08081 Abstracts Collection
Data Structures
— Dagstuhl Seminar —

Lars Arge[1], Robert Sedgewick[2], and Raimund Seidel[3]

[1] BRICS - Aarhus, DK
large@madalgo.au.dk
[2] Princeton University, USA
rs@cs.princeton.edu
[3] Universität des Saarlandes, DE
rseidel@cs.uni-sb.de

Abstract. From February 17th to 22nd 2008, the Dagstuhl Seminar 08081 "Data Structures" was held in the International Conference and Research Center (IBFI), Schloss Dagstuhl. It brought together 49 researchers from four continents to discuss recent developments concerning data structures in terms of research but also in terms of new technologies that impact how data can be stored, updated, and retrieved. During the seminar a fair number of participants presented their current research. There was discussion of ongoing work, and in addition an open problem session was held. This paper first describes the seminar topics and goals in general, then gives the minutes of the open problem session, and concludes with abstracts of the presentations given during the seminar. Where appropriate and available, links to extended abstracts or full papers are provided.

Keywords. Data structures, information retrieval, complexity, algorithms, flash memory

08081 Summary – Data Structures

The purpose of this workshop was to discuss recent developments in various aspects of data structure research and also to familiarize the community with some of problems that arise in the context of using so-called flash memory as a storage medium. 49 researchers attended the workshop producing a congenial and productive atmosphere.

A number of contributions reported on new advances on old problems in data structure research: E.g. John Iacono summarized recent advances regarding the dynamic optimality conjecture of splay trees, Bob Tarjan showed new results on dynamically maintaining an acyclic graph, Bob Sedgewick presented a particularly simple version of red-black trees, and Mihai Patrascu offered a new, plausible approach for proving super-polylogarithmic lower bounds to dynamic data structure problems.

Dagstuhl Seminar Proceedings 08081
Data Structures
http://drops.dagstuhl.de/opus/volltexte/2008/1532

The design of data structures and algorithms that can deal well with the memory hierarchy used in today's computers was another core topic of the workshop. E.g. Nobert Zeh discussed this problem in the context of red-blue line segment intersection, Rico Jacob considered the multiplication of dense vectors by a sparse matrix, and Michael Bender presented so-called streaming B-trees that can implement dictionaries well in this memory hierarchy context.

Reducing the actual space requirement of data structures to a minimum was the topic of several contributions. For instance Martin Dietzfelbinger and also Arash Farzan considered this problem from the theoretical point of view, and Peter Sanders and also Holger Bast from a very practical point of view.

A highlight of the workshop were the presentations by San Lyul Min and by Ethan Miller on so-called flash memory, which is physical memory that has found its way into many storage devices but has technological properties that lead to the following peculiarities: (i) Before a bit is written it has to be cleared, however this can only be achieved by clearing an entire block of bits. (ii) Every block of bits can only be cleared a (large) constant number of times before the storage capabilities of this block become too unreliable.

There were many more interesting contributions than listed here and, of course, countless discussions and collaborations. The Dagstuhl atmosphere provided just the right environment for all this.

Open Problem Session

There was a very lively open problem session on the second evening of the workshop. Here are some of the problems presented during that session:

0.1 Holger Bast

Let $I = I_1, .., I_e$, I_i random subsets of $[n]$, $|I_i| = M = \delta \cdot n$ and let B be some constant.

Consider the following algorithm:

Pick $i, j \in [e]$ at random.

If $|I_i \cap I_j| > B$ then remove any B elements in $I_i \cap I_j$ from both I_i and I_j.

Repeat.

Let $||I|| := \sum_i |I_i|$. Eventually the process will end (i.e. no more elements are removed) and then $||I|| = \epsilon \cdot e \cdot M$.

Assuming all the I_i's stay random and decrease the same way, the process will go on as long as $(\epsilon \cdot \delta)^2 \cdot n > B$, so the expected value of ϵ is $1/\delta \cdot \sqrt{B/n}$. Can one prove this bound (since those I_i's are not random any more after one step)?

0.2 Riko Jacob

Consider permuting in the IO-model: It is known that for $\log_{M/B}(N/M) < B$ sorting is the right thing to do, while otherwise we need one IO for each element. This is asymptotically optimal and almost all permutations will need this time.

Now, what about a concrete permutation? Can one decide, whether a specific permutation is "hard"?

0.3 Valerie King

Consider a sensor network of a long line of sensors, where every sensor sends its message to the next r sensors. The sensors send out their message from left to right, one after the other. Some of those sensors can have failures, so that they do not work at all, but we know, that at least half of every r sensors in a row are working. Now, we want to broadcast a message from the first to the last sensor and want it definitely to reach its destination (assuming first and last sensor work). How many sensors does every sensor have to listen to?

An easier case is the following: there are just $r+1$ sensors, the first r already have the message, but are possibly not working (at least $r/2$ work!) and we want the last sensor to definitely get the message. Then the last sensor has to listen to an expected number of $\mathcal{O}(\sqrt{r})$ sensors: He samples randomly $\sqrt{r}$ of the first half of the r sensors and listens to them. If he does not have the message, yet, he listens to all of the rest of the sensors, listening to at most $\sqrt{r} + r/2$, but expected $\mathcal{O}(\sqrt{r})$ sensors.

0.4 Rajeev Raman

We are generalizing the problem of computing a minimum spanning tree: We have a graph and no known edge lengths, but only intervals in which the edge lengths are known to lie. We are allowed to query an edge to get the actual length. Now we want to compute the edge set of the MST getting a low competitive ratio. An optimal ratio has been obtained for the deterministic case, but the randomized competitive ratio is unknown.

0.5 Bob Tarjan

We are given subsets $A_1, .., A_n$ of a universe U. Now we want to find all the pairs (i, j) such that $A_i \subset A_j$. For $|U| \approx n$ the trivial solution needs cubic time. Is there a faster solution? (Rasmus Pagh pointed out we can do this in the same time bound as matrix multiplication. But of interest here would be a combinatorial algorithm.)

0.6 Moshe Lewenstein

Consider a graph G and a memory with k cells for nodes of the graph. The memory "eats" an edge (u, v) if u and v are in the memory at the same time. Now the problem is to preprocess the graph to find an order of insertions of nodes into memory cells such that all edges are eaten with as few insertions as possible.

For $k = 2$ the cost is obviously $\leq 2m$. With m insertions we can probably only eat up caterpillar graphs: A path were we connect single vertices to arbitrary nodes on the path.
Minutes by Karl Bringmann

Kinetic kd-Trees and Longest-Side kd-Trees

Mohammad Abam (Univ. of Aarhus, DK)

We propose a simple variant of kd-trees, called rank-based kd-trees, for sets of points in $\mathcal{R}^d$.

We show that a rank-based kd-tree, like an ordinary kd-tree, supports range search queries in $O(n^{1-1/d} + k)$ time, where k is the output size. The main advantage of rank-based kd-trees is that they can be efficiently kinetized: the KDS processes $O(n^2)$ events in the worst case, assuming that the points follow constant-degree algebraic trajectories, each event can be handled in $O(\log n)$ time, and each point is involved in $O(1)$ certificates.

We also propose a variant of longest-side kd-trees, called rank-based longest-side kd-trees (RBLS kd-trees, for short), for sets of points in $\mathcal{R}^2$. RBLS kd-trees can be kinetized efficiently as well and like longest-side kd-trees, RBLS kd-trees support nearest-neighbor, farthest-neighbor, and approximate range search queries in $O((1/\epsilon)\log^2 n)$ time.

The KDS processes $O(n^3 \log n)$ events in the worst case, assuming that the points follow constant-degree algebraic trajectories; each event can be handled in $O(\log^2 n)$ time, and each point is involved in $O(\log n)$ certificates.

Keywords: Kinetic data structures, kd-tree, longest-side kd-tree

Joint work of: Abam, Mohammad; de Berg, Mark; Speckmann, Bettina

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2008/1530

Full Paper:
http://portal.acm.org/citation.cfm?doid=1247069.1247133

Computing contour maps and answering contour queries

Pankaj Kumar Agarwal (Duke University, US)

A terrain M is the graph of a bivariate function. We assume that M is represented as a triangulated surface with N vertices. A contour (or isoline) of M is a connected component of a level set of M. Generically, each contour is a closed polygonal curve; at "critical" levels these curves may touch each other or collapse to a point. We present I/O-efficient algorithms for the following two problems related to computing contours of M:

(i) Given a sequence $l_1, \ldots, l_s$ of real numbers, we present an I/O-optimal algorithm that reports all contours of M at heights $l_1, \ldots, ls$ using $O(\text{sort}(N) + T/B)$ I/Os, where T is the total number edges in the output contours, B is the "block size," and $\text{sort}(N)$ is the number of I/Os needed to sort N elements. The algorithm uses $O(N/B)$ disk blocks. Each contour is generated individually with its composing segments sorted in clockwise or counterclockwise order. Moreover, our algorithm generates information on how the contours are nested.

(ii) We can preprocess M, using $O(\text{sort}(N))$ I/Os, into a linear-size data structure so that all contours at a given height can be reported using $O(\log_B N + T/B)$ I/Os, where T is the output size. Each contour is generated individually with its composing segments sorted in clockwise or counterclockwise order.

Keywords: Terrain, contours, GIS

Joint work of: Agarwal, Pankaj Kumar; Arge, L; Molhave, T.; Sadri B.

IO-Efficient Faceted Search

Holger Bast (MPI für Informatik - Saarbrücken, DE)

This is work in progress on a particular indexability problem related to faceted search. The problem is the following. We are given an array $A[1..n]$. Queries are of the form, given a subset I of $1..n$, compute the sequence a_i for i in I. The problem is trivially solved with $|I|$ operations. When counting number of IO operations the problem becomes more interesting, however. The ultimate goal there would be $|I|/B$ IO-operations when a single IO-operation can read a block of B values. As a preprocessing, we are allowed to create an array of size N with copies of the elements from A. The question is how large N must be to achieve or come close to the optimal number of IOs for arbitrary queries, or for an interesting and relevant subset of all queries.

Keywords: Faceted Search, IO-efficiency, Indexability

Joint work of: Bast, Holger; Amini, Omid; Chan, Hubert; Karrenbauer, Andreas

Cache-Oblivious Streaming B-trees

Michael A. Bender (SUNY at Stony Brook, US)

A streaming B-tree is a dictionary that efficiently implements insertions and range queries. We present two cache-oblivious streaming B-trees, the shuttle tree, and the cache-oblivious lookahead array (COLA).

For block-transfer size B and on N elements, the shuttle tree implements searches in optimal $O(\log_{B+1} N)$ transfers, range queries of L successive elements in optimal $O(\log_{B+1} N + L/B)$ transfers, and insertions in

$O((\log_{B+1} N)/B^{\Theta(1/(\log\log B)^2)} + (\log^2 N)/B)$ transfers, which is an asymptotic speedup over traditional B-trees if $B \geq (\log N)^{1+c/\log\log\log^2 N}$ for any constant $c > 1$.

A COLA implements searches in $O\log N$ transfers, range queries in $O(\log N + L/B)$ transfers, and insertions in amortized $O((\log N)/B)$ transfers, matching the bounds for a (cache-aware) buffered repository tree. A partially deamortized COLA matches these bounds but reduces the worst-case insertion cost to $O(\log N)$ if memory size $M = \Omega(\log N)$. We also present a cache-aware version of the COLA, the Lookahead Array, which achieves the same bounds as Brodal and Fagerberg's (cache-aware) B$^\epsilon$-tree.

We compare our COLA implementation to a traditional B-tree. Our COLA implementation runs 790 times faster for random insertions, 3.1 times slower for insertions of sorted data, and 3.5 times slower for searches.

Keywords: Cache-oblivious, COLA, External memory, Lookahead array, Streaming B-tree

Joint work of: Bender, Michael A.; Farach-Colton, Martin; Fineman, Jeremy T.; Fogel, Yonatan R.; Kuszmaul, Bradley C.; Nelson, Jelani

On Geometric Spanners

Prosenjit Bose (Carleton University, CA)

Given a triangulation G, whose vertex set V is a set of n points in the plane, and given a real number γ with $0 < \gamma < \pi$, we design an $O(n)$-time algorithm that constructs a connected spanning subgraph G' of G whose maximum degree is at most $14 + \lceil 2\pi/\gamma \rceil$. If G is the Delaunay triangulation of V, and $\gamma = 2\pi/3$, we show that G' is a t-spanner of V (for some constant t) with maximum degree at most 17, thereby improving the previously best known degree bound of 23. If G is a triangulation satisfying the diamond property, then for a specific range of values of γ dependent on the angle of the diamonds, we show that G' is a t-spanner of V (for some constant t) whose maximum degree is bounded by a constant dependent on γ. If G is the graph consisting of all Delaunay edges of length at most 1, and $\gamma = \pi/3$, we show that a modified version of the algorithm produces a plane subgraph G' of the unit-disk graph which is a t-spanner (for some constant t) of the unit-disk graph of V, whose maximum degree is at most 20, thereby improving the previously best known degree bound of 25.

Keywords: Plane Spanners, Geometric Graphs, Bounded Degree

Joint work of: Bose, Prosenjit; Smid, Michiel; Xu, Daming

Full Paper:
http://cg.scs.carleton.ca/~jit/publications/publications.html

Is 2 x 1D better than 2D?

Andrej Brodnik (University of Primorska, SI)

In the talk we present two classes of solutions to the closest neighbor problem in two dimensions. The first class of solutions consists of a 2D data structure k-d tree. The second class of solutions uses hierarchically two single dimension data structures: first split the universe into stripes (hashing) and second one of simple 1D data structures RB trees, (a,b)-deterministic skip lists or (a,b)-trees.

We tested the mentioned data structures on a problem of iterative insertion and search for the closest neighbor: i.e., for every point p, insert the point and find the closest point among already inserted ones. Although the worst case of 1D structures is $O(n)$ per iteration and consequently $O(n^2)$ for all operations, the empirical tests on synthetic (Gaussian, uniform, clustered etc. distribution) and real data from a GIS database show that 1D structures outperform 2D structure.

The presented results show that the worst case analysis is not appropriate and hence an average case or amortized analysis is necessary to explain the results and behavior of our solution. The 1D solution is actually used in a practical production line.

Keywords: Closest neighbour, emprical results, 2D, 1D

Joint work of: Brodnik, Andrej; Zadravec, Mirko; Žalik, Borut

Succinct data structures for retrieval and approximate membership

Martin Dietzfelbinger (TU Ilmenau, DE)

The *retrieval problem* is the problem of associating data with keys in a set. Formally, the data structure must store a function $f\colon U \to \{0,1\}^r$ that has specified values on the elements of a given set $S \subseteq U$, $|S| = n$, but may have any value on elements outside S.

Minimal perfect hashing makes it possible to avoid storing the set S, but this induces a space overhead of $\Theta(n)$ bits in addition to the nr bits needed for function values.

In the talk we show how an approach of (*Chazelle et al., SODA 2004*) ("hash-read-add") can be extended to eliminate this overhead. If we allow logarithmic evaluation time, the additive overhead can be reduced to $O(\log\log n)$ bits whp. Moreover, we show that for any k query time $O(k)$ can be achieved using space that is within a factor $1 + e^{-k}$ of optimal, asymptotically for large n. The time to construct the data structure is $O(n)$, expected.

A main technical ingredient is to utilize existing tight bounds on the probability of almost square random matrices with rows of low weight to have full row rank.

Further, we propose a general reduction that transfers the results on retrieval into analogous results on *approximate membership*, a problem traditionally addressed using Bloom filters. Again, we show how to eliminate the space overhead present in previously known methods, and get arbitrarily close to the lower bound.

The evaluation procedures of our data structures are extremely simple (as in the work of Bloom, and Chazelle *et al.*) For the results stated above we assume free access to fully random hash functions. However, we show how to justify this assumption using extra space $o(n)$ to simulate full randomness on a RAM.

Finally, we point out a tight relationship between retrieval structures of the hash-read-add type and dictionary structures that use the cuckoo hashing (or balanced allocation) approach. This gives tighter space bounds for d-ary cuckoo hashing and cache-friendly retrieval structures.

Joint work of: Dietzfelbinger, Martin; Pagh, Rasmus

A Uniform Approach Towards Succinct Representation of Trees

Arash Farzan (University of Waterloo, CA)

We propose a uniform approach for succinct representation of various families of trees. The method is based on two recursive decomposition of trees into subtrees. Recursive decomposition of a structure into substructures is a common technique in succinct data structures and has already been shown fruitful in succinct representation of ordinal trees and dynamic binary trees. We take an approach that simplifies the existing representation of ordinal trees allowing the same full set of navigational operations.

The approach applied to cardinal (i.e. k-ary) trees yields a space-optimal succinct representation which allows full set of cardinal-type operations (e.g. determining child labeled i) as well as the full set of ordinal-type operations (e.g. reporting subtree size).

Existing space-optimal succinct representations had not been able to support both types of operations. We demonstrate how the approach can be applied to obtain a space-optimal succinct representation for the family of free trees where the order of children is insignificant.

Furthermore, we show that our approach can be used to obtain entropy-based succinct representations. We show that our approach matches the degree-distribution entropy suggested by Jansson et al. (2007). We discuss that our approach can be made adaptive to various other entropy measures.

Keywords: Succinct data structures, trees

Trimming of Graphs, with Application to Point Labeling

Torben Hagerup (Universität Augsburg, DE)

For $t, g > 0$, a vertex-weighted graph of total weight W is (t, g)-*trimmable* if it contains a vertex-induced subgraph of total weight at least $(1 - 1/t)W$ and with no simple path of more than g edges.

A family of graphs is *trimmable* if for each constant $t > 0$, there is a constant $g = g(t)$ such that every vertex-weighted graph in the family is (t, g)-trimmable.

We show that every family of graphs of bounded domino treewidth is trimmable. This implies that every family of graphs of bounded degree is trimmable if the graphs in the family have bounded treewidth or are planar. Based on this result, we derive a polynomial-time approximation scheme for the problem of labeling weighted points with nonoverlapping sliding labels of unit height and given lengths so as to maximize the total weight of the labeled points.

This settles one of the last major open questions in the theory of map labeling.

Keywords: Graph trimming, domino treewidth, planar graphs, label placement, map labeling, polynomial-time approximation schemes (PTAS)

Joint work of: Erlebach, Thomas; Hagerup, Torben; Jansen, Klaus; Minzlaff, Moritz; Wolff, Alexander;

See also: Proc., 25th Annual Symposium on Theoretical Aspects of Computer Science (STACS 2008), pp. 265-276.

Space-filling curves for R-trees

Herman J. Haverkort (TU Eindhoven, NL)

We consider the use of space-filling curves to divide a set of points or rectangles in the plane among blocks of external memory, such that each block has a small bounding box; this facilitates efficient queries. We show that traditional measures of the locality-preserving properties of space-filling curves do not predict well if we will indeed get small bounding boxes. Therefore we propose a new measure of the "bounding-box quality" of space-filling curves, and discuss the value of this measure for several well-known and new curves, lower bounds, and experimental results.

Rectangles in the plane are specified by four coordinates, and can therefore be grouped into blocks using a four-dimensional space-filling curve. We discuss a property that a four-dimensional space-filling curve should have to distribute both small rectangles (almost points) and large rectangles into blocks well. We found such a curve, and our experiments show that it leads to R-trees with excellent query performance across several types of data sets.

Keywords: Space-filling curves, Peano curve, R-trees, locality measures

The Geometry of Binary Seach Trees

John Iacono (Polytechnic Univ. - New York, US)

In this talk we discuss a new geometric interpretation of binary search trees. In this geometric view we cast new light on the theorems, conjectures, and lower bounds of binary search trees.

Keywords: Trees, splaying, dynamic optimality conjecture

Optimal Sparse Matrix Dense Vector Multiplication in the I/O-Model

Riko Jacob (TU München, DE)

We analyze the problem of sparse-matrix dense-vector multiplication (SPMV) in the I/O-model. The task of SPMV is to compute $y := Ax$, where A is a sparse $N \times N$ matrix and x and y are vectors. Here, sparsity is expressed by the parameter k that states that A has a total of at most kN nonzeros, i.e., an average number of k nonzeros per column. The extreme choices for parameter k are well studied special cases, namely for $k = 1$ permuting and for $k = N$ dense matrix-vector multiplication.

In the first part, we study the worst-case complexity of this computational task, i.e., what is the best possible upper bound on the number of I/Os depending on k and N only? We determine this complexity up to a constant factor for large ranges of the parameters. By our arguments, we find that most matrices with kN nonzeros require this number of I/Os, even if the program may depend on the structure of the matrix.

The model of computation for the lower bound is a combination of the I/O-models of Aggarwal and Vitter, and of Hong and Kung. We study two variants of the problem, depending on the memory layout of A.
If A is stored in column major layout, SPMVhas I/O complexity

$$\Theta(\min\left\{\frac{kN}{B}\left(1+\log_{M/B}\frac{N}{\max\{M,k\}}\right), kN\right\})$$

for $k \leq N^{1-\varepsilon}$ and any constant $1 > \varepsilon > 0$. If the algorithm can choose the memory layout, the I/O complexity of SPMVis

$$\Theta(\min\left\{\frac{kN}{B}\left(1+\log_{M/B}\frac{N}{kM}\right), kN\right\})$$

for $k \leq \sqrt[3]{N}$.

In the cache oblivious setting with tall cache assumption $M \geq B^{1+\varepsilon}$, the I/O complexity is $O(\frac{kN}{B}\left(1+\log_{M/B}\frac{N}{k}\right))$ for A in column major layout.

This first part is joint work with Michael A. Bender, Gerth Stølting Brodal, Rolf Fagerberg, and Elias Vicari, published at SPAA 2007, San Diego.

In the second part, we consider extensions of the above problem to skew matrices, to the situation where several vectors are to be multiplied with the same matrix, and the case where the access to the matrix does not cost any I/O.

Keywords: External memory algorithms

Optimal Speedup on a Low-Degree Multi-Core Parallel Architecture (LoPRAM)

Alejandro Lopez-Ortiz (University of Waterloo, CA)

We propose a new model with small degreee of parallelism that reflects current and future multicore architectures in practice. The model is based on the PRAM architecture and hence it inherits many of its interesting theoretical properties. The key observations and differences are that the degree of parallelism (i.e. number of processors or cores) is bounded by O(log n), the synchronization model is looser and the use of parallelism is at a higher level unless explicitly specified otherwise. Surprisingly, these three rather minor variants result in a model in which obtaining work optimal algorithms is significantly easier than for the PRAM.

The new model is called Low-degree PRAM or LoPRAM for short. Lastly we observe that there are thresholds in complexity of programming at p=O(log n) and p=O(sqrt(n)) and provide references for specific problems for which this threshold has been formally proven.

Keywords: PRAM, multicore architectures, parallelism, algorithms, dynamic programming, divide and conquer

Extended Abstract: http://drops.dagstuhl.de/opus/volltexte/2008/1531

Updating K-d trees

Conrado Martinez (TU of Catalonia - Barcelona, ES)

In this work we present an in-depth study of randomized relaxed K-d trees concerning two fundamental aspects: the randomized algorithms that allow to preserve the random properties of relaxed K-d trees and the mathematical analysis of the expected performance of these algorithms. In particular, we describe two different versions of randomized update algorithms for K-d trees: updates based on the split and join algorithms of Duch, Estivill-Castro, Martínez (1998) and updates based on the copy-based insertion at the root and deletion of root of Broutin et al. (2006). We carry out a refined analysis of the expected cost of all

these algorithms, complementing the previous analysis given by Broutin et al., and providing the first result for the average cost of copy-based updates when $K > 2$. Specifically, we show that the average cost of split and join is of the form $\zeta(K) \cdot n^{\phi(K)} + \text{l.o.t.}$, with $1 \leq \phi(K) < 1.561552813$, and we give explicit formulæ for both $\zeta(K)$ and $\phi(K)$. We prove also that the average cost of the copy-based updates is of the form $\gamma(K) \cdot n^{\varrho(K)} \ln n + \text{l.o.t.}$, with $\varrho(K) = \rho(1/K) < 1$, where $\rho(x)$ is the exponent that appears in the expected performance of partial matches in relaxed K-d trees.

These results on the average performance of split and join imply that the expected cost of an insertion or deletion is $\Theta(n^{\phi(K)-1})$; the results on the average performance of copy-based updates imply that the expected cost of an insertion or deletion is $\Theta(\log n)$, for any fixed K. The logarithmic costs are related to the search phase in insertions and deletions; the "reconstruction" phase of insertions and deletions using copy-based updates has actually constant expected time.

Keywords: Analytic combinatorics, continuous master theorem, K-dimensional trees, multidimensional data structures, randomized algorithms, relaxed K-d trees

Joint work of: Martinez, Conrado; Duch, Amalia

On Dynamic Breadth-First Search in External-Memory

Ulrich Carsten Meyer (J.W. Goethe Universität Frankfurt, DE)

We provide the first non-trivial result on dynamic breadth-first search (BFS) in external-memory:

For general sparse undirected graphs of initially n nodes and $O(n)$ edges and monotone update sequences of either $\Theta(n)$ edge insertions or $\Theta(n)$ edge deletions, we prove an amortized high-probability bound of $O(n/B^{2/3} + \text{sort}(n) \cdot \log B)$ I/Os per update. In contrast, the currently best approach for static BFS on sparse undirected graphs requires $\Omega(n/B^{1/2} + \text{sort}(n))$ I/Os.

Keywords: External Memory, Dynamic Graph Algorithms, BFS, Randomization

Full Paper:
http://drops.dagstuhl.de/opus/volltexte/2008/1316/

See also: Proc. 25th STACS 2008, pp. 551-560

Reliability Issues in Non-Volatile Memories

Ethan Miller (Univ. California - Santa Cruz, US)

Non-volatile byte addressable memories are becoming more common, and are increasingly used for critical data that must not be lost.

However, existing NVRAM-based file systems do not include features that guard against file system corruption or NVRAM corruption. Furthermore, most file systems check consistency only after the system has already crashed. We are designing a file system to address these problems by providing file storage that can survive multiple errors in NVRAM, whether caused by errant operating system writes or by media corruption. Our file system uses an erasure-encoded log structure to store persistent metadata, making it possible to periodically verify the correctness of file system operations while achieving throughput rates of an order of magnitude higher than page-protection during small writes. It also checks integrity on every operation and performs on-line scans of the entire NVRAM to ensure that the file system is consistent. If errors are found, they can be corrected using file system logs and extensive error correction information.

Keywords: Reliability, non-volatile memory, error correction

Joint work of: Miller, Ethan; Greenan, Kevin

Flash Memory Technology

Sang Lyul Min (Seoul Nat. University, KR)

In this talk, first I will discuss the basics of flash memory that is increasingly being used as a storage medium in mobile devices because of its low power consumption, fast random access, and high shock resistance.

Then, I will explain a software layer called the FTL (flash translation layer) that essentially emulates the functionality of HDD using functionalies provided by flash memory. Finally, I will briefly discuss emerging new non- volatile memories such as FeRAM, MRAM, and PRAM.

Keywords: Flash memory, FTL (Flash Translation Layer), FeRAM, PCM, MRAM

Hydra Solid State Disk Architecture

Sang Lyul Min (Seoul Nat. University, KR)

Flash memory is increasingly being used as a storage medium for mobile devices such as mobile phones, PMPs, and UMPCs because of low power consumption, fast random access, and high shock and vibration resistance. In this talk, first I will explain the basics and current trends of NAND flash memory, the type of flash memory used for storage applications. Second, I will introduce the concept of a flash memory solid state disk (SSD) that provides an interface identical to that of hard disk drive (HDD) out of flash memory. Third, I will explain a new trend called "stateless computing" using SSD's that enables a complete decoupling of state and engine in personal computing as it is in GSM phones using sim cards in personal mobile communication. Finally, I will explain the

motivations, design, implementation, and performance evaluation of the Hydra solid state disk that our research group has been working on for the past three years.

Keywords: Solid State Disk, Flash memory, Stateless PC

Data structures without data

Rasmus Pagh (IT University of Copenhagen, DK)

We consider the use of techniques from data structures in applications where there is no data as such. Some preliminary results, and some challenges, are presented.

Hard data structure problems

Mihai Patrascu (MIT - Cambridge, US)

We discuss several dynamic problems for which we do not currently know any poly-logarithmic solution. Should we keep looking for a better algorithm, or is there a fundamental limitation that makes poly-log running times impossible?

We propose a "core hard problem" with the following features:

* intuitively, we think that a polylogarithmic solution will be impossible for this problem.

* the problem reduces to the interesting problems considered before, showing that a polylog solution is unlikely for these also.

* we can describe a complexity-theoretic approach which may lead to a lower bound for our core problem.

Keywords: Lower bounds, complexity

Non-linearity in Davenport-Schinzel Sequences

Seth Pettie (Univ. of Michigan - Ann Arbor, US)

A *generalized* Davenport-Schinzel sequence is one over a finite alphabet that contains no subsequences isomorphic to a fixed *forbidden subsequence*. One of the fundamental problems in this area is bounding (asymptotically) the maximum length of such sequences.

Following Klazar, let $\mathrm{Ex}(\sigma, n)$ be the maximum length of a sequence over an alphabet of size n avoiding subsequences isomorphic to σ. It has been proved that for every σ, $\mathrm{Ex}(\sigma, n)$ is either linear or very close to linear; in particular it is $O(n2^{\alpha(n)^{O(1)}})$, where α is the inverse-Ackermann function and $O(1)$ depends on σ. However, very little is known about the properties of σ that induce superlinearity of $\mathrm{Ex}(\sigma, n)$.

In this paper we exhibit an infinite family of independent superlinear forbidden subsequences. To be specific, we show that there are 17 *prototypical* superlinear forbidden subsequences, some of which can be made arbitrarily long through a simple padding operation.

Perhaps the most novel part of our constructions is a new succinct code for representing superlinear forbidden subsequences.

Keywords: Davenport-Schinzel Sequences, lower envelopes, splay trees

Full Paper: http://drops.dagstuhl.de/opus/volltexte/2008/1529

Computing Minimum Spanning Trees with Uncertainty

Rajeev Raman (University of Leicester, GB)

We consider the minimum spanning tree problem in a setting where information about the edge weights of the given graph is uncertain. Initially, for each edge e of the graph only a set A_e, called an uncertainty area, that contains the actual edge weight we is known.

The algorithm can 'update' e to obtain the edge weight $w_e \in A_e$. The task is to output the edge set of a minimum spanning tree after a minimum number of updates. An algorithm is k-update competitive if it makes at most k times as many updates as the optimum. We present a 2-update competitive algorithm if all areas A_e are open or trivial, which is the best possible among deterministic algorithms. The condition on the areas Ae is to exclude degenerate inputs for which no constant update competitive algorithm can exist.

Next, we consider a setting where the vertices of the graph correspond to points in Euclidean space and the weight of an edge is equal to the distance of its endpoints. The location of each point is initially given as an uncertainty area, and an update reveals the exact location of the point. We give a general relation between the edge uncertainty and the vertex uncertainty versions of a problem and use it to derive a 4-update competitive algorithm for the minimum spanning tree problem in the vertex uncertainty model. Again, we show that this is best possible among deterministic algorithms.

Keywords: Computing with Uncertainty, Minimum Spanning Tree, Competitive analysis

Joint work of: Erlebach, Thomas; Hoffmann, Michael; Krizanc, Danny; Mihalak, Matus; Raman, Rajeev

Full Paper:
http://drops.dagstuhl.de/opus/volltexte/2008/1358

Compressed Inverted Indexes for In-Memory Search Engines

Peter Sanders (Universität Karlsruhe, DE)

We present the algorithmic core of a full text data base that allows fast Boolean queries, phrase queries, and document reporting using less space than the input text. The system uses a carefully choreographed combination of classical data compression techniques and inverted index based search data structures. In our experiments it outperforms compressed suffix array based techniques for all the above operations for real world (natural language) texts.

Keywords: Search engine, inverted index, compressed suffix array, algorithm engineering

Full Paper:
http://www.siam.org/proceedings/alenex/2008/alenex08.php

Left-Leaning Red-Black Trees

Robert Sedgewick (Princeton University, US)

The red-black tree model for implementing balanced search trees, introduced by Guibas and Sedgewick 30 years ago, is now found throughout our computational infrastructure. Red-black trees are the underlying data structure in the symbol-table implementation in C++, Java, Python, BSD Unix, and many other modern systems. However, current implementations have sacrificed some of the original design goals (primarily in order to develop an effective delete implementation, which was incompletely specified in the original paper), so a new look is worthwhile.

In this talk, we describe a new variant of red-black trees that meets many of the original design goals and leads to substantially simpler code for insert/delete, less than one-fourth as much code as in implementations in common use.

Keywords: Balanced trees, search algorithms, red-black trees

Dynamic Topological Ordering

Robert E. Tarjan (Princeton University, US)

We consider the problem of maintaining a topological order of a directed acyclic graph under arc additions, and the related problems of reporting a cycle when one is created and maintaining the strong components of the graph. By modifying an existing approach, we obtain an algorithm with an amortized time bound of $O(m^{1/2})$ per arc addition, if the total number of arcs added is m. For sparse graphs, this improves the previous best algorithm by a logarithmic factor. This is joint work with Bernhard Haeupler and Siddhartha Sen.

Keywords: Dynamic graph algorithms, DAG

I/O-Efficient Map Overlay and Point Location in Low-Density Subdivisions

Laura I. Toma (Bowdoin College - Brunswick, US)

We present improved and simplified I/O-efficient algorithms for two problems on planar low-density subdivisions, namely map overlay and point location. More precisely, we show how to preprocess a low-density subdivision with n edges in $O(sort(n))$ I/Os into a compressed linear quadtree such that one can:

(1) compute the overlay of two such preprocessed subdivisions in $O(scan(n))$ I/Os, where n is the total number of edges in the two subdivisions,

(2) answer a single point location query in $O(\log_B n)$ I/Os and k batched point location queries in $O(scan(n) + \text{sort}(k))$ I/Os.

For the special case where the subdivision is a fat triangulation, we show how to obtain the same bounds with an ordinary (uncompressed) quadtree, and we show how to make the structure fully dynamic using $O(\log_B n)$ I/Os per update. Our algorithms and data structures improve on the previous best known bounds for general subdivisions both in the number of I/Os and storage usage, they are significantly simpler, and several of our algorithms are cache-oblivious.

Keywords: I/O-efficient, map overlay, point location

Joint work of: de Berg, Mark; Haverkort, Herman; Thite, Shripad; Toma, Laura

Klee's Measure Problem Revisited: Two Notes and an Open Problem

Jan Vahrenhold (Technische Universität Dortmund, DE)

1. We present the first in-place algorithm for solving Klee's measure problem for a set of n axis-parallel rectangles in the plane. Our algorithm runs in $O(n^{3/2} \log n)$ time and uses $O(1)$ extra words in addition to the space needed for representing the input. The algorithm is surprisingly simple and thus very likely to yield an implementation that could be of practical interest. As a byproduct, we develop an optimal algorithm for solving Klee's measure problem for a set of n intervals; this algorithm runs in optimal time $O(n \log n)$ and uses $O(1)$ extra space.

2. Computing the hypervolume indicator (used in multicriteria optimization) has been shown to be a special case of solving Klee's measure problem. For this special case, we obtain a lower bound of $\Omega(n \log n)$ for the complexity of computing the hypervolume indicator in any dimension $d > 1$. For the three dimensional case, an $O(n \log n)$ algorithm is given, which is derived from an algorithm for finding maxima of a point set.

(2) is joint work with Nicola Beume, Carlos M. Fonseca, Manuel Lopez-Ibanez, and Luis Paquete.

Keywords: Klee's Measure Problem, Pareto-Optimal Points, Hypervolume Indicator, In-Place Algorithms

Efficient algorithms for universal denoising

Alfredo Viola (INCO - Montevideo, UY)

The classical framework of context-tree models used in sequential decision problems such as compression and prediction has been recently generalized to a setting where the observations are multidimentional. Context set definitions tree representations and prunning algorithms have recently been extended from the classical unidirectional setting to a bidimensional one. In this case, it may be beneficial to consider contexts comprised of possibly different number of symbols from each direction. In this talk we present an efficient algorithm to implement this framework in a two dimensional setting, with special application to universal decoding. This is a joint work with Marcelo Weinberger (HP Labs, California) and Fernando Fernandez (Universidad de la Republica, Uruguay).

Keywords: Suffix trees, bidirectional graphs, context-tree models

Cache-oblivious red-blue line segment intersection

Norbert Zeh (Dalhousie University, CA)

Given a set R of nonintersecting red segments and a set B of nonintersecting blue segments in the plane, we present a cache-oblivious algorithm that finds all intersections in between red and blue segments in $O(\text{sort}(N) + T/B)$ memory transfers, where N is the total number of segments, $\text{sort}(N)$ denotes the cost of sorting N elements cache-obliviously, T is the number of reported intersections, and B is the cache block size. This algorithm is optimal and is the first efficient cache-oblivious algorithm for an intersection problem in the plane that involves non-axis-aligned objects.

Keywords: External memory algorithms, segment intersection

Joint work of: Arge, Lars; Moelhave, Thomas; Zeh, Norbert