

Data Structures and Advanced Models of Computation on Big Data

Edited by

Alejandro López-Ortiz¹, Ulrich Carsten Meyer², and Robert Sedgewick³

1 University of Waterloo, CA, alopez-o@uwaterloo.ca

2 Goethe-Universität Frankfurt am Main, DE, umeyer@cs.uni-frankfurt.de

3 Princeton University, US, rs@cs.princeton.edu

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 14091 “Data Structures and Advanced Models of Computation on Big Data”. In today’s computing environment vast amounts of data are processed, exchanged and analyzed. The manner in which information is stored profoundly influences the efficiency of these operations over the data. In spite of the maturity of the field many data structuring problems are still open, while new ones arise due to technological advances.

The seminar covered both recent advances in the “classical” data structuring topics as well as new models of computation adapted to modern architectures, scientific studies that reveal the need for such models, applications where large data sets play a central role, modern computing platforms for very large data, and new data structures for large data in modern architectures.

The extended abstracts included in this report contain both recent state of the art advances and lay the foundation for new directions within data structures research.

Seminar February 24–28, 2014 – <http://www.dagstuhl.de/14091>

1998 ACM Subject Classification E.1 Data Structures, F.1 Computation by Abstract Devices, F.2 Analysis of Algorithms and Problem Complexity, G.2 Discrete Mathematics

Keywords and phrases data structures, big data, models of computation, I/O Model, sorting, quicksort, graph algorithms, hashing, compression, succinct data structures, trajectories, text search, GPU algorithms, MapReduce

Digital Object Identifier 10.4230/DagRep.4.2.129

Edited in cooperation with Timo Bingmann

1 Executive Summary

Alejandro López-Ortiz

Ulrich Carsten Meyer

Robert Sedgewick

License © Creative Commons BY 3.0 Unported license
© Alejandro López-Ortiz, Ulrich Carsten Meyer, and Robert Sedgewick

A persistent theme in the presentations in this Dagstuhl seminar is the need to refine our models of computation to adapt to modern architectures, if we are to develop a scientific basis for inventing efficient algorithms to solve real-world problems. For example, Mehlhorn’s presentation on the cost of memory translation, Iacono’s reexamination of the cache-oblivious model, and Sanders’ description of communication efficiency all left many participants



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Data Structures and Advanced Models of Computation on Big Data, *Dagstuhl Reports*, Vol. 4, Issue 2, pp. 129–149

Editors: Alejandro López-Ortiz, Ulrich Carsten Meyer, and Robert Sedgewick



DAGSTUHL
REPORTS

Dagstuhl Reports
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

questioning basic assumptions they have carried for many years and are certain to stimulate new research in the future.

Better understanding of the properties of modern processors certainly can be fruitful. For example, several presentations, such as the papers by Aumüller, López-Ortiz, and Wild on Quicksort and the paper by Bingmann on string sorting, described faster versions of classic algorithms that are based on careful examination of modern processor design.

Overall, many presentations described experience with data from actual applications. For example, the presentations by Driemel and Varenhold on trajectory data described a relatively new big-data application that underscores the importance and breadth of application of classic techniques in computational geometry and data structure design.

Other presentations which discussed large data sets on modern architectures were the lower bound on parallel external list ranking by Jacob, which also applies on the MapReduce and BSP models commonly used in large distributed platforms; and by Hagerup who considered the standard problem of performing a depth first search (DFS) on a graph, a task that is trivial in small graphs but extremely complex on “big data” sets such as the Facebook graph. He proposed a space efficient algorithm that reduces the space required by DFS by a $\log n$ factor or an order of magnitude on practical data sets.

Schweikardt gave a model for MapReduce computations, a very common computing platform for very large server farms. Salinger considered the opposite end of the spectrum namely how to simplify the programming task as to take optimal advantage of a single server which also has its own degree of parallelism from multiple cores, GPUs and other parallel facilities.

In terms of geometric data structures for large data sets Afshani presented sublinear algorithms for the I/O model which generalize earlier work on sublinear algorithms. Sublinear algorithms are of key importance on very large data sets, which are thus presumably unable to fit in main memory. Yet most of the previously proposed algorithms assumed that such large data sets were hosted in main memory. Toma gave an external memory representation of the popular quad tree data structure commonly used in computer graphics as well as other spatial data applications.

2 Table of Contents

Executive Summary

Alejandro López-Ortiz, Ulrich Carsten Meyer, and Robert Sedgwick 129

Overview of Talks

Sublinear Geometric Algorithms in the I/O Model <i>Peyman Afshani</i>	133
Distance Oracles for Large Real-World Graphs <i>Deepak Ajwani</i>	133
RAM-Efficient External Memory Algorithms <i>Lars Arge</i>	134
Optimal Partitioning for Multi-Pivot Quicksort <i>Martin Aumüller</i>	134
Semantic Search on Big Data <i>Hannah Bast</i>	134
Parallel String Sorting with Super Scalar String Sample Sort <i>Timo Bingmann</i>	135
Range Minimum Queries (Part II) <i>Gerth Stølting Brodal</i>	135
ERa – A Practical Approach to Parallel Construction of Suffix Trees <i>Andrej Brodnik</i>	135
Data Structures for Trajectories <i>Anne Driemel</i>	136
Biased Search for Bounded Universes <i>Rolf Fagerberg</i>	136
High-Order Entropy Compressed Bit Vectors with Rank/Select <i>Johannes Fischer</i>	137
Space-Economical Depth-First Search <i>Torben Hagerup</i>	137
Algorithms for Graph Connectivity Problems – Connectivity Augmentation, All-Pairs Minimum Cut, and Cohesive Subset Decomposition <i>Tanja Hartmann</i>	137
Space-filling Curves for 3D Mesh Traversals <i>Herman J. Haverkort</i>	138
Path Queries in Weighted Trees <i>Meng He</i>	138
Does the Cache-Oblivious Model Make Any Sense? <i>John Iacono</i>	138
The Parallel External Memory Complexity of List Ranking <i>Riko Jacob</i>	139
Faster Clustering via Preprocessing <i>Tsvi Kopelowitz</i>	139

Multi-Pivot Quicksort: Theory and Experiments <i>Alejandro López-Ortiz</i>	139
The Cost of Address Translation <i>Kurt Mehlhorn</i>	140
When Less is More – Energy-Efficient Sorting <i>Ulrich Carsten Meyer</i>	140
Outperforming LRU via Competitive Analysis on Parametrized Inputs <i>Gabriel Moruz</i>	141
Succinct Data Structures for Representing Equivalence Classes <i>J. Ian Munro</i>	141
On the Energy Consumption of Sorting and Searching <i>Markus E. Nebel</i>	142
Lattice Compression <i>Patrick K. Nicholson</i>	142
Expected Linear Time Sorting for Word Size $\Omega(\log^2 n \log \log n)$ <i>Jesper A. Sindahl Nielsen</i>	142
Gunrock: High-Performance, High-Level Iterative Graph Computation on GPUs <i>John Owens</i>	143
One Answer, Two Questions, and Some Very Old Slides <i>Rasmus Pagh</i>	143
Encoding Top- k and Range Selection <i>Rajeev Raman</i>	143
Algorithms in the Ultra-Wide Word Model <i>Alejandro Salinger</i>	144
Communication Efficient Algorithms <i>Peter Sanders</i>	144
New Worst-Case Approaches in Robust Optimization <i>Anita Schöbel</i>	145
MapReduce Transducers: A Formal Model for Distributed Streaming Computations <i>Nicole Schweikardt</i>	145
Counting Arbitrary Subgraphs in Data Streams <i>He Sun</i>	146
Strings, Geometry, Special Cases... <i>Sharma V. Thankachan</i>	146
An Edge Quadtree for External Memory <i>Laura I. Toma</i>	146
Of Motifs and Goals: Mining Trajectory Data <i>Jan Vahrenhold</i>	147
Dual-Pivot Quicksort – Asymmetries in Sorting <i>Sebastian Wild</i>	147
The Link Assessment Problem of Low Intensity Relationships in Complex Networks <i>Katharina A. Zweig</i>	148
Participants	149

3 Overview of Talks

3.1 Sublinear Geometric Algorithms in the I/O Model

Peyman Afshani (Aarhus University, DK)

License  Creative Commons BY 3.0 Unported license
© Peyman Afshani

Joint work of Afshani, Peyman; Matulef, Kevin; Wilkinson, Bryan

Being able to handle massive data is one of the main motivations to consider sublinear algorithms. Typically in these problems, the data is assumed to be stored in a standard representation and with no extra assumptions and thus with no extra preprocessing. The goal is to perform certain computational tasks in sublinear time, (i.e., without reading the whole input) and this is almost always done using a randomized algorithm. Despite the fact that the motivation comes from massive data, most of the previous results in sublinear algorithms were obtained in internal memory. We observe that building sublinear algorithms in the I/O model results in dealing with novel and interesting challenges. We introduce some new sublinear algorithms in the I/O model. These include, searching a linked list for an element, and detecting the intersection of two polygons. We prove that our results are optimal up to an extra log factor. Obtaining further sublinear algorithms in the I/O model is an obvious area left open by this work.

3.2 Distance Oracles for Large Real-World Graphs


Deepak Ajwani (Bell Labs – Dublin, IE)

License  Creative Commons BY 3.0 Unported license
© Deepak Ajwani

Processing and analyzing massive real-world graphs rapidly has emerged as an important challenge in data analysis in recent years. Many important graph processing algorithms, such as p -center clustering, require computation of shortest-path distances between arbitrary numbers of node pairs in the graph. Since computation of exact distances between all node-pairs of an arbitrary large graph (e.g., 10M nodes and up) is prohibitively expensive both in computational time and storage space, distance approximation is often used in place of exact computation. A distance oracle is a data structure that answers inter-point distance queries in $o(n^2)$ time and storage space, where n is the number of nodes in the graph. While there is a rich body of theoretical literature on approximate distance oracles with worst-case bounds, there are only a handful of (mostly recent) approaches that are known to provide strong approximation in practice on real-world graphs. In this talk, we present theoretical evidence that for many real-world graphs, with intrinsic small hyperbolicity structure, we can naturally leverage this topological feature to build a high fidelity distance approximation oracle. We show that such an oracle (i) can be constructed efficiently, (ii) requires $O(n)$ words of storage space and (iii) results in a small average additive distortion of distance.

3.3 RAM-Efficient External Memory Algorithms

Lars Arge (Aarhus University, DK)

License  Creative Commons BY 3.0 Unported license
© Lars Arge

In recent years a large number of problems have been considered in external memory models of computation, where the complexity measure is the number of blocks of data that are moved between slow external memory and fast internal memory (also called I/Os). In practice, however, internal memory time often dominates the total running time once I/O-efficiency has been obtained. In this talk we discuss sorting algorithms that are simultaneously I/O-efficient and internal memory efficient in the RAM model of computation. We also describe interesting open problems in particular in relation to batched geometric problems.

3.4 Optimal Partitioning for Multi-Pivot Quicksort

Martin Aumüller (TU Ilmenau, DE)

License  Creative Commons BY 3.0 Unported license
© Martin Aumüller

Joint work of Aumüller, Martin; Dietzfelbinger, Martin; Klaue, Pascal

Main reference M. Aumüller, M. Dietzfelbinger, “Optimal Partitioning for Dual Pivot Quicksort,” in Proc. of the 40th Int’l Colloquium on Automata, Languages, and Programming (ICALP’13), LNCS, Vol. 7965, pp. 33–44, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-39206-1_4

This talk considers variants of classical Quicksort where k pivots are used to split the input into $k + 1$ segments. This can be done in different ways, giving rise to different algorithms. In the first part of this talk, we consider Dual-Pivot Quicksort. We study three different algorithms which all achieve the lowest possible average comparison count w.r.t. to the leading term: $1.8 n \ln n$. This improves on the average comparison count of $1.9 n \ln n$ achieved by Yaroslavskiy’s algorithm, the standard internal sorting algorithm in Java 7. The second part of this talk will consider generalizations which use more than two pivots. Surprisingly, preliminary results indicate that the average comparison count of optimal k pivot Quicksort almost coincides with the average comparison count achieved by Median-of- k Quicksort.

3.5 Semantic Search on Big Data

Hannah Bast (Universität Freiburg, DE)

License  Creative Commons BY 3.0 Unported license
© Hannah Bast

I presented a demo of our prototype for semantic full-text search. It provides a deep integration of classical full-text search (in the demo: on the English Wikipedia) with search in knowledge bases (in the demo: Freebase). I the discussed five major challenges in the context of big data: (1) getting hold of big data, (2) making sense of big data, (3) fast query processing via tailor-made index data structures, (4) Quality Evaluation with Amazon Mechanical Turk, and (4) enabling reproducibility of results via a web application.

3.6 Parallel String Sorting with Super Scalar String Sample Sort

Timo Bingmann (KIT – Karlsruhe Institute of Technology, DE)

License © Creative Commons BY 3.0 Unported license
© Timo Bingmann

Joint work of Bingmann, Timo; Sanders, Peter

Main reference T. Bingmann, P. Sanders, “Parallel String Sample Sort,” in Proc. of the 21st Annual European Symp. on Algorithms (ESA’13), LNCS, Vol. 8125, pp. 169–180, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-40450-4_15

We present the currently fastest parallel string sorting algorithm for modern multi-core shared memory architectures. First, we describe the challenges posed by these new architectures, and discuss key points to achieving high performance gains. Then we give an overview of existing sequential and parallel string sorting algorithms and implementations. Thereafter, we continue by developing super scalar string sample sort (S^5), which is easily parallelizable and yields higher parallel speedups than all previously known algorithms. For NUMA systems, we also given an outlook into parallel K -way LCP-aware merging of presorted sequences with LCPs.

3.7 Range Minimum Queries (Part II)

Gerth Stølting Brodal (Aarhus University, DK)

License © Creative Commons BY 3.0 Unported license
© Gerth Stølting Brodal

Main reference G. S. Brodal, A. Brodnik, P. Davoodi, “The Encoding Complexity of Two Dimensional Range Minimum Data Structures,” in Proc. of the 21st Annual European Symp. on Algorithms (ESA’13), LNCS, Vol. 8125, pp. 229–240, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-40450-4_20

In the two-dimensional range minimum query problem an input matrix A of dimension $m \times n$, $m \leq n$, has to be preprocessed into a data structure such that given a query rectangle within the matrix, the position of a minimum element within the query range can be reported. We consider the space complexity of the encoding variant of the problem where queries have access to the constructed data structure but can not access the input matrix A , i.e., all information must be encoded in the data structure. Previously it was known how to solve the problem with space $O(mn \min\{m, \log n\})$ bits (and with constant query time), but the best lower bound was $\Omega(mn \log m)$ bits, i.e., leaving a gap between the upper and lower bounds for non-quadratic matrices. We show that this space lower bound is optimal by presenting an encoding scheme using $O(mn \log m)$ bits. We do not consider query time

3.8 ERa – A Practical Approach to Parallel Construction of Suffix Trees

Andrej Brodnik (University of Primorska, SI)

License © Creative Commons BY 3.0 Unported license
© Andrej Brodnik

Joint work of Brodnik, Andrej; Jekovec, Matevž

URL http://lusy.fri.uni-lj.si/sites/lusy.fri.uni-lj.si/files/publications/era-dagstuhl_2014.pdf

The most frequently used data structure to answer queries *where* and *how often* a pattern appears in a preprocessed text is a suffix tree. Therefore, it is an important challenge to

design practically efficient algorithms to construct large suffix trees. Moreover, computers nowadays contain processors with 32 and even more cores, which, in case of human genome of length $n = 2.8$ Gbp, represents essentially $\log n$ processors. In this contribution we analyze an efficient parallel algorithm called ERa (Elastic Range). In our presentation we first give a quick overview of known theoretically efficient suffix tree construction algorithms that are supplemented by a brief survey of suffix tree construction algorithms used in practice. The survey is followed by a more detailed presentation of ERa, currently the best known parallel algorithm for suffix tree construction in practice. A short theoretical analysis of ERa follows the presentation. The concluding part gives an experimental evaluation of ERa applied to a human genome. We show that the bottleneck for running time of the algorithm is indeed I/O and in particular writing the resulting suffix tree to the disk. We also show, that further improvements in scheduling of parallel construction of suffix subtrees is possible. We conclude with open challenges that include a design of theoretically efficient and practically fast algorithm.

3.9 Data Structures for Trajectories

Anne Driemel (TU Dortmund, DE)

License © Creative Commons BY 3.0 Unported license
© Anne Driemel

Joint work of Driemel, Anne; Har-Peled, Sariel

Main reference A. Driemel, S. Har-Peled, “Jaywalking Your Dog: Computing the Fréchet Distance with Shortcuts,” SIAM Journal on Computing, 42(5):1830–1866, September 2013.

URL <http://dx.doi.org/10.1137/120865112>

The Fréchet distance is a popular distance measure for trajectories of moving objects. It can be used for map-matching a trajectory to a street network, for finding similar subtrajectories and for clustering trajectories. So far, only few data structures have been proposed to support such operations efficiently. We develop several new tools: (a) a data structure for preprocessing a curve that supports $(1 + \varepsilon)$ -approximate Fréchet distance queries between a subcurve (of the original curve) and a line segment; (b) a near linear time algorithm that computes a permutation of the vertices of a curve, such that any prefix of $2k - 1$ vertices of this permutation forms an optimal approximation (up to a constant factor) to the original curve compared to any polygonal curve with k vertices, for any $k > 0$; and (c) a data structure for preprocessing a curve that supports approximate Fréchet distance queries between a subcurve and query polygonal curve. The query time depends quadratically on the complexity of the query curve and only (roughly) logarithmically on the complexity of the original curve. To our knowledge, these are the first data structures to support these kind of queries efficiently.

3.10 Biased Search for Bounded Universes

Rolf Fagerberg (University of Southern Denmark – Odense, DK)

License © Creative Commons BY 3.0 Unported license
© Rolf Fagerberg

We consider the problem of performing predecessor searches in a bounded universe while achieving query times that depend on the distribution of queries. We obtain several data

structures with various properties: in particular, we give data structures that achieve expected query times logarithmic in the entropy of the distribution of queries but with space bounded in terms of universe size, as well as data structures that use only linear space but with query times that are higher (but still sublinear) functions of the entropy. For these structures, the distribution is assumed to be known. We also consider data structures with general weights on universe elements, as well as the case when the distribution is not known in advance.

3.11 High-Order Entropy Compressed Bit Vectors with Rank/Select


Johannes Fischer (TU Dortmund, DE)

License  Creative Commons BY 3.0 Unported license
© Johannes Fischer

We show how to compress bit-vectors to support efficient rank-queries (counting the number of ones up to a given point). Unlike previous approaches, which either store the bit vectors plainly, or focus on compressing bit-vectors with low densities of ones or zeros, we aim at low entropies of higher order, for example 101010...10. Our implementations achieve very good compression ratios, while showing only a modest increase in lookup time.

3.12 Space-Economical Depth-First Search


Torben Hagerup (Universität Augsburg, DE)

License  Creative Commons BY 3.0 Unported license
© Torben Hagerup
Joint work of Hagerup, Torben; Elmasry, Amr; Kammer, Frank

As part of a wider investigation of space-bounded RAM computation, we consider the special problem of depth-first search (DFS) on a RAM with a read-only input, a small working memory and a write-only output. The usual recursive DFS algorithm needs $\Theta(n \log n)$ bits of stack memory and $O(n + m)$ time to process an input graph with n vertices and m edges. We show how to reduce the number of bits of working memory needed to $O(n \log \log n)$ while preserving a linear running time. Alternatively, $O(n)$ bits suffice if a running time of $O((n + m) \log \log n)$ is acceptable, and a tradeoff between these two extremes is possible. The latter result depends on a dynamic dictionary for small universes.

3.13 Algorithms for Graph Connectivity Problems – Connectivity Augmentation, All-Pairs Minimum Cut, and Cohesive Subset Decomposition


Tanja Hartmann (KIT – Karlsruhe Institute of Technology, DE)

License  Creative Commons BY 3.0 Unported license
© Tanja Hartmann

Challenges, results and open problems on three topics related to graph connectivity – planar k -regular connectivity augmentation, all-pairs minimum cut and Gomory-Hu trees in dynamic scenarios, and cohesive subset decomposition (which is similar to graph clustering problems).

3.14 Space-filling Curves for 3D Mesh Traversals

Herman J. Haverkort (TU Eindhoven, NL)

License  Creative Commons BY 3.0 Unported license
© Herman J. Haverkort

Two-dimensional finite element methods require repeated traversals of a mesh of squares or triangles. By processing the mesh elements in order along a well-chosen space-filling curve, one can avoid storing the mesh in complicated and cache-inefficient data structures: only a small number of stacks are needed. For three-dimensional meshes, however, known space-filling curves do not quite suffice. In this presentation I present our latest results on what desirable properties of such space-filling curves can and cannot be realized.

3.15 Path Queries in Weighted Trees

Meng He (Dalhousie University, CA)

License  Creative Commons BY 3.0 Unported license
© Meng He

Main reference M. He, J. I. Munro, G. Zhou, “Path Queries in Weighted Trees,” in Proc. of the 22nd Int’l Symp. on Algorithms and Computation, LNCS, Vol. 7074, pp. 140–149, Springer, 2011.

URL http://dx.doi.org/10.1007/978-3-642-25591-5_16

This talk presents new results on several path queries in trees in which each node is assigned a weight value. This is motivated by the needs of the manipulation of tree-structured data such as XML trees and tree network topology. One such query is path counting, in which we are given an arbitrary query path defined by two nodes and a query weight range, and the goal is to count the number of nodes along this path whose weights are within the given range. We designed succinct data structures that can answer path counting in optimal time. More precisely, our structure occupies $nH(W_T) + o(n \lg \sigma)$ bits, where n denotes the number of nodes in the tree, $H(W_T)$ denotes the entropy of the node weights, and σ is the distinct number of weights, and answers queries in $O(\lg \sigma / \lg \lg n + 1)$ time. Several other path queries can be defined in a similar way, including path median, path min and path reporting, and we achieved new results for these problems. All these generalize queries in 2D point sets and arrays to trees. The above results are from three papers. The first two are co-authored by Meng He, J. Ian Munro and Gelin Zhou, and Timothy M. Chan is also a coauthor of the third article.

3.16 Does the Cache-Oblivious Model Make Any Sense?

John Iacono (Polytechnic Institute of NYU – Brooklyn, US)

License  Creative Commons BY 3.0 Unported license
© John Iacono

Several limitations of the cache-oblivious model are presented, some well-known, and others less so. However, algorithms in the cache-oblivious model perform well because they lead to optimizing locality of reference. We then explore how one might want to more directly model a modern computer using locality of reference as a central paradigm.

3.17 The Parallel External Memory Complexity of List Ranking

Riko Jacob (*ETH Zürich, CH*)

License © Creative Commons BY 3.0 Unported license
© Riko Jacob

Joint work of Jacob, Riko; Lieber, Tobias; Sitchinava, Nodari

We study the problem of list ranking in the PEM model. We present an $\Omega(\log^2 N)$ (for $P = M$, $N = MB$) lower bound for a non-standard variant of the problem. This variant is also difficult in the bulk synchronous parallel (BSP) and MapReduce models.

3.18 Faster Clustering via Preprocessing

Tsvi Kopelowitz (*University of Michigan – Ann Arbor, US*)

License © Creative Commons BY 3.0 Unported license
© Tsvi Kopelowitz

Joint work of Kopelowitz, Tsvi; Krauthgamer, Robert

Main reference T. Kopelowitz, R. Krauthgamer, “Faster Clustering via Preprocessing,” arXiv:1208.5247v1 [cs.DS], 2012.

URL <http://arxiv.org/abs/1208.5247v1>

We examine the efficiency of clustering a set of points, when the encompassing metric space may be preprocessed in advance. In computational problems of this genre, there is a first stage of preprocessing, whose input is a point set M and a distance function $d(\cdot, \cdot)$; the next stage receives as input a query point set Q , and should report a clustering of Q according to some objective, such as 1-median, in which case the answer is a point $a \in M$ minimizing $\sum_{q \in Q} d(a, q)$. We design algorithms that solve such problems within $(1 + \varepsilon)$ -approximation under standard clustering objectives like p -center and p -median, and are fast when the metric M has low doubling dimension. By leveraging the preprocessing stage, our algorithms achieve query time that is near-linear in the query size $n = |Q|$, and is (almost) independent of the total number of points $m = |M|$. Moreover, our preprocessing data structure supports updates to M , and its storage requirement is a truly linear $O(m)$ independent of the doubling dimension.

3.19 Multi-Pivot Quicksort: Theory and Experiments

Alejandro López-Ortiz (*University of Waterloo, CA*)

License © Creative Commons BY 3.0 Unported license
© Alejandro López-Ortiz

Joint work of Kushagra, Shrinu; López-Ortiz, Alejandro; Munro, J. Ian; Qiao, Aurick

Main reference S. Kushagra, A. López-Ortiz, A. Qiao, J. I. Munro, “Multi-Pivot Quicksort: Theory and Experiments,” in Proc. of the 16th Workshop on Algorithm Engineering and Experiments (ALENEX’14), pp. 47–60, SIAM, 2014.

URL <http://dx.doi.org/10.1137/1.9781611973198.6>

The idea of multi-pivot quicksort has recently received the attention of researchers after Vladimir Yaroslavskiy proposed a dual pivot quicksort algorithm that, contrary to prior intuition, outperforms standard quicksort by a significant margin under the Java JVM. More recently, this algorithm has been analysed in terms of comparisons and swaps by Wild and Nebel. Our contributions to the topic are as follows. First, we perform the previous experiments using a native C implementation thus removing potential extraneous effects

of the JVM. Second, we provide analyses on cache behavior of these algorithms. We then provide strong evidence that cache behavior is causing most of the performance differences in these algorithms. Additionally, we build upon prior work in multi-pivot quicksort and propose a 3-pivot variant that performs very well in theory and practice. We show that it makes fewer comparisons and has better cache behavior than the dual pivot quicksort in the expected case. We validate this with experimental results, showing a 7–8% performance improvement in our tests.

3.20 The Cost of Address Translation

Kurt Mehlhorn (MPI für Informatik – Saarbrücken, DE)

License © Creative Commons BY 3.0 Unported license
© Kurt Mehlhorn

Joint work of Jurkiewicz, Tomasz; Mehlhorn, Kurt

Main reference T. Jurkiewicz, K. Mehlhorn, “The Cost of Address Translation,” arXiv:1212.0703v2 [cs.DS], 2012.

URL <http://arxiv.org/abs/1212.0703v2>

Modern computers are not random access machines (RAMs). They have a memory hierarchy, multiple cores, and a virtual memory. We address the computational cost of the address translation in the virtual memory. Starting point for our work on virtual memory is the observation that the analysis of some simple algorithms (random scan of an array, binary search, heapsort) in either the RAM model or the EM model (external memory model) does not correctly predict growth rates of actual running times. We propose the VAT model (virtual address translation) to account for the cost of address translations and analyze the algorithms mentioned above and others in the model. The predictions agree with the measurements. We also analyze the VAT-cost of cache-oblivious algorithms.

3.21 When Less is More – Energy-Efficient Sorting

Ulrich Carsten Meyer (Goethe-Universität Frankfurt am Main, DE)

License © Creative Commons BY 3.0 Unported license
© Ulrich Carsten Meyer

Joint work of Beckmann, Andreas; Meyer, Ulrich; Sanders, Peter; Singler, Johannes

Main reference A. Beckmann, U. Meyer, P. Sanders, J. Singler, “Energy-efficient sorting using solid state disks,” in *Sustainable Computing: Informatics and Systems*, 1(2):151–163, June 2011.

URL <http://dx.doi.org/10.1016/j.suscom.2011.02.004>

The energy efficiency of data processing is becoming more and more important for both economical and ecological reasons. In this talk, we take sorting of large data sets as a representative case study for data-intensive applications. We review our system for the JouleSort competitions 2010/11. Guided by theoretical algorithmic considerations and taking practical limitations into account, we carefully chose the components for building an energy-efficient computer for this task. These decisions were backed up by performance and power measurements of several competing options. Finally, we chose a low-power Intel Atom 330 processor, supported by four solid state disks, which have little power consumption and provide high bandwidths. Our sophisticated implementation of the sorting algorithms did not only feature great CPU efficiency. By employing overlapping, it loads all available resources in parallel, resulting in a good overall balance between I/O and computation. Using this setup, we have beaten the former records in the JouleSort category of the well-established

Sort Benchmark for inputs from 10 GB to 1 TB of data, by factors of up to 5.1. This usually comes without a penalty in running time. We also speculate on the consequences of future hardware for the Sort Benchmark contest, and identify certain problems, also relating to the monetary cost of energy.

3.22 Outperforming LRU via Competitive Analysis on Parametrized Inputs

Gabriel Moruz (Universität Frankfurt, DE)

License © Creative Commons BY 3.0 Unported license

© Gabriel Moruz

Main reference G. Moruz, A. Negoescu, “Outperforming LRU via Competitive Analysis on Parametrized Inputs for Paging,” in Proc. of the 23rd Annual ACM-SIAM Symp. on Discrete Algorithms (SODA’12), pp. 1669–1680, SIAM 2012.

URL <http://dx.doi.org/10.1137/1.9781611973099.132>

Competitive analysis was often criticized because of its too pessimistic guarantees which do not reflect the behavior of paging algorithms in practice. For instance, many deterministic paging algorithms achieve the optimal competitive ratio of k , yet LRU and its variants clearly outperform the rest in practice. In this paper we aim to reuse and refine insights from the competitive analysis to obtain new algorithms that cause few cache misses in practice. We propose a new measure of the “evilness” of the adversary, which results in a parametrization of the input that we denote *attack rate*. This measure is based on the characterization of the optimal offline algorithm by Koutsoupias and Papadimitriou and uses the fact that a number of pages are for sure in its memory. We show that the attack rate r is a tight bound on the competitive ratio of deterministic paging algorithms and give experimental results which show that r is usually much smaller than the cache size k and thus provides more realistic upper bounds for the competitive ratio of existing algorithms. We use a priority-based framework, which always yields r -competitive algorithms regardless of the priority assignment. In this framework, LRU can be obtained under a certain priority assignment and is thus only one algorithm among many other r -competitive ones. Using the enhanced flexibility given by this framework, we give a priority policy which leads to an algorithm outperforming LRU, RLRU and other practical algorithms on a wide selection of real-world cache traces.

3.23 Succinct Data Structures for Representing Equivalence Classes

J. Ian Munro (University of Waterloo, CA)

License © Creative Commons BY 3.0 Unported license

© J. Ian Munro


Joint work of El-Zein, Hicham; Lewenstein, Mosche; Munro, J. Ian; Raman, Venkatesh

We examine the problem of representing (naming) n elements in an arbitrary set of equivalences so that given a pair of elements we can (quickly) determine whether they are in the same class. We also want the extra space required for the representation of these classes to be minimal. If no extra space (other than the value n) is permitted, then an address space of $n \ln n + O(n)$ is necessary and sufficient. If the name space is required to be $[n]$, then $\Theta(\sqrt{n})$ extra bits are necessary and sufficient for a representation. We give an $O(\lg n)$ time, $O(\sqrt{n})$ bit data structure and an $O(1)$, $O(\sqrt{n} \lg n)$ bit method. An efficient method of supporting

updates on the classes is also discussed. Finally we consider the situation in which the name space is relaxed a little to $O(n)$. For the name space $[2n]$ we give a constant time method requiring $O((\log n)^2)$ extra bits and one with name space $[4n]$ using $O(\lg n \lg \lg n)$ bits.

3.24 On the Energy Consumption of Sorting and Searching


Markus E. Nebel (TU Kaiserslautern, DE)

License  Creative Commons BY 3.0 Unported license
© Markus E. Nebel

In this talk, based on the specification of the ARM CPU, we introduce a model for the energy consumption of programs being executed. Considering examples from sorting and searching, we show that, according to our model, a faster algorithm not always consumes less energy than a slower one. This observation most often holds for the average-case behavior on large inputs but sometimes is valid for the worst-case as well. This even remains true if so-called leakage power is taken into consideration. Many of our findings have been confirmed experimentally. Our study raises questions with respect to the design of energy efficient algorithms.

3.25 Lattice Compression

Patrick K. Nicholson (MPI für Informatik – Saarbrücken, DE)

License  Creative Commons BY 3.0 Unported license
© Patrick K. Nicholson

In this talk I survey some recent succinct data structures for representing partial orders. I pose the problem of representing lattice orders, i.e., their Hasse diagrams, using space matching the information theoretic lower bound to within constant factors. The current best data structures occupy $O(n^{3/2} \log n)$ bits of space, whereas the information theoretic bound is $\Theta(n^{3/2})$ bits; the constant hidden by the theta is unknown. The core challenge is to represent a $K_{2,2}$ -free bipartite graph using $\Theta(n^{3/2})$ bits of space, such that it can be determined, given two vertices, whether there is an edge between them.

3.26 Expected Linear Time Sorting for Word Size $\Omega(\log^2 n \log \log n)$

Jesper A. Sindahl Nielsen (Aarhus University, DK)

License  Creative Commons BY 3.0 Unported license
© Jesper A. Sindahl Nielsen

Sorting n integers in the word-RAM model is a fundamental problem and a long-standing open problem is whether integer sorting is possible in linear time when the word size is $\omega(\log n)$. In this paper we give an algorithm for sorting integers in expected linear time when the word size is $\Omega(\log^2 n \log \log n)$. Previously expected linear time sorting was only possible for word size $\Omega(\log^{2+\epsilon} n)$. Part of our construction is a new packed sorting algorithm that sorts n integers of $\frac{w}{b}$ -bits packed in $O(\frac{n}{b})$ words, where b is the number of integers packed in a word of size w bits. The packed sorting algorithm runs in expected $O(\frac{n}{b}(\log n + \log^2 b))$ time.

3.27 Gunrock: High-Performance, High-Level Iterative Graph Computation on GPUs

John Owens (University of California – Davis, US)

License © Creative Commons BY 3.0 Unported license
© John Owens

We are building Gunrock, a GPU library for bulk-synchronous programmable graph computation. Gunrock’s goals are a high-level programming model with high performance. The implementation today includes fast load-balanced traversal and programmable operators and enactors to implement a (currently) small number of graph primitives: breadth-first search, connected component, betweenness centrality, single-source shortest path, and PageRank. We note interesting future problems in the directions of traversal (push vs. pull), idempotent vs. non-idempotent operators, an operator formulation, bipartite and mutable graphs, and scalability.

3.28 One Answer, Two Questions, and Some Very Old Slides

Rasmus Pagh (IT University of Copenhagen, DK)

License © Creative Commons BY 3.0 Unported license
© Rasmus Pagh
Joint work of Pagh, Rasmus; Stöckel, Morten

The talk has three parts: (1) A new algorithm for sparse matrix multiplication in the I/O model. (Joint work with Morten Stöckel.) (2) Two intriguing questions on near-neighbor search that are part of a new ERC-funded project. (3) Revisiting a presentation that I gave at Dagstuhl 10 years ago in view of current hardware.

3.29 Encoding Top- k and Range Selection

Rajeev Raman (University of Leicester, GB)

License © Creative Commons BY 3.0 Unported license
© Rajeev Raman
Joint work of Davoodi, Pooya; Grossi, Roberto; Iacono, John; Navarro, Gonzalo; Rao, Srinivasa
Main reference R. Grossi, J. Iacono, G. Navarro, R. Raman, S. S. Rao, “Encodings for Range Selection and Top- k Queries,” in Proc. of the 21st Annual European Symp. on Algorithms (ESA’13), LNCS, Vol. 8125, pp. 553–564, Springer, 2013.
URL http://dx.doi.org/10.1007/978-3-642-40450-4_47

Many applications that deal with big data make use of the low effective entropy of data structuring problems: it is possible to pre-process the input into a data structure that can answer queries without accessing the input, whose space usage is significantly smaller than the space usage of the input. A classical example is as follows: given an array A of n values, pre-process it to answer Range Maximum Queries (RMQ), which given two indices l and r , returns the position containing the maximum element in $A[l..r]$. It is known that there is a data structure of size $2n + o(n)$ bits that answers RMQ in $O(1)$ time, in contrast to the $O(n \log n)$ bits needed to store A itself. We consider generalizations of the RMQ problem with low effective entropy defined on an array A of values including: Range Top-2 (returning the positions of the largest and second-largest elements in a sub-range of A),

Range Top- k (returning the positions of the top k values in a sub-range of A) and Range Selection (returning the position of the k -th largest value in a sub-range of A , given the sub-range and the value k as parameters). The work on range top- k appeared in ESA. The work on range second max is to appear in the Philosophical Transactions of the Royal Society A journal and can be found on arXiv <http://arxiv.org/abs/1311.4394>.

3.30 Algorithms in the Ultra-Wide Word Model

Alejandro Salinger (Universität des Saarlandes, DE)

License © Creative Commons BY 3.0 Unported license

© Alejandro Salinger

Joint work of Salinger, Alejandro; Farzan, Arash; López-Ortiz, Alejandro; Nicholson, Patrick K.

Main reference A. Farzan, A. López-Ortiz, P. K. Nicholson, A. Salinger, “Algorithms in the Ultra-Wide Word Model,” University of Waterloo Technical Report CS-2012-21.

URL <https://cs.uwaterloo.ca/research/tr/2012/CS-2012-21.pdf>

The effective use of parallel computing resources to speed up algorithms in current multi-core and other parallel architectures remains a difficult challenge, with ease of programming playing a key role in the eventual success of these architectures. In this talk we consider an alternative view of parallelism in the form of an ultra-wide word processor. We introduce the Ultra-Wide Word architecture and model, an extension of the word-RAM model that allows for constant time operations on thousands of bits in parallel. Word parallelism as exploited by the word-RAM model does not suffer from the more difficult aspects of parallel programming, namely synchronization and concurrency. In practice, the speedups obtained by word-RAM algorithms are moderate, as they are limited by the word size. We argue that a large class of word-RAM algorithms can be implemented in the Ultra-Wide Word model, obtaining speedups comparable to multi-threaded computations while keeping the simplicity of programming of the sequential RAM model. We show that this is the case by describing implementations of Ultra-Wide Word algorithms for dynamic programming and string searching. In addition, we show that the Ultra-Wide Word model can be used to implement a non-standard memory architecture, which enables the sidestepping of lower bounds of important data structure problems such as priority queues and dynamic prefix sums.

3.31 Communication Efficient Algorithms

Peter Sanders (KIT – Karlsruhe Institute of Technology, DE)

License © Creative Commons BY 3.0 Unported license

© Peter Sanders

Joint work of Sanders, Peter; Schlag, Sebastian; Müller, Ingo

Main reference P. Sanders, S. Schlag, I. Müller, “Communication efficient algorithms for fundamental big data problems,” in Proc. of the 2013 IEEE Int’l Conf. on Big Data, pp. 15–23, IEEE, 2013.


URL <http://dx.doi.org/10.1109/BigData.2013.6691549>

Big Data applications often store or obtain their data distributed over many computers connected by a network. Since the network is usually slower than the local memory of the machines, it is crucial to process the data in such a way that not too much communication takes place. Indeed, only communication volume sublinear in the input size may be affordable. We believe that this direction of research deserves more intensive study. We give examples

for several fundamental algorithmic problems where nontrivial algorithms with sublinear communication volume are possible. Our main technical contributions are several related results on distributed Bloom filter replacements, duplicate detection, and data base join.

3.32 New Worst-Case Approaches in Robust Optimization

Anita Schöbel (*Universität Göttingen, DE*)

License  Creative Commons BY 3.0 Unported license
© Anita Schöbel

Analysis of algorithms and data structures is often a worst-case analysis. In mathematical optimization, worst cases are considered in the field of robust optimization. The two best known robustness concepts are strict robustness (see, e.g., Ben-Tal, Ghaoui and Nemirovski, 2009) and regret robustness (see, e.g., Kouvelis and Yu, 1997). Although well researched, they are not suitable for all types of problems which is due to their over-conservatism. Consequently, the robust optimization community is developing new robustness concepts, also dealing with worst cases, but in a less conservative way. Among these are adjustable robustness (Ben-Tal, Goryashko, Guslitzer, Nemirovski, 2003), Gamma-robustness (Bertsimas and Sim, 2004), light robustness (Fischetti and Monaci, 2009), and different versions of recovery robustness (Liebchen, Lübbecke, Möhring and, Stiller, 2009) (Goerigk and Schöbel, 2011). In my talk I review these concepts and discuss their applicability for the design and analysis of data structures and algorithms.

3.33 MapReduce Transducers: A Formal Model for Distributed Streaming Computations

Nicole Schweikardt (*Goethe-Universität Frankfurt am Main, DE*)

License  Creative Commons BY 3.0 Unported license
© Nicole Schweikardt

Joint work of Neven, Frank; Servais, Frederic; Schweikardt, Nicole; Tan, Tony

We study the expressiveness of MapReduce where reducers are restricted to the generic formalism of finite memory computations. We utilise the framework of register automata and define MapReduce Automata (MRA) and MapReduce Transducers (MRT). The only difference between MRT and MRA is on the reducer level: within an MRA, a reducer is a register automaton which makes one pass over its input, and which outputs a single bag of values depending on its final configuration. In contrast, within an MRT, a reducer makes two passes over the input: in the first pass the reducer acts like a register automaton while in the second pass, it acts like a transducer that for each input value, outputs a bag of values. We can prove the following: (1) MRTs are strictly stronger than MRAs; (2) MRTs cannot detect the presence of a triangle in a graph; (3) for bounded degree graphs, MRAs can check all first-order properties with modulo counting quantifiers; (4) MRTs can evaluate all semijoin algebra queries, while MRAs cannot; (5) within MRAs and MRTs there is a strict hierarchy w.r.t. the number of rounds.

3.34 Counting Arbitrary Subgraphs in Data Streams

He Sun (MPI für Informatik – Saarbrücken, DE)

License © Creative Commons BY 3.0 Unported license
© He Sun

Joint work of Kane, Daniel; Mehlhorn, Kurt; Sauerwald, Thomas; Sun, He

Main reference D. M. Kane, K. Mehlhorn, T. Sauerwald, H. Sun, “Counting Arbitrary Subgraphs in Data Streams,” in Proc. of the 39th Int’l Colloquium on Automata, Languages, and Programming (ICALP’12), LNCS, Vol. 7392, pp. 598–609, Springer, 2012.

URL http://dx.doi.org/10.1007/978-3-642-31585-5_53

We study the subgraph counting problem in data streams. We provide the first non-trivial estimator for approximately counting the number of occurrences of an arbitrary subgraph H of constant size in a (large) graph. Our estimator works in the turnstile model, i.e., can handle both edge-insertions and edge-deletions, and is applicable in a distributed setting. Prior to this work, only for a few non-regular graphs estimators were known in case of edge-insertions, leaving the problem of counting general subgraphs in the turnstile model wide open. We further demonstrate the applicability of our estimator by analyzing its concentration for several graphs H and the case where the underlying graph is a power law graph.

3.35 Strings, Geometry, Special Cases...

Sharma V. Thankachan (Louisiana State University, US)

License © Creative Commons BY 3.0 Unported license
© Sharma V. Thankachan

Main reference R. Shah, C. Sheng, S. V. Thankachan, J. S. Vitter, “Top- k Document Retrieval in External Memory,” in Proc. of the 21st Annual European Symp. on Algorithms (ESA’13), LNCS, Vol. 8125, pp. 803–814, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-40450-4_68

Often times, many string indexing problems can be reduced to special geometric problems, for which efficient solutions may exist. We present several such examples. One of them is the shared constraint range reporting (SCRr) problem, which is a special $(2, 1, 1)$ -range reporting query where a corner of the query rectangle touches the line $x = y$. Notice that any I/O-optimal data structure for processing a general $(2, 1, 1)$ query must take $\Omega(n \log n / \log \log n)$ space. However, our special query can be optimally handled using an $O(n \log \log n)$ space structure. The problem is motivated from string indexing problems for top- k queries.

3.36 An Edge Quadtree for External Memory

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

License © Creative Commons BY 3.0 Unported license
© Laura I. Toma

Main reference H. Haverkort, M. McGranaghan, L. Toma, “An Edge Quadtree for External Memory,” in Proc. of the 12th Int’l Symp. on Experimental Algorithms (SEA’13), LNCS, Vol. 7933, pp. 115–126, Springer, 2013.

URL http://dx.doi.org/10.1007/978-3-642-38527-8_12

We consider the problem of building a quadtree subdivision for a set E of n non-intersecting edges in the plane. Our approach is to first build a quadtree on the vertices corresponding to the endpoints of the edges, and then compute the intersections between E and the cells in the subdivision. For any $k \geq 1$, we call a K -quadtree a linear compressed quadtree that

has $O(n/k)$ cells with $O(k)$ vertices each, where each cell stores the edges intersecting the cell. We show how to build a K-quadtrees in $O(\text{sort}(n+l))$ I/Os, where $l = O(n^2/k)$ is the number of such intersections. The value of k can be chosen to trade off between the number of cells and the size of a cell in the quadtree. We give an empirical evaluation in external memory on triangulated terrains and USA TIGER data. As an application, we consider the problem of map overlay, or finding the pairwise intersections between two sets of edges. Our findings confirm that the K-quadtrees is viable for these types of data and its construction is scalable to hundreds of millions of edges.

3.37 Of Motifs and Goals: Mining Trajectory Data

Jan Vahrenhold (*Universität Münster, DE*)

License © Creative Commons BY 3.0 Unported license
© Jan Vahrenhold

Joint work of Gudmundsson, Joachim; Thom, Andreas; Vahrenhold, Jan

Main reference J. Gudmundsson, A. Thom, J. Vahrenhold, “Of Motifs and Goals: Mining Trajectory Data,” in Proc. of the 20th Int’l Conf. on Advances in Geographic Information Systems (SIGSPATIAL’12), pp. 129–138, ACM, 2012.

URL <http://dx.doi.org/10.1145/2424321.2424339>

In response to the increasing volume of trajectory data obtained, e.g., from tracking athletes, animals, or meteorological phenomena, we present a new space-efficient algorithm for the analysis of trajectory data. The algorithm combines techniques from computational geometry, data mining, and string processing and offers a modular design that allows for a user-guided exploration of trajectory data incorporating domain-specific constraints and objectives.

3.38 Dual-Pivot Quicksort – Asymmetries in Sorting

Sebastian Wild (*TU Kaiserslautern, DE*)

License © Creative Commons BY 3.0 Unported license
© Sebastian Wild

Joint work of Wild, Sebastian; Nebel, Markus

Main reference S. Wild and M. E. Nebel, “Average Case Analysis of Java 7’s Dual Pivot Quicksort,” in Proc. of the 20th Annual European Symp. on Algorithms (ESA’12), LNCS, Vol. 7501, pp. 825–836, Springer, 2012.

URL http://dx.doi.org/10.1007/978-3-642-33090-2_71

We review the history of Yaroslavskiy’s dual-pivot Quicksort that is used in Oracle’s Java library since Java 7 and show how asymmetries in the algorithm lead to fewer needed comparisons than in classic Quicksort. These asymmetries can be amplified by choosing pivots with a systematic skew.

References

- 1 Sebastian Wild, Markus E. Nebel. *Average Case Analysis of Java 7’s Dual Pivot Quicksort*. ESA 2012, LNCS 7501 pp. 825–836, Springer, 2012
- 2 Sebastian Wild. *Java 7’s Dual Pivot Quicksort*. Master Thesis, 2012
<http://nbn-resolving.de/urn/resolver.pl?urn:nbn:de:hbz:386-kluedo-34638>
- 3 Sebastian Wild, Markus E. Nebel, Ralph Neininger. *Average Case and Distributional Analysis of Java 7’s Dual Pivot Quicksort*. accepted for publication in ACM Transactions on Algorithms

3.39 The Link Assessment Problem of Low Intensity Relationships in Complex Networks

Katharina A. Zweig (TU Kaiserslautern, DE)

License  Creative Commons BY 3.0 Unported license
© Katharina A. Zweig

Many large network data sets are noisy and contain links representing low intensity relationships that are difficult to differentiate from false-positive, random interactions. This is especially true for high-throughput data from systems biology, large-scale ecological data, but also for Web 2.0 data such as product ratings by users. In these networks with false-positive and false-negative edges, it is possible to clean the data based on the principle of structural similarity which assesses the number of common neighbors between any two nodes. Using similarity indices to globally rank all possible node pairs links and choosing the top-ranked pairs, true-positive edges can be validated, missing edges inferred, and spurious observations removed. While many similarity indices have been proposed to this end, there is no conclusive decision on which one to use. In this article we first contribute benchmark sets for complex networks from three very different settings (e-commerce, systems biology, and social networks) to enable a quantitative performance analysis of classic node similarity measures. Second, we propose a new node similarity measure called z^* which is the only stable top performing similarity measure in all cases. A third contribution is the central insight that, instead of using a global ranking, the performance can be tremendously increased by choosing the highest ranked neighbors for each single node (local ranking). There are many open problems connected to this problem, starting from parallelizing the algorithms, expected number of common neighbors in random graphs, to space/time tradeoffs depending on the size of the input.

Participants

- Peyman Afshani
Aarhus University, DK
- Deepak Ajwani
Bell Labs – Dublin, IE
- Lars Arge
Aarhus University, DK
- Martin Aumüller
TU Ilmenau, DE
- Hannah Bast
Universität Freiburg, DE
- Timo Bingmann
KIT – Karlsruhe Institute of Technology, DE
- Gerth Stølting Brodal
Aarhus University, DK
- Andrej Brodnik
University of Primorska, SI
- Martin Dietzfelbinger
TU Ilmenau, DE
- Anne Driemel
TU Dortmund, DE
- Rolf Fagerberg
University of Southern Denmark – Odense, DK
- Johannes Fischer
TU Dortmund, DE
- Mordecai Golin
HKUST – Kowloon, HK
- Torben Hagerup
Universität Augsburg, DE
- Tanja Hartmann
KIT – Karlsruhe Institute of Technology, DE
- Herman J. Haverkort
TU Eindhoven, NL
- Meng He
Dalhousie University, CA
- John Iacono
Polytechnic Institute of NYU – Brooklyn, US
- Riko Jacob
ETH Zürich, CH
- Tsvi Kopelowitz
University of Michigan – Ann Arbor, US
- Moshe Lewenstein
Bar-Ilan University, IL
- Alejandro Lopez-Ortiz
University of Waterloo, CA
- Kurt Mehlhorn
MPI für Informatik – Saarbrücken, DE
- Ulrich Carsten Meyer
Goethe-Universität Frankfurt am Main, DE
- Gabriel Moruz
Universität Frankfurt, DE
- Ian Munro
University of Waterloo, CA
- Markus E. Nebel
TU Kaiserslautern, DE
- Patrick K. Nicholson
MPI für Informatik – Saarbrücken, DE
- Jesper A. Sindahl Nielsen
Aarhus University, DK
- John Owens
Univ. of California – Davis, US
- Rasmus Pagh
IT Univ. of Copenhagen, DK
- Rajeev Raman
University of Leicester, GB
- Alejandro Salinger
Universität des Saarlandes, DE
- Peter Sanders
KIT – Karlsruhe Institute of Technology, DE
- Anita Schöbel
Universität Göttingen, DE
- Nicole Schweikardt
Goethe-Universität Frankfurt am Main, DE
- Robert Sedgewick
Princeton University, US
- He Sun
MPI für Informatik – Saarbrücken, DE
- Sharma V. Thankachan
Louisiana State University, US
- Laura I. Toma
Bowdoin College – Brunswick, US
- Jan Vahrenhold
Universität Münster, DE
- Sebastian Wild
TU Kaiserslautern, DE
- Katharina A. Zweig
TU Kaiserslautern, DE

