsilon > 0$, the query time is constant. For both of these special cases, the best previous upper bound was $O(\log\log n)$. Our method is provably optimal for large values of w relative to n.

2 A New Local Coordinate System Based on the Voronoi Diagram and Its Application to Surface Interpolation

Kokichi Sugihara
University of Tokyo, Japan

Sibson found a "local coordinate property" associated with the Voronoi diagram, and applied it to the surface interpolation from given heights at arbitrarily located sites. This paper presents another local coordinate property that is also based on the Voronoi diagram, where the coordinates are proportional to the length of the Voronoi edge shared by the target point and its neighbor divided by the distance between them. Our new property is simpler in that the coordinates can be computed from the Voronoi diagram directly while Sibson's coordinates require the second-order Voronoi diagram. Moreover, our formula is robust in the sense that it is valid even if we roughly guess the structure of the Voronoi diagram. On the basis of this formula, a new interpolation scheme is also constructed.

3 Kinematic Voronoi Diagrams: Motion meets Geometry

Thomas Roos
ETH Zurich, Switzerland

This talk gives a survey of static, dynamic, and kinematic Voronoi diagrams as a basic tool for Geographic Information Systems (GIS). We show how the Voronoi diagram with its dual, the Delaunay multigraph, can be used to maintain the topology of the map objects of a GIS. The presented method allows the insertion, deletion, and translation of points and line segments in a Voronoi diagram of n generators. All elementary operations are available in $O(\log n)$ expected time using expected linear storage. The Voronoi approach also greatly simplifies some of the basic traditional GIS queries and allows even new types of higher level queries. The concept of a persistent, locally-modifiable spatial data structure that is always complete provides an important new approach to spatial data handling that is not available with existing systems.

This is joint work with Christopher Gold and Peter Remmele

4 A Faster Output-Sensitive Algorithm for Enumerating the Cells of an Arrangement

Nora Sleumer
ETH Zurich, Switzerland

We present a practical algorithm for enumerating the cells C of an arrangement of m hyperplanes. For fixed dimension its time complexity is $O(m|C|)$. This is an improvement by a factor of m over the reverse search algorithm by Avis and Fukuda. The algorithm needs little space, is output-sensitive and straightforward to parallelize.

5 On the Use of Nondeterminism in Data Structures for Boolean Functions

Ingo Wegener
Universität Dortmund, Germany

Data structures for Boolean functions are used in CAD-tools, for verification, test pattern generation, timing analysis but also for combinatorial problems. They have to be compact for many (important) functions and have to support a list of operations among them evaluation, satisfiability test and count, equality test, synthesis, and quantification. The most often used representation are OBDDs (ordered binary decision diagrams). Their disadvantage is that they need exponential size even for quite simple functions. Hence, generalizations are investigated. If one uses EXOR-nondeterminism, the operations take polynomial time but are not efficient enough. Using OR-nondeterminism, negation may cause an exponential blow-up. Partitioned OBDDs restrict OR-nondeterminism such that all operations can be performed efficiently. Complexity theoretical results on partitioned OBDDs are proved. As example it is shown how the number of knight's tours on an 8 x 8-chessboard can be determined with partitioned OBDDs.

6 A Succinct Representation of Huffman Trees for Fast Decoding

Andrej Brodnik
Luleå University, Sweden and University of Ljubljana, Slovenia

We present a succinct data structure storing the Huffman encoding that permits sub-linear decoding in the number of transmitted bits. The size of the extra storage except for the storage of the symbols in the alphabet for the new data structure is $O(l \log N)$ bits, where l is the longest Huffman code and N is the number of symbols in the alphabet. We present a solution that typically decodes texts of sizes ranging from a few hundreds up to 68 000 with only one third to one fifth of the number of memory accesses of that of regular Huffman implementations. In our solution the typical texts mentioned

above have the overhead structure, where we do all but one memory access to, that is never larger than 350 bytes. This will with a very high probability reside in cache, which means that the actual decoding time compares even better.

This is joint work with Svante Carlsson.

7 Selection from Read-Only Memory with Limited Storage

Venkatesh Raman
Institute of Mathematical Sciences, Chennai, India

We consider the selection problem of finding an element of a given rank in a totally ordered set, given in a read-only array. We give efficient algorithms for the problem when close to $O(\log n)$ extra cells are available. Previous work on the problem on a similar model either assumed $\Omega(\log^2 n)$ extra cells or $O(\sqrt{(}\log n/\log\log n))$ cells. When $O(\log n)$ space is available, for example, our algorithm takes $O(n\log^2 n)$ time improving upon the earlier algorithm for the problem that takes $\Theta(n^{1+o(1)})$ time (which though used $o(\log n)$ space, it was not clear how to beat the runtime using $O(\log n)$ space).

This is joint work with Sarnath Ramnath.

8 The Bulk Index Join: A Generic Approach to Processing Non-Equijoins

Jochen van den Bercken
Universität Marburg, Germany

Efficient join algorithms have been developed for processing different types of non-equijoins like spatial-join, band-join, temporal-join or similarity join. Each of these previously proposed join algorithms is tailor-cut for a specific

type of join, and a generalization of these algorithms to other join types is not evident. We present an efficient algorithm called bulk index join that can be easily applied to a broad class of non-equijoins. Similar to the well-known hash join algorithms, the bulk index join performs in two phases. In the build-phase an appropriate index structure is created that serves as a partitioning function on the first relation. In the probing-phase, the objects of the second relation are probed against the first relation. In order to support both phases efficiently, we adopt a technique recently proposed for bulk loading index structures. We show that this technique can be exploited for probing the tuples of the second relation in bulk. Similar to the generic bulk loading approach, only a predefined set of routines of the index structure is used for implementing our join algorithm. This set is generally available in tree-based index structures. We discuss in detail how to apply our generic approach to the band join. At first, we analyze the worst-case performance of the band join. The upper bounds derived from our analysis are the lowest ones we are aware of. Experimental results show the I/O performance of our method in practice. In particular, we show that the probing-phase of the bulk index join is substantially more efficient than the traditional method where queries are performed one at a time.

This is joint work with Bernhard Seeger and Peter Widmayer.

9 On Rectilinear Steiner Minimum Trees

Michael Kaufmann
Universität Tubingen, Germany

We report on our experiments on the computation for RSMTs. Starting from a slow preprocessing phase that enables us to solve instances of size 55 optimally, we improve the implementations in several steps such that now we are on the border to use the fast branch & bound approach by David Warme. We are hoping to be able to solve instances of size 1000 much fast than before in the near future.

This is joint work with Uli Fößmeier and Bernd Schatz.

10 Exploring an Unknown Simple Polygon

Rolf Klein
FernUniversität Hagen, Germany

Suppose a mobile robot equipped with a 360° vision system is standing at some point s on the boundary of some simple polygon. The robot does not know the polygon. Its task is to walk around inside the polygon until it has seen each point of the polygon at least once, and to return to s. In order to give a performance guarantee, we want to compare the length of the robot's path against the length of the optimum watchman tour, i.e. the length of the shortest tour through s from which each point is visible. In this talk we present a strategy that can guarantee the former to be at most 27 times longer than the latter. This improves on the best previous bound of 133.
Our analysis depends on a new geometric structure called the angle hull. Let D be a relatively convex subset of some simple polygon, P. Then the angle hull, AH(D), is defined to be the set of all points inside P that can see two points of D at a right angle. We can show that the perimeter of AH(D) is at most twice the perimeter of D, and that this bound is tight (without the surrounding polygon P the tight bound would be $\frac{\pi}{2}$).

This is joint work with F. Hoffman, K. Kriegel and C. Icking.

11 On Search Trees with Relaxed Balance

Kim S. Larsen
Odense University, Denmark

Search trees with relaxed balance are structures where the rebalancing has been uncoupled from the updating such that rebalancing can be performed at any time and in small steps.
Considering the variety of balanced binary search tree schemes, red-black trees are particularly interesting because they have logarithmic worst-case and amortized constant time rebalancing. Additionally, the maximum number of restructuring operations which can be carried out in response to an update is constant. We show how a relaxed search tree scheme can be defined as a generalization of red-black trees in such a way that all the desirable properties of red-black trees are maintained.

12 Conflict-Free Access to Data Structures in Parallel Memory Systems

Sajal K. Das
University of North Texas, Denton, U.S.A.

Mapping the nodes of a data structure (represented as a graph) to as few memory modules as possible in a multiprocessor architecture, such that different subsets of nodes (called templates) can be accessed in parallel and without memory conflicts, is an important problem. This can be formulated as a hypergraph coloring problem and hence is NP-hard. So we will seek load balanced mappings of various templates in special host data structures like k-ary trees, binomial trees and hypercubes. In trees, the templates of interest are subtrees of a given height, paths from the root to the leaves, or nodes at any level; whereas in hypercubes the potential templates are subcubes of any size or the set of nodes at a given distance from a source. As applications, path templates play a crucial role in designing work-optimal algorithms for maintaining a distributed priority queue, subtree templates in image processing and range queries in search trees, while subcube templates in concurrent files access with designated signatures. The intricacy of the mappings with an optimal number of memory modules is due to the overlappings of template instances. We will conclude with a few open problems.

This is joint work with Cristina Pinotti.

13 On-line Searching in Geometric Trees

Sven Schuierer
Universität Freiburg, Germany

In this talk we study the problem of a robot searching for a target in an unknown geometric tree with m leaves. The target can only be detected if the robot reaches the location of the target. The search cost is proportional to the distance traveled by the robot. We are interested in the *competitive ratio*, that is the ratio of the distance traveled by the robot to the length of the shortest path to reach the goal. We provide optimal upper and lower

bounds for the competitive ratio of search strategies in geometric trees.
As an application of our strategy we present an algorithm to search in simple polygons. It a competitive ratio of $1 + 2(2k)^{2k}/(2k-1)^{2k-1}$ if the polygon boundary can be decomposed into k convex and k reflex chains.

14 On the Exact Query Complexity of Planar Point Location

Raimund Seidel
Universität des Saarlandes, Saarbrücken, Germany

Let $\mathcal{P}$ be a polygonal subdivision of the plane with n edges in total. The planar point location problem asks for a data structure that admits fast answers to queries of the form "Which polygon of $\mathcal{P}$ contains query point q?".
Solutions to this problem with $O(\log n)$ worst case query time using $O(n)$ space have been known for a long time. We consider the EXACT worst case complexity $Q(n)$ of this problem (counting sidedness tests of the query point against lines as primitive steps). Contrary to conjectures that $Q(n) \geq 2\log n$ we show

$$Q(n) \leq \log n + 2\sqrt{\log n} + 1/2 \log\log n + O(1).$$

This is achieved via a relatively straightforward method that is similar to Preparata's trapezoidal method and uses weighted search trees. Its space requirement is superlinear. However, by the use of cuttings, this can be reduced to $O(n)$ while increasing the query time only by $O(\log^{1/4} n)$.
For every natural model of computation that encompasses almost all known point location methods (bucketing and also Kirkpatrick's method are notable exceptions) we show an almost matching lower bound of

$$Q(n) \geq \log n + 2\sqrt{\log n} - 1/2 \log\log n - O(1).$$

15 On Parsing LL-Languages

Norbert Blum
Universität Bonn, Germany

Usually, a parser for an $LL(k)$ - grammar G is a deterministic pushdown transducer which produces a leftmost derivation for a given input string $x \in L(G)$. In 1983, Ukkonen has given a family of $LL(2)$-grammars proving that every parser for these grammars has exponential size. If we add to a parser the possibility to manipulate a constant number of pointers which point to positions within the constructed part of the leftmost derivation and to change the output in such positions, we obtain an extended parser for the $LL(k)$-grammar G. Given an arbitrary $LL(k)$-grammar G, we will show how to construct an extended parser of polynomial size manipulating at most k^2 pointers.

16 Hashing with Linear Maps

Martin Dietzfelbinger
TU Ilmenau, Germany

Consider the set $\mathcal{H}$ of all linear (or affine) transformations between two vector spaces over a finite field F. We study how good $\mathcal{H}$ is as a class of hash functions, namely we consider hashing a set S of size n into a range having the same cardinality n by a randomly chosen function from $\mathcal{H}$ and look at the expected size of the largest hash bucket. $\mathcal{H}$ is a universal class of hash functions for any finite field, but with respect to our measure different fields behave differently.
If the finite field F has n elements then there is a bad set $S \subset F^2$ of size n with expected maximal bucket size $\Omega(n^{1/3})$. If n is a perfect square then there is even a bad set with largest bucket size *always* at least $\sqrt{n}$. (This is worst possible, since with respect to a universal class of hash functions every set of size n has expected largest bucket size below $\sqrt{n} + 1/2$.)
If, however, we consider the field of two elements then we get much better bounds. The best previously known upper bound on the expected size of the largest bucket for this class was $O(2^{\sqrt{\log n}})$. We reduce this upper bound to

$O(\log n \log\log n)$. Note that this is not far from the guarantee for a random function. There, the average largest bucket would be $\Theta(\log n/\log\log n)$.
In the course of our proof we develop a tool which may be of independent interest. Suppose we have a subset S of a vector space D over $\mathbf{Z}_2$, and consider a random linear mapping of D to a smaller vector space R. If the cardinality of S is larger than $c_\varepsilon |R| \log |R|$ then with probability $1-\varepsilon$, the image of S will cover all elements in the range.

This is joint work with Noga Alon, Peter Bro Miltersen, Erez Petrank and Gabor Tardos.

17 k ⌜ k Sorting on the Multi-Mesh

Manfred Kunde
TU Ilmenau, Germany

We present sorting algorithms on the recently introduced multi-mesh, a network consisting of n^2 meshes of size $n \times n$ which are connected by the free marginal links of the meshes. Our algorithm takes $41n + o(n)$ steps which is a significant improvement to previously known algorithms. The sorting algorithm is based on a technique using interchange of data between the $n \times n$ submeshes to distribute information uniformly, an approach which is similar to an all-to-all mapping. Furthermore, with this approach we can also handle $k-k$ problems on the multi-mesh, where each processor contains k elements initially and finally. We show that the $k-k$ sorting problem can be solved in about $9.5kn$ steps, provided $k \geq 12$.

This is joint work with Alfons Avermiddig and Andre Osterloh.

18 I/O-Efficient Algorithms for Contour-Line Extraction and Planar Graph Blocking

Lars Arge
Duke University, Durham, U.S.A.

For a polyhedral terrain Σ, the contour at z-coordinate h, denoted C_h, is defined to be the intersection of the plane $z = h$ with Σ. In this talk, we study the contour-line extraction problem, where we want to preprocess Σ into a data structure so that given a query z-coordinate h, we can report C_h quickly. This is a central problem that arises in geographic information systems (GIS), where terrains are often stored as Triangular Irregular Networks (TINs). We present an I/O-optimal algorithm for this problem which stores a terrain Σ with N vertices using $O(N/B)$ blocks, where B is the size of a disk block, so that for any query h, the contour C_h can be computed using $O(\log_B N + |C_h|/B)$ I/O operations, where $|C_h|$ denotes the size of C_h.
We also present an improved algorithm for a more general problem of blocking bounded-degree planar graphs such as TINs (i.e., storing them on disk so that any graph traversal algorithm can traverse the graph in an I/O-efficient manner), and apply it to two problems that arise in GIS.

This is joint work with Pakaj Agarwal, T.M. Murali, Kasturi Varadarijan and Jeff Vitter.

19 Branch-and-Cut Algorithms for the Maximum Weight Trace Problem

Hans-Peter Lenhof
MPI für Informatik, Saarbrücken, Germany

Multiple sequence alignment is an important problem in computational molecular biology. We study the Maximum Trace formulation introduced by Kececioglu. We first phrase the problem in terms of forbidden subgraphs, which enables us to express Maximum Trace as an integer linear-programming problem, and then solve the integer linear program using methods from polyhedral

combinatorics. The trace *polytope* is the convex hull of all feasible solutions to the Maximum Trace problem; for the case of two sequences, we give a complete characterization of this polytope. This yields a polynomial-time algorithm for a general version of pairwise sequence alignment that, perhaps surprisingly, does not use dynamic programming; this yields, for instance, a non-dynamic-programming algorithm for sequence comparison under the 0-1 metric, which gives another answer to a long-open question in the area of string algorithms. For the multiple-sequence case, we derive several classes of facet-defining inequalities and show that for all but one class, the corresponding separation problem can be solved in polynomial time. This leads to a branch-and-cut algorithm for multiple sequence alignment, and we report on our first computational experience. It appears that a polyhedral approach to multiple sequence alignment can solve instances that are beyond present dynamic-programming approaches.

This is joint work with John Kececioglu, Petra Mutzel, Kurt Mehlhorn, Knut Reinert and Martin Vingron.

20 Optimal External Memory Algorithms for Some Geometric Problems

Kurt Mehlhorn
MPI für Informatik, Saarbrücken, Germany

We present optimal randomized external memory algorithms for problems like 2d and 3d convex hull, trapezoidal decomposition, ... All algorithms follow the same paradigm: graded randomized incremental construction.

This is joint work with Andreas Cramer, Paolo Ferragina, Uli Neyer and Edgar Ramos.

21 Data Structures and Algorithms for Detecting Unusual Words

Alberto Apostolico
University of Padova, Italy

The problem of characterizing and detecting over- or under-represented words in sequences arises ubiquitously in diverse applications and has been studied rather extensively in Computational Molecular Biology. In most approaches to the detection of unusual frequencies of words in sequences, the words (up to a certain length) are enumerated more or less exhaustively and individually checked in terms of observed and expected frequencies, variances, and scores of discrepancy and significance thereof. We take instead the global approach of annotating a suffix trie or automaton of a sequence with some such values and scores, with the objective of using it as a collective detector of all unexpected behaviors, or perhaps just as a preliminary filter for words suspicious enough to warrant further and more accurate scrutiny. We show that such annotations can be carried out in a time- and space efficient fashion for the mean, variance and some of the adopted measures of significance, even without a-priori limits on the length of the words considered. Specifically, we concentrate on the simple probabilistic model in which sequences are produced by a random source emitting symbols from a known alphabet independently and according to a given distribution. We discuss data structures and tools for computing and storing the expected value and variance of all substrings of a given sequence of n symbols in (optimal) $O(n^2)$ overall worst-case, $O(n \log n)$ expected time and space. The $O(n^2)$ time bound constitutes an improvement by a linear factor over the direct method. We show that under several accepted measures of deviation from expected frequency, the candidate over- and under-represented words are $O(n)$ only. This surprising fact is a consequence of combinatorial properties that constrain the distributions of word occurrences in a string. Based on this, we design global detectors of favored and unfavored words for our probabilistic framework, and display the results of some preliminary experiments.

This is joint work with M.E. Bock, S. Lonardi and X. Xu.

22 Efficient Algorithms for Petersen's Matching Theorem

Erik Demaine
University of Waterloo, Canada

Petersen's theorem is a classic result in matching theory from 1891. It states that every 3-regular bridgeless graph has a perfect matching. Our work explores algorithms for finding perfect matchings in such graphs. This has several applications, including dynamic mesh refinement and terrain guarding. Previously, the only relevant matching algorithms were for general graphs, so the best algorithm for these graphs ran in $O(n^{3/2})$ time. We have developed an $O(n \log^4 n)$-time algorithm for matching in a 3-regular bridgeless graph. For when the graph is also planar, which is the case for the applications mentioned above, we have an optimal $O(n)$-time algorithm.

This is joint work with Therese Biedl, Prosenjit Bose and Anna Lubiw.

23 Efficient Algorithms for Constructing Fault-Tolerant Geometric Spanners

Michiel Smid
University of Magdeburg, Germany

Let S be a set of n points in a metric space, and k a positive integer. Algorithms are given that construct k-fault-tolerant spanners for S. If in such a spanner at most k vertices and/or edges are removed, then each pair of points in the remaining graph is still connected by a "short" path. First, an algorithm is given that transforms an arbitrary spanner into a k-fault-tolerant spanner. For the Euclidean metric in d-dimensional space, this leads to an $O(n \log n + c^k n)$–time algorithm that constructs a k-fault-tolerant spanner of degree $O(c^k)$, whose total edge length is bounded by $O(c^k)$ times the weight of a minimum spanning tree of S, for some constant c. For constant values of k, this result is optimal. In the second part of the talk, an algorithm is presented for the Euclidean metric in $d - space$. This algorithm constructs in $O(n \log n + k^2 n)$ time a k-fault-tolerant spanner with $O(k^2 n)$ edges.

This is joint work with Christos Levcopoulos and Giri Narasimhan.

24 Fast and Simple External-Memory Planar Point-Location

Jan Vahrenhold
University of Münster, Germany

We present an algorithm for external-memory planar point-location that is both effective and easy to implement. The base algorithm is an external-memory variant of the bucket method by Edahiro, Kokubo and Asano that is combined with a batched variant of Dobkin and Lipton's slab method for the internal-memory part of the point-location process. We analyze this combined algorithm in terms of its I/O- and internal-memory efficiency and show by empirical evaluation that - despite of a non-optimal worst-case I/O complexity - this method performs very fast for real-world data. The techniques used for preprocessing and locating can take advantage of eventually existing buffering strategies of the underlying operating system since there are mainly sequential I/Os to be performed. Furthermore, the algorithm can be extended in a straightforward way to use multiple disks and/or CPUs.

This is joint work with Klaus Hinrichs.

25 Dynamic Trees

Stephen Alstrup
University of Copenhagen, Denmark

Sleator and Tarjan's dynamic trees, ST-trees, is one of the most used data structures. ST-trees can maintain a dynamic forest of trees and support different operations: e.g. find a edge with minimum cost on a path. However ST-trees are difficult to implement and can only be used for a certain class of problems.
This is the reason why we introduce TOP-trees. TOP-trees give the user the maximum possibility for different kind of operations and are much simpler to implement. Furthermore we achieve quadratic improvement on the complexity for certain kind of problems.

This is joint work with Jacob Holm, Krisitan de Litchenberg and Mikkel Thorup.

26 Bulk Loading External Index Structures

Bernhard Seeger
Universität Marburg, Germany

We consider the problem of building an index from scratch for a given set of data items. We assume that main memory is too small to keep the data set (and the index) resident in memory. We present a framework based on buffer-trees. The framework allows us to do bulk-loading for a broad class of index structures. For index structures with $O(\log_B N)$ insertion time, bulk loading can be done in $O(N/B \log_{M/B} N/B)$, where N is the number of items, M is the size of available main memory and B is the page capacity.

This is joint work with Jochen van den Bercken and Peter Widmayer.

27 Slow Preprocessing of Large Graphs for Very Fast Shortest Path Calculations

Ulrich Lauther
Siemens AG, München, Germany

We present a new preprocessing method that needs space linear in the number of edges and allows for extremly fast shortest path calculations on road maps. On a road map of Germany containing more than 1 Million edges, a shortest path across the whole network is calculated within 30 milliseconds. Methods for speeding up the preprocessing phase are discussed.

This is joint work with Reinhard Enders.

28 The VASCO Spatial Data Structure Applet

Hanan Samet
University of Maryland, U.S.A.

Features of the VASCO spatial data structure applet are described to support its demonstration. This includes an examination of the different R-tree node splitting methods implemented in VASCO. Different quadtree and k-d tree representations of point, line, and rectangle data are described. The applet enables users to visualize different spatial data structures and as well as observe their behavior, in an animated manner, for finding the nearest neighbors to a query point and the objects within a given query window in an incremental manner. The applet can be found at

`http://www.cs.umd.edu/~hjs/quadtree/index.html`

This is joint work with Frantisek Brabec.

29 Randomized Splay Trees

Martin Fürer
Pennsylvania State University, USA

In order to compare the performance of different methods to handle binary search trees, we need to investigate the constant factors in the running times. We use the following simple measure. Traversing an edge to access an item costs one time unit, while a single rotation costs R time units. Sleator and Tarjan's semisplaying runs with an amortized time bound of $2(1+R)\log n + O(1)$ per access (except for insertions).
We study several randomized versions of splaying algorithms using a moderate amount of randomness ($O(1)$ coin tosses per access). The factor of $2(1+R)$ of the expected running time can be decreased to a value close to 2. If we insist on the previous (worst case) amortized time, then the factor in the expected time can still be decreased to $1.5(1+R)$. Furthermore, we observe a trade-off between expected and amortized time.

Finally, a structural global splaying method allows more significant savings in terms of the number of pointer modifications rather than rotations. Interesting open problems concern the dependencies between the Dynamic Optimality Conjectures for the various splaying methods.

30 Worst-Case Efficient External-Memory Priority Queues

Gerth Stølting Brodal
MPI für Informatik, Saarbrücken, Germany

A priority queue Q is a data structure that maintains a collection of elements, each element having an associated priority drawn from a totally ordered universe, under the operations INSERT, which inserts an element into Q, and DELETEMIN, which deletes an element with the minimum priority from Q. In this paper a priority-queue implementation is given which is efficient with respect to the number of block transfers or I/Os performed between the internal and external memories of a computer. Let B and M denote the respective capacity of a block and the internal memory measured in elements. The developed data structure handles any intermixed sequence of INSERT and DELETEMIN operations such that in every disjoint interval of B consecutive priority-queue operations at most $c \log_{M/B} \frac{N}{M}$ I/Os are performed, for some positive constant c. These I/Os are divided evenly among the operations: if $B \geq c \log_{M/B} \frac{N}{M}$, one I/O is necessary for every $B/(c \log_{M/B} \frac{N}{M})$-th operation and if $B < c \log_{M/B} \frac{N}{M}$, $\frac{c}{B} \log_{M/B} \frac{N}{M}$ I/Os are performed per every operation. Moreover, every operation requires $O(\log_2 N)$ comparisons in the worst case. The best earlier solutions can only handle a sequence of S operations with $O(\sum_{i=1}^{S} \frac{1}{B} \log_{M/B} \frac{N_i}{M})$ I/Os, where N_i denotes the number of elements stored in the data structure prior to the i-th operation, without giving any guarantee for the performance of the individual operations.

This is joint work with Jyrki Katajainen, University of Copenhagen

31 $\mathbf{O}(\log\log\mathbf{N})$ Time Algorithms for Hamiltonian-Suffix and Min-Max-Pair Heap Operations on the Hypercube

M. Cristina Pinotti
Istituto di Elaborazione dell'Informazione, CNR, Pisa, Italy

This paper deals with a fast implementation of a heap data structure on a synchronous, distributed multicomputer system based on the hypercube topology. In particular, we present an efficient mapping of a min-max-pair heap on the hypercube architecture in which the load on each processor's local memory is balanced and no additional communication overhead is incurred. Optimal parallel algorithms are proposed which require $O(\frac{\log N}{p} + \log p)$ time to perform single insertion, deletemin and deletemax operations on a min-max-pair heap of size N, where p is the number of processors. Our novel approach is based on an optimal mapping of the paths of a binary heap into a hypercube such that in $O(\frac{\log N}{p} + \log p)$ time we can compute the Hamiltonian-Suffix, which is defined as a pipelined suffix-minima computation on any path of the heap embedded into the Hamiltonian path of the hypercube according to the binary reflected Gray codes. Note however that the binary tree underlying the heap data structure is not altered by the mapping process.

32 Efficient External Memory Algorithms by Simulating Coarse-Grained Parallel Algorithms

Frank Dehne
Carleton University, Ottawa, Canada

We provide simulation techniques that will produce efficient EM algorithms from efficient parallel algorithms for the coarse-grained multicomputer model. Our techniques can accommodate one or multiple processors on a target machine with one or multiple disks per processor.

33 Concurrency and Recovery for Search Structures

Eljas Soisalon-Soininen
Helsinki University of Technology, Finland

Efficient management of large databases requires maintaining index structures for accessing data. Concurrent transactions require that index structures can be accessed concurrently. Moreover, in case of failures it is often important that efficient recovery methods exist for indices, not only for transactions. In this talk we discuss methods for combining concurrency with efficient recoverability in the case of search tree indices. As an example of an application in which fast recovery is extremely important we consider switching systems for mobile telephones.

34 Open Problems in the Analysis of Sorting and Searching Algorithms

Robert Sedgewick
Princeton University, U.S.A.

We survey the $\approx$ 60 open problems described in Knuth's volume 3 (those rated 46–50) and discuss the $\approx$ 60 that have been solved in the 25 years since the publication of the book.
Three open problems are discussed in detail:

1. Average-case analysis of shellsort.
2. Are balanced trees asymptotically optimal?
3. Practical sorting networks.

These problems have been open for 30 years and more, despite the many attempts to solve them.

35 Fully Dynamic Distributed Search Trees

Enrico Nardelli
University of L'Aquila, Italy

In this talk we consider the dictionary problem in a message-passing distributed environment. We introduce a new version of distributed search trees, the first to be fully scalable, that is capable to both grow and shrink as long as keys are inserted and deleted. We prove that in the worst case a key can be inserted, searched or deleted with $O(\log^2 N)$ messages. We show that this bound is tight.

This is joint work with Fabio Barillari and Massimo Pepe.