Algorithms and Data Structures for External Memory

Algorithms and Data Structures for External Memory

Jeffrey Scott Vitter

Department of Computer Science
Purdue University
West Lafayette
Indiana, 47907–2107
USA
jsv@purdue.edu

the essence of knowledge

Boston – Delft

Foundations and Trends® in Theoretical Computer Science

Published, sold and distributed by:
now Publishers Inc.
PO Box 1024
Hanover, MA 02339
USA
Tel. +1-781-985-4510
www.nowpublishers.com
sales@nowpublishers.com

Outside North America:
now Publishers Inc.
PO Box 179
2600 AD Delft
The Netherlands
Tel. +31-6-51115274

The preferred citation for this publication is J. S. Vitter, Algorithms and Data Structures for External Memory, Foundation and Trends® in Theoretical Computer Science, vol 2, no 4, pp 305–474, 2006

ISBN: 978-1-60198-106-6

Foundations and Trends® in Theoretical Computer Science

Volume 2 Issue 4, 2006

Editorial Scope

Foundations and Trends® in Theoretical Computer Science will publish survey and tutorial articles in the following topics:

- Algorithmic game theory
- Computational algebra
- Computational aspects of combinatorics and graph theory
- Computational aspects of communication
- Computational biology
- Computational complexity
- Computational geometry
- Computational learning
- Computational Models and Complexity
- Computational Number Theory
- Cryptography and information security
- Data structures
- Database theory
- Design and analysis of algorithms
- Distributed computing
- Information retrieval
- Operations Research
- Parallel algorithms
- Quantum Computation
- Randomness in Computation

Information for Librarians

Foundations and Trends® in Theoretical Computer Science, 2006, Volume 2, 4 issues. ISSN paper version 1551-305X. ISSN online version 1551-3068. Also available as a combined paper and online subscription.

Foundations and Trends® in
Theoretical Computer Science
Vol. 2, No. 4 (2006) 305–474

DOI: 10.1561/0400000014

Algorithms and Data Structures for External Memory

Jeffrey Scott Vitter

Department of Computer Science, Purdue University, West Lafayette, Indiana, 47907–2107, USA, jsv@purdue.edu

Abstract

Data sets in large applications are often too massive to fit completely inside the computer's internal memory. The resulting input/output communication (or I/O) between fast internal memory and slower external memory (such as disks) can be a major performance bottleneck. In this manuscript, we survey the state of the art in the design and analysis of algorithms and data structures for *external memory* (or *EM* for short), where the goal is to exploit locality and parallelism in order to reduce the I/O costs. We consider a variety of EM paradigms for solving batched and online problems efficiently in external memory.

For the batched problem of sorting and related problems like permuting and fast Fourier transform, the key paradigms include distribution and merging. The paradigm of disk striping offers an elegant way to use multiple disks in parallel. For sorting, however, disk striping can be nonoptimal with respect to I/O, so to gain further improvements we discuss distribution and merging techniques for using the disks independently. We also consider useful techniques for batched EM problems involving matrices, geometric data, and graphs.

In the online domain, canonical EM applications include dictionary lookup and range searching. The two important classes of indexed data structures are based upon extendible hashing and B-trees. The paradigms of filtering and bootstrapping provide convenient means in online data structures to make effective use of the data accessed from disk. We also re-examine some of the above EM problems in slightly different settings, such as when the data items are moving, when the data items are variable-length such as character strings, when the data structure is compressed to save space, or when the allocated amount of internal memory can change dynamically.

Programming tools and environments are available for simplifying the EM programming task. We report on some experiments in the domain of spatial databases using the TPIE system (Transparent Parallel I/O programming Environment). The newly developed EM algorithms and data structures that incorporate the paradigms we discuss are significantly faster than other methods used in practice.

Preface

I first became fascinated about the tradeoffs between computing and memory usage while a graduate student at Stanford University. Over the following years, this theme has influenced much of what I have done professionally, not only in the field of external memory algorithms, which this manuscript is about, but also on other topics such as data compression, data mining, databases, prefetching/caching, and random sampling.

The reality of the computer world is that no matter how fast computers are and no matter how much data storage they provide, there will always be a desire and need to push the envelope. The solution is not to wait for the next generation of computers, but rather to examine the fundamental constraints in order to understand the limits of what is possible and to translate that understanding into effective solutions.

In this manuscript you will consider a scenario that arises often in large computing applications, namely, that the relevant data sets are simply too massive to fit completely inside the computer's internal memory and must instead reside on disk. The resulting input/output communication (or I/O) between fast internal memory and slower external memory (such as disks) can be a major performance bottleneck. This manuscript provides a detailed overview of the design and analysis of algorithms and data structures for *external memory* (or simply *EM*), where the goal is to exploit locality and parallelism in order to reduce the I/O costs. Along the way, you will learn a variety of EM paradigms for solving batched and online problems efficiently.

For the batched problem of sorting and related problems like permuting and fast Fourier transform, the two fundamental paradigms are distribution and merging. The paradigm of disk striping offers an elegant way to use multiple disks in parallel. For sorting, however, disk striping can be nonoptimal with respect to I/O, so to gain further improvements we discuss distribution and merging techniques for using the disks independently, including an elegant duality property that yields state-of-the-art algorithms. You will encounter other useful techniques for batched EM problems involving matrices (such as matrix multiplication and transposition), geometric data (such as finding intersections and constructing convex hulls) and graphs (such as list ranking, connected components, topological sorting, and shortest paths).

In the online domain, which involves constructing data structures to answer queries, we discuss two canonical EM search applications: dictionary lookup and range searching. Two important paradigms for developing indexed data structures for these problems are hashing (including extendible hashing) and tree-based search (including B-trees). The paradigms of filtering and bootstrapping provide convenient means in online data structures to make effective use of the data accessed from disk. You will also be exposed to some of the above EM problems in slightly different settings, such as when the data items are moving, when the data items are variable-length (e.g., strings of text), when the data structure is compressed to save space, and when the allocated amount of internal memory can change dynamically.

Programming tools and environments are available for simplifying the EM programming task. You will see some experimental results in the domain of spatial databases using the TPIE system, which stands for Transparent Parallel I/O programming Environment. The newly developed EM algorithms and data structures that incorporate the paradigms discussed in this manuscript are significantly faster than other methods used in practice.

I would like to thank my colleagues for several helpful comments, especially Pankaj Agarwal, Lars Arge, Ricardo Baeza-Yates, Adam Buchsbaum, Jeffrey Chase, Michael Goodrich, Wing-Kai Hon, David Hutchinson, Gonzalo Navarro, Vasilis Samoladas, Peter Sanders, Rahul Shah, Amin Vahdat, and Norbert Zeh. I also thank the referees and editors for their help and suggestions, as well as the many wonderful staff members I've had the privilege to work with. Figure 1.1 is a modified version of a figure by Darren Vengroff, and Figures 2.1 and 5.2 come from [118, 342]. Figures 5.4–5.8, 8.2–8.3, 10.1, 12.1, 12.2, 12.4, and 14.1 are modified versions of figures in [202, 47, 147, 210, 41, 50, 158], respectively.

This manuscript is an expanded and updated version of the article in *ACM Computing Surveys*, Vol. 33, No. 2, June 2001. I am very appreciative for the support provided by the National Science Foundation through research grants CCR–9522047, EIA–9870734, CCR–9877133, IIS–0415097, and CCF–0621457; by the Army Research Office through MURI grant DAAH04–96–1–0013; and by IBM Corporation. Part of this manuscript was done at Duke University, Durham, North Carolina; the University of Aarhus, Århus, Denmark; INRIA, Sophia Antipolis, France; and Purdue University, West Lafayette, Indiana.

I especially want to thank my wife Sharon and our three kids (or more accurately, young adults) Jillian, Scott, and Audrey for their ever-present love and support. I most gratefully dedicate this manuscript to them.

West Lafayette, Indiana — J. S. V.
March 2008

Contents

1
Introduction

The world is drowning in data! In recent years, we have been deluged by a torrent of data from a variety of increasingly data-intensive applications, including databases, scientific computations, graphics, entertainment, multimedia, sensors, web applications, and email. NASA's Earth Observing System project, the core part of the Earth Science Enterprise (formerly Mission to Planet Earth), produces petabytes (10^{15} bytes) of raster data per year [148]. A petabyte corresponds roughly to the amount of information in one billion graphically formatted books. The online databases of satellite images used by Microsoft TerraServer (part of MSN Virtual Earth) [325] and Google Earth [180] are multiple terabytes (10^{12} bytes) in size. Wal-Mart's sales data warehouse contains over a half petabyte (500 terabytes) of data. A major challenge is to develop mechanisms for processing the data, or else much of the data will be useless.

For reasons of economy, general-purpose computer systems usually contain a hierarchy of memory levels, each level with its own cost and performance characteristics. At the lowest level, CPU registers and caches are built with the fastest but most expensive memory. For internal main memory, dynamic random access memory (DRAM) is

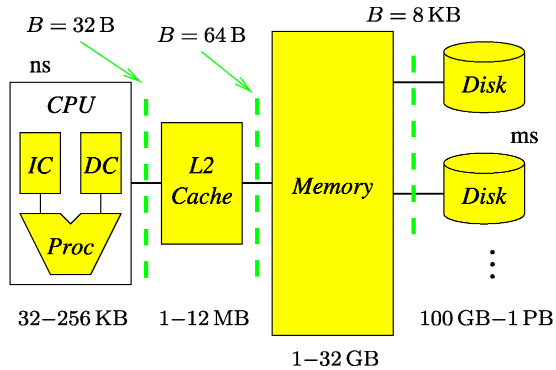

Fig. 1.1 The memory hierarchy of a typical uniprocessor system, including registers, instruction cache, data cache (level 1 cache), level 2 cache, internal memory, and disks. Some systems have in addition a level 3 cache, not shown here. Memory access latency ranges from less than one nanosecond (ns, 10^{-9} seconds) for registers and level 1 cache to several milliseconds (ms, 10^{-3} seconds) for disks. Typical memory sizes for each level of the hierarchy are shown at the bottom. Each value of B listed at the top of the figure denotes a typical block transfer size between two adjacent levels of the hierarchy. All sizes are given in units of bytes (B), kilobytes (KB, 10^3 B), megabytes (MB, 10^6 B), gigabytes (GB, 10^9 B), and petabytes (PB, 10^{15} B). (In the PDM model defined in Chapter 2, we measure the block size B in units of items rather than in units of bytes.) In this figure, 8 KB is the indicated physical block transfer size between internal memory and the disks. However, in batched applications we often use a substantially larger logical block transfer size.

typical. At a higher level, inexpensive but slower magnetic disks are used for external mass storage, and even slower but larger-capacity devices such as tapes and optical disks are used for archival storage. These devices can be attached via a network fabric (e.g., Fibre Channel or iSCSI) to provide substantial external storage capacity. Figure 1.1 depicts a typical memory hierarchy and its characteristics.

Most modern programming languages are based upon a programming model in which memory consists of one uniform address space. The notion of virtual memory allows the address space to be far larger than what can fit in the internal memory of the computer. Programmers have a natural tendency to assume that all memory references require the same access time. In many cases, such an assumption is reasonable (or at least does not do harm), especially when the data sets are not large. The utility and elegance of this programming model are to a large extent why it has flourished, contributing to the productivity of the software industry.

However, not all memory references are created equal. Large address spaces span multiple levels of the memory hierarchy, and accessing the data in the lowest levels of memory is orders of magnitude faster than accessing the data at the higher levels. For example, loading a register can take a fraction of a nanosecond (10^{-9} seconds), and accessing internal memory takes several nanoseconds, but the latency of accessing data on a disk is multiple milliseconds (10^{-3} seconds), which is about one million times slower! In applications that process massive amounts of data, the *Input/Output* communication (or simply *I/O*) between levels of memory is often the bottleneck.

Many computer programs exhibit some degree of *locality* in their pattern of memory references: Certain data are referenced repeatedly for a while, and then the program shifts attention to other sets of data. Modern operating systems take advantage of such access patterns by tracking the program's so-called "working set" — a vague notion that roughly corresponds to the recently referenced data items [139]. If the working set is small, it can be cached in high-speed memory so that access to it is fast. Caching and prefetching heuristics have been developed to reduce the number of occurrences of a "fault," in which the referenced data item is not in the cache and must be retrieved by an I/O from a higher level of memory. For example, in a page fault, an I/O is needed to retrieve a disk page from disk and bring it into internal memory.

Caching and prefetching methods are typically designed to be general-purpose, and thus they cannot be expected to take full advantage of the locality present in every computation. Some computations themselves are inherently nonlocal, and even with omniscient cache management decisions they are doomed to perform large amounts of I/O and suffer poor performance. Substantial gains in performance may be possible by incorporating locality *directly* into the algorithm design and by explicit management of the contents of each level of the memory hierarchy, thereby bypassing the virtual memory system.

We refer to algorithms and data structures that explicitly manage data placement and movement as *external memory* (or *EM*) *algorithms and data structures*. Some authors use the terms *I/O algorithms* or *out-of-core algorithms*. We concentrate in this manuscript on the I/O communication between the random access internal memory and the magnetic disk external memory, where the relative difference in access speeds is most apparent. We therefore use the term I/O to designate the communication between the internal memory and the disks.

1.1 Overview

In this manuscript, we survey several paradigms for exploiting locality and thereby reducing I/O costs when solving problems in external memory. The problems we consider fall into two general categories:

(1) *Batched problems*, in which no preprocessing is done and the entire file of data items must be processed, often by streaming the data through the internal memory in one or more passes.
(2) *Online problems*, in which computation is done in response to a continuous series of query operations. A common technique for online problems is to organize the data items via a hierarchical index, so that only a very small portion of the data needs to be examined in response to each query. The data being queried can be either *static*, which can be preprocessed for efficient query processing, or *dynamic*, where the queries are intermixed with updates such as insertions and deletions.

We base our approach upon the *parallel disk model* (PDM) described in the next chapter. PDM provides an elegant and reasonably accurate model for analyzing the relative performance of EM algorithms and data structures. The three main performance measures of PDM are *the number of (parallel) I/O operations, the disk space usage,* and *the (parallel) CPU time*. For reasons of brevity, we focus on the first two measures. Most of the algorithms we consider are also efficient in terms of CPU time. In Chapter 3, we list four fundamental I/O bounds that pertain to most of the problems considered in this manuscript. In Chapter 4, we show why it is crucial for EM algorithms to exploit locality, and we discuss an automatic load balancing technique called disk striping for using multiple disks in parallel.

Our general goal is to design optimal algorithms and data structures, by which we mean that their performance measures are within a constant factor of the optimum or best possible.[1] In Chapter 5, we look at the canonical batched EM problem of external sorting and the related problems of permuting and fast Fourier transform. The two important paradigms of distribution and merging — as well as the notion of duality that relates the two — account for all well-known external sorting algorithms. Sorting with a single disk is now well understood, so we concentrate on the more challenging task of using multiple (or parallel) disks, for which disk striping is not optimal. The challenge is to guarantee that the data in each I/O are spread evenly across the disks so that the disks can be used simultaneously. In Chapter 6, we cover the fundamental lower bounds on the number of I/Os needed to perform sorting and related batched problems. In Chapter 7, we discuss grid and linear algebra batched computations.

For most problems, parallel disks can be utilized effectively by means of disk striping or the parallel disk techniques of Chapter 5, and hence we restrict ourselves starting in Chapter 8 to the conceptually simpler single-disk case. In Chapter 8, we mention several effective paradigms for batched EM problems in computational geometry. The paradigms include distribution sweep (for spatial join and finding all nearest neighbors), persistent B-trees (for batched point location and visibility), batched filtering (for 3-D convex hulls and batched point location), external fractional cascading (for red-blue line segment intersection), external marriage-before-conquest (for output-sensitive convex hulls), and randomized incremental construction with gradations (for line segment intersections and other geometric problems). In Chapter 9, we look at EM algorithms for combinatorial problems on graphs, such as list ranking, connected components, topological sorting, and finding shortest paths. One technique for constructing I/O-efficient EM algorithms is to simulate parallel algorithms; sorting is used between parallel steps in order to reblock the data for the simulation of the next parallel step.

[1] In this manuscript we generally use the term "optimum" to denote the absolute best possible and the term "optimal" to mean within a constant factor of the optimum.

In Chapters 10–12, we consider data structures in the online setting. The dynamic dictionary operations of insert, delete, and lookup can be implemented by the well-known method of hashing. In Chapter 10, we examine hashing in external memory, in which extra care must be taken to pack data into blocks and to allow the number of items to vary dynamically. Lookups can be done generally with only one or two I/Os. Chapter 11 begins with a discussion of B-trees, the most widely used online EM data structure for dictionary operations and one-dimensional range queries. Weight-balanced B-trees provide a uniform mechanism for dynamically rebuilding substructures and are useful for a variety of online data structures. Level-balanced B-trees permit maintenance of parent pointers and support cut and concatenate operations, which are used in reachability queries on monotone subdivisions. The buffer tree is a so-called "batched dynamic" version of the B-tree for efficient implementation of search trees and priority queues in EM sweep line applications. In Chapter 12, we discuss spatial data structures for multidimensional data, especially those that support online range search. Multidimensional extensions of the B-tree, such as the popular R-tree and its variants, use a linear amount of disk space and often perform well in practice, although their worst-case performance is poor. A nonlinear amount of disk space is required to perform 2-D orthogonal range queries efficiently in the worst case, but several important special cases of range searching can be done efficiently using only linear space. A useful design paradigm for EM data structures is to "externalize" an efficient data structure designed for internal memory; a key component of how to make the structure I/O-efficient is to "bootstrap" a static EM data structure for small-sized problems into a fully dynamic data structure of arbitrary size. This paradigm provides optimal linear-space EM data structures for several variants of 2-D orthogonal range search.

In Chapter 13, we discuss some additional EM approaches useful for dynamic data structures, and we also investigate kinetic data structures, in which the data items are moving. In Chapter 14, we focus on EM data structures for manipulating and searching text strings. In many applications, especially those that operate on text strings, the data are highly compressible. Chapter 15 discusses ways to develop data structures that are themselves compressed, but still fast to query.

Table 1.1 Paradigms for I/O efficiency discussed in this manuscript.

Paradigm	Section
Batched dynamic processing	11.4
Batched filtering	8
Batched incremental construction	8
Bootstrapping	12
Buffer trees	11.4
B-trees	11, 12
Compression	15
Decomposable search	13.1
Disk striping	4.2
Distribution	5.1
Distribution sweeping	8
Duality	5.3
External hashing	10
Externalization	12.3
Fractional cascading	8
Filtering	12
Lazy updating	11.4
Load balancing	4
Locality	4.1
Marriage before conquest	8
Merging	5.2
Parallel block transfer	4.2
Parallel simulation	9
Persistence	11.1
Random sampling	5.1
R-trees	12.2
Scanning (or streaming)	2.2
Sparsification	9
Time-forward processing	11.4

In Chapter 16, we discuss EM algorithms that adapt optimally to dynamically changing internal memory allocations.

In Chapter 17, we discuss programming environments and tools that facilitate high-level development of efficient EM algorithms. We focus primarily on the TPIE system (Transparent Parallel I/O Environment), which we use in the various timing experiments in this manuscript. We conclude with some final remarks and observations in the Conclusions.

Table 1.1 lists several of the EM paradigms discussed in this manuscript.

References

[1] D. J. Abel, "A B^{+}-tree structure for large quadtrees," *Computer Vision, Graphics, and Image Processing*, vol. 27, pp. 19–31, July 1984.

[2] J. Abello, A. L. Buchsbaum, and J. Westbrook, "A functional approach to external graph algorithms," *Algorithmica*, vol. 32, no. 3, pp. 437–458, 2002.

[3] J. Abello, P. M. Pardalos, and M. G. Resende, eds., *Handbook of Massive Data Sets*. Norwell, Mass.: Kluwer Academic Publishers, 2002.

[4] A. Acharya, M. Uysal, and J. Saltz, "Active disks: Programming model, algorithms and evaluation," *ACM SIGPLAN Notices*, vol. 33, pp. 81–91, November 1998.

[5] M. Adler, "New coding techniques for improved bandwidth utilization," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, (Burlington, VT), pp. 173–182, October 1996.

[6] P. K. Agarwal, L. Arge, G. S. Brodal, and J. S. Vitter, "I/O-efficient dynamic point location in monotone planar subdivisions," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 11–20, ACM Press, 1999.

[7] P. K. Agarwal, L. Arge, and A. Danner, "From LIDAR to GRID DEM: A scalable approach," in *Proceedings of the International Symposium on Spatial Data Handling*, 2006.

[8] P. K. Agarwal, L. Arge, and J. Erickson, "Indexing moving points," *Journal of Computer and System Sciences*, vol. 66, no. 1, pp. 207–243, 2003.

[9] P. K. Agarwal, L. Arge, J. Erickson, P. G. Franciosa, and J. S. Vitter, "Efficient searching with linear constraints," *Journal of Computer and System Sciences*, vol. 61, pp. 194–216, October 2000.

[10] P. K. Agarwal, L. Arge, T. M. Murali, K. Varadarajan, and J. S. Vitter, "I/O-efficient algorithms for contour line extraction and planar graph blocking," in

Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, pp. 117–126, ACM Press, 1998.

[11] P. K. Agarwal, L. Arge, O. Procopiuc, and J. S. Vitter, "A framework for index bulk loading and dynamization," in *Proceedings of the International Colloquium on Automata, Languages, and Programming*, pp. 115–127, Crete, Greece: Springer-Verlag, 2001.

[12] P. K. Agarwal, L. Arge, and J. Vahrenhold, "Time responsive external data structures for moving points," in *Proceedings of the Workshop on Algorithms and Data Structures*, pp. 50–61, 2001.

[13] P. K. Agarwal, L. Arge, J. Yang, and K. Yi, "I/O-efficient structures for orthogonal range-max and stabbing-max queries," in *Proceedings of the European Symposium on Algorithms*, pp. 7–18, Springer-Verlag, 2003.

[14] P. K. Agarwal, L. Arge, and K. Yi, "I/O-efficient construction of constrained delaunay triangulations," in *Proceedings of the European Symposium on Algorithms*, pp. 355–366, Springer-Verlag, 2005.

[15] P. K. Agarwal, L. Arge, and K. Yi, "An optimal dynamic interval stabbing-max data structure?," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 803–812, ACM Press, 2005.

[16] P. K. Agarwal, L. Arge, and K. Yi, "I/O-efficient batched union-find and its applications to terrain analysis," in *Proceedings of the ACM Symposium on Computational Geometry*, ACM Press, 2006.

[17] P. K. Agarwal, M. de Berg, J. Gudmundsson, M. Hammar, and H. J. Haverkort, "Box-trees and R-trees with near-optimal query time," *Discrete and Computational Geometry*, vol. 28, no. 3, pp. 291–312, 2002.

[18] P. K. Agarwal and J. Erickson, "Geometric range searching and its relatives," in *Advances in Discrete and Computational Geometry*, (B. Chazelle, J. E. Goodman, and R. Pollack, eds.), pp. 1–56, Providence, Rhode Island: American Mathematical Society Press, 1999.

[19] P. K. Agarwal and S. Har-Peled, "Maintaining the approximate extent measures of moving points," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 148–157, Washington, DC: ACM Press, January 2001.

[20] A. Aggarwal, B. Alpern, A. K. Chandra, and M. Snir, "A model for hierarchical memory," in *Proceedings of the ACM Symposium on Theory of Computing*, pp. 305–314, New York: ACM Press, 1987.

[21] A. Aggarwal, A. Chandra, and M. Snir, "Hierarchical memory with block transfer," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 204–216, Los Angeles, 1987.

[22] A. Aggarwal and C. G. Plaxton, "Optimal parallel sorting in multi-level storage," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 659–668, ACM Press, 1994.

[23] A. Aggarwal and J. S. Vitter, "The Input/Output complexity of sorting and related problems," *Communications of the ACM*, vol. 31, no. 9, pp. 1116–1127, 1988.

[24] C. Aggarwal, *Data Streams: Models and Algorithms*. Springer-Verlag, 2007.

[25] M. Ajtai, M. Fredman, and J. Komlós, "Hash functions for priority queues," *Information and Control*, vol. 63, no. 3, pp. 217–225, 1984.

[26] D. Ajwani, I. Malinger, U. Meyer, and S. Toledo, "Characterizing the performance of flash memory storage devices and its impact on algorithm design," in *Proceedings of the International Workshop on Experimental Algorithmics*, (Provincetown, Mass.), pp. 208–219, Springer-Verlag, 2008.

[27] D. Ajwani, U. Meyer, and V. Osipov, "Improved external memory BFS implementation," in *Proceedings of the Workshop on Algorithm Engineering and Experiments*, (New Orleans), pp. 3–12, January 2007.

[28] S. Albers and M. Büttner, "Integrated prefetching and caching in single and parallel disk systems," *Information and Computation*, vol. 198, no. 1, pp. 24–39, 2005.

[29] B. Alpern, L. Carter, E. Feig, and T. Selker, "The uniform memory hierarchy model of computation," *Algorithmica*, vol. 12, no. 2–3, pp. 72–109, 1994.

[30] L. Arge, "The I/O-complexity of ordered binary-decision diagram manipulation," in *Proceedings of the International Symposium on Algorithms and Computation*, pp. 82–91, Springer-Verlag, 1995.

[31] L. Arge, "External memory data structures," in *Handbook of Massive Data Sets*, (J. Abello, P. M. Pardalos, and M. G. Resende, eds.), ch. 9, pp. 313–358, Norwell, Mass.: Kluwer Academic Publishers, 2002.

[32] L. Arge, "The buffer tree: A technique for designing batched external data structures," *Algorithmica*, vol. 37, no. 1, pp. 1–24, 2003.

[33] L. Arge, G. Brodal, and R. Fagerberg, "Cache-oblivious data structures," in *Handbook on Data Structures and Applications*, (D. Mehta and S. Sahni, eds.), CRC Press, 2005.

[34] L. Arge, G. S. Brodal, and L. Toma, "On external-memory MST, SSSP, and multi-way planar graph separation," *Journal of Algorithms*, vol. 53, no. 2, pp. 186–206, 2004.

[35] L. Arge, J. S. Chase, P. Halpin, L. Toma, J. S. Vitter, D. Urban, and R. Wickremesinghe, "Efficient flow computation on massive grid datasets," *GeoInformatica*, vol. 7, pp. 283–313, December 2003.

[36] L. Arge, A. Danner, H. J. Haverkort, and N. Zeh, "I/O-efficient hierarchical watershed decomposition of grid terrains models," in *Proceedings of the International Symposium on Spatial Data Handling*, 2006.

[37] L. Arge, A. Danner, and S.-H. Teh, "I/O-efficient point location using persistent B-trees," in *Workshop on Algorithm Engineering and Experimentation*, 2003.

[38] L. Arge, M. de Berg, H. J. Haverkort, and K. Yi, "The priority R-tree: A practically efficient and worst-case optimal R-tree," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 347–358, ACM Press, 2004.

[39] L. Arge, P. Ferragina, R. Grossi, and J. S. Vitter, "On sorting strings in external memory," in *Proceedings of the ACM Symposium on Theory of Computing*, pp. 540–548, ACM Press, 1997.

[40] L. Arge, M. T. Goodrich, M. Nelson, and N. Sitchinava, "Fundamental parallel algorithms for private-cache chip multiprocessors," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, Munich: ACM Press, June 2008.

[41] L. Arge, K. H. Hinrichs, J. Vahrenhold, and J. S. Vitter, “Efficient bulk operations on dynamic R-trees,” *Algorithmica*, vol. 33, pp. 104–128, January 2002.
[42] L. Arge, M. Knudsen, and K. Larsen, “A general lower bound on the I/O-complexity of comparison-based algorithms,” in *Proceedings of the Workshop on Algorithms and Data Structures*, pp. 83–94, Springer-Verlag, 1993.
[43] L. Arge, U. Meyer, and L. Toma, “External memory algorithms for diameter and all-pairs shortest-paths on sparse graphs,” in *Proceedings of the International Colloquium on Automata, Languages, and Programming*, pp. 146–157, Springer-Verlag, 2004.
[44] L. Arge, U. Meyer, L. Toma, and N. Zeh, “On external-memory planar depth first search,” *Journal of Graph Algorithms and Applications*, vol. 7, no. 2, pp. 105–129, 2003.
[45] L. Arge and P. Miltersen, “On showing lower bounds for external-memory computational geometry problems,” in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 139–159, Providence, Rhode Island: American Mathematical Society Press, 1999.
[46] L. Arge, O. Procopiuc, S. Ramaswamy, T. Suel, J. Vahrenhold, and J. S. Vitter, “A unified approach for indexed and non-indexed spatial joins,” in *Proceedings of the International Conference on Extending Database Technology*, Konstanz, Germany: ACM Press, March 2000.
[47] L. Arge, O. Procopiuc, S. Ramaswamy, T. Suel, and J. S. Vitter, “Scalable sweeping-based spatial join,” in *Proceedings of the International Conference on Very Large Databases*, pp. 570–581, New York: Morgan Kaufmann, August 1998.
[48] L. Arge, O. Procopiuc, S. Ramaswamy, T. Suel, and J. S. Vitter, “Theory and practice of I/O-efficient algorithms for multidimensional batched searching problems,” in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 685–694, ACM Press, January 1998.
[49] L. Arge, O. Procopiuc, and J. S. Vitter, “Implementing I/O-efficient data structures using TPIE,” in *Proceedings of the European Symposium on Algorithms*, pp. 88–100, Springer-Verlag, 2002.
[50] L. Arge, V. Samoladas, and J. S. Vitter, “Two-dimensional indexability and optimal range search indexing,” in *Proceedings of the ACM Conference on Principles of Database Systems*, pp. 346–357, Philadelphia: ACM Press, May–June 1999.
[51] L. Arge, V. Samoladas, and K. Yi, “Optimal external memory planar point enclosure,” in *Proceedings of the European Symposium on Algorithms*, pp. 40–52, Springer-Verlag, 2004.
[52] L. Arge and L. Toma, “Simplified external memory algorithms for planar DAGs,” in *Proceedings of the Scandinavian Workshop on Algorithmic Theory*, pp. 493–503, 2004.
[53] L. Arge and L. Toma, “External data structures for shortest path queries on planar digraphs,” in *Proceedings of the International Symposium on Algorithms and Computation*, pp. 328–338, Springer-Verlag, 2005.
[54] L. Arge, L. Toma, and J. S. Vitter, “I/O-efficient algorithms for problems on grid-based terrains,” in *Workshop on Algorithm Engineering and Experimentation*, San Francisco: Springer-Verlag, January 2000.

[55] L. Arge, L. Toma, and N. Zeh, "I/O-efficient topological sorting of planar DAGs," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, pp. 85–93, ACM Press, 2003.

[56] L. Arge and J. Vahrenhold, "I/O-efficient dynamic planar point location," *Computational Geometry*, vol. 29, no. 2, pp. 147–162, 2004.

[57] L. Arge, D. E. Vengroff, and J. S. Vitter, "External-memory algorithms for processing line segments in geographic information systems," *Algorithmica*, vol. 47, pp. 1–25, January 2007.

[58] L. Arge and J. S. Vitter, "Optimal external memory interval management," *SIAM Journal on Computing*, vol. 32, no. 6, pp. 1488–1508, 2003.

[59] L. Arge and N. Zeh, "I/O-efficient strong connectivity and depth-first search for directed planar graphs," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 261–270, 2003.

[60] C. Armen, "Bounds on the separation of two parallel disk models," in *Proceedings of the Workshop on Input/Output in Parallel and Distributed Systems*, pp. 122–127, May 1996.

[61] D. Arroyuelo and G. Navarro, "A Lempel–Ziv text index on secondary storage," in *Proceedings of the Symposium on Combinatorial Pattern Matching*, pp. 83–94, Springer-Verlag, 2007.

[62] M. J. Atallah and S. Prabhakar, "(Almost) optimal parallel block access for range queries," *Information Sciences*, vol. 157, pp. 21–31, 2003.

[63] R. A. Baeza-Yates, "Expected behaviour of B^+-trees under random insertions," *Acta Informatica*, vol. 26, no. 5, pp. 439–472, 1989.

[64] R. A. Baeza-Yates, "Bounded disorder: The effect of the index," *Theoretical Computer Science*, vol. 168, pp. 21–38, 1996.

[65] R. A. Baeza-Yates and P.-A. Larson, "Performance of B^+-trees with partial expansions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 1, pp. 248–257, June 1989.

[66] R. A. Baeza-Yates and B. Ribeiro-Neto, eds., *Modern Information Retrieval*. Addison Wesley Longman, 1999.

[67] R. A. Baeza-Yates and H. Soza-Pollman, "Analysis of linear hashing revisited," *Nordic Journal of Computing*, vol. 5, pp. 70–85, 1998.

[68] R. D. Barve, E. F. Grove, and J. S. Vitter, "Simple randomized mergesort on parallel disks," *Parallel Computing*, vol. 23, no. 4, pp. 601–631, 1997.

[69] R. D. Barve, M. Kallahalla, P. J. Varman, and J. S. Vitter, "Competitive analysis of buffer management algorithms," *Journal of Algorithms*, vol. 36, pp. 152–181, August 2000.

[70] R. D. Barve, E. A. M. Shriver, P. B. Gibbons, B. K. Hillyer, Y. Matias, and J. S. Vitter, "Modeling and optimizing I/O throughput of multiple disks on a bus," in *Procedings of ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 83–92, Atlanta: ACM Press, May 1999.

[71] R. D. Barve and J. S. Vitter, "A theoretical framework for memory-adaptive algorithms," in *Proceedings of the IEEE Symposium on Foundations of*

Computer Science, pp. 273–284, New York: IEEE Computer Society Press, October 1999.

[72] R. D. Barve and J. S. Vitter, "A simple and efficient parallel disk mergesort," *ACM Trans. Computer Systems*, vol. 35, pp. 189–215, March/April 2002.

[73] J. Basch, L. J. Guibas, and J. Hershberger, "Data structures for mobile data," *Journal of Algorithms*, vol. 31, pp. 1–28, 1999.

[74] R. Bayer and E. McCreight, "Organization of large ordered indexes," *Acta Informatica*, vol. 1, pp. 173–189, 1972.

[75] R. Bayer and K. Unterauer, "Prefix B-trees," *ACM Transactions on Database Systems*, vol. 2, pp. 11–26, March 1977.

[76] B. Becker, S. Gschwind, T. Ohler, B. Seeger, and P. Widmayer, "An asymptotically optimal multiversion B-tree," *VLDB Journal*, vol. 5, pp. 264–275, December 1996.

[77] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and rectangles," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 322–331, ACM Press, 1990.

[78] A. L. Belady, "A study of replacement algorithms for virtual storage computers," *IBM Systems Journal*, vol. 5, pp. 78–101, 1966.

[79] M. A. Bender, E. D. Demaine, and M. Farach-Colton, "Cache-oblivious B-trees," *SIAM Journal on Computing*, vol. 35, no. 2, pp. 341–358, 2005.

[80] M. A. Bender, M. Farach-Colton, and B. Kuszmaul, "Cache-oblivious string B-trees," in *Proceedings of the ACM Conference on Principles of Database Systems*, pp. 233–242, ACM Press, 2006.

[81] M. A. Bender, J. T. Fineman, S. Gilbert, and B. C. Kuszmaul, "Concurrent cache-oblivious B-trees," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, pp. 228–237, ACM Press, 2005.

[82] J. L. Bentley, "Multidimensional divide and conquer," *Communications of the ACM*, vol. 23, no. 6, pp. 214–229, 1980.

[83] J. L. Bentley and J. B. Saxe, "Decomposable searching problems I: Static-to-dynamic transformations," *Journal of Algorithms*, vol. 1, pp. 301–358, December 1980.

[84] S. Berchtold, C. Böhm, and H.-P. Kriegel, "Improving the query performance of high-dimensional index structures by bulk load operations," in *Proceedings of the International Conference on Extending Database Technology*, pp. 216–230, Springer-Verlag, 1998.

[85] M. Berger, E. R. Hansen, R. Pagh, M. Pătraşcu, M. Ružić, and P. Tiedemann, "Deterministic load balancing and dictionaries in the parallel disk model," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, ACM Press, 2006.

[86] R. Bhatia, R. K. Sinha, and C.-M. Chen, "A hierarchical technique for constructing efficient declustering schemes for range queries," *The Computer Journal*, vol. 46, no. 4, pp. 358–377, 2003.

[87] N. Blum and K. Mehlhorn, "On the average number of rebalancing operations in weight-balanced trees," *Theoretical Computer Science*, vol. 11, pp. 303–320, July 1980.

[88] R. S. Boyer and J. S. Moore, "A fast string searching algorithm," *Communications of the ACM*, vol. 20, pp. 762–772, October 1977.

[89] C. Breimann and J. Vahrenhold, "External memory computational geometry revisited," in *Algorithms for Memory Hierarchies*, pp. 110–148, 2002.

[90] K. Brengel, A. Crauser, P. Ferragina, and U. Meyer, "An experimental study of priority queues in external memory," *ACM Journal of Experimental Algorithmics*, vol. 5, p. 17, 2000.

[91] G. S. Brodal and R. Fagerberg, "Lower bounds for external memory dictionaries," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 546–554, ACM Press, 2003.

[92] G. S. Brodal and J. Katajainen, "Worst-case efficient external-memory priority queues," in *Proceedings of the Scandinavian Workshop on Algorithmic Theory*, pp. 107–118, Stockholm: Springer-Verlag, July 1998.

[93] A. L. Buchsbaum, M. Goldwasser, S. Venkatasubramanian, and J. R. Westbrook, "On external memory graph traversal," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 859–860, ACM Press, January 2000.

[94] P. Callahan, M. T. Goodrich, and K. Ramaiyer, "Topology B-trees and their applications," in *Proceedings of the Workshop on Algorithms and Data Structures*, pp. 381–392, Springer-Verlag, 1995.

[95] P. Cao, E. W. Felten, A. R. Karlin, and K. Li, "Implementation and performance of integrated application-controlled file caching, prefetching and disk scheduling," *ACM Transactions on Computer Systems*, vol. 14, pp. 311–343, November 1996.

[96] L. Carter and K. S. Gatlin, "Towards an optimal bit-reversal permutation program," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 544–553, November 1998.

[97] G. Chaudhry and T. H. Cormen, "Oblivious vs. distribution-based sorting: An experimental evaluation," in *Proceedings of the European Symposium on Algorithms*, pp. 317–328, Springer-Verlag, 2005.

[98] B. Chazelle, "Filtering search: A new approach to query-answering," *SIAM Journal on Computing*, vol. 15, pp. 703–724, 1986.

[99] B. Chazelle, "Lower bounds for orthogonal range searching: I. The reporting case," *Journal of the ACM*, vol. 37, pp. 200–212, April 1990.

[100] B. Chazelle and H. Edelsbrunner, "Linear space data structures for two types of range search," *Discrete and Computational Geometry*, vol. 2, pp. 113–126, 1987.

[101] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "RAID: High-performance, reliable secondary storage," *ACM Computing Surveys*, vol. 26, pp. 145–185, June 1994.

[102] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter, "Efficient indexing methods for probabilistic threshold queries over uncertain data," in *Proceedings of the International Conference on Very Large Databases*, Toronto: Morgan Kaufmann, August 2004.

[103] R. Cheng, Y. Xia, S. Prabhakar, R. Shah, and J. S. Vitter, "Efficient join processing over uncertain-valued attributes," in *Proceedings of the International*

ACM Conference on Information and Knowledge Management, Arlington, Va.: ACM Press, November 2006.

[104] Y.-J. Chiang, "Experiments on the practical I/O efficiency of geometric algorithms: Distribution sweep vs. plane sweep," *Computational Geometry: Theory and Applications*, vol. 8, no. 4, pp. 211–236, 1998.

[105] Y.-J. Chiang, M. T. Goodrich, E. F. Grove, R. Tamassia, D. E. Vengroff, and J. S. Vitter, "External-memory graph algorithms," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 139–149, ACM Press, January 1995.

[106] Y.-J. Chiang and C. T. Silva, "External memory techniques for isosurface extraction in scientific visualization," in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 247–277, Providence, Rhode Island: American Mathematical Society Press, 1999.

[107] Y.-F. Chien, W.-K. Hon, R. Shah, and J. S. Vitter, "Geometric Burrows–Wheeler transform: Linking range searching and text indexing," in *Proceedings of the Data Compression Conference*, Snowbird, Utah: IEEE Computer Society Press, March 2008.

[108] R. A. Chowdhury and V. Ramachandran, "External-memory exact and approximate all-pairs shortest-paths in undirected graphs," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 735–744, ACM Press, 2005.

[109] V. Ciriani, P. Ferragina, F. Luccio, and S. Muthukrishnan, "Static optimality theorem for external memory string access," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 219–227, 2002.

[110] D. R. Clark and J. I. Munro, "Efficient suffix trees on secondary storage," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 383–391, Atlanta: ACM Press, June 1996.

[111] K. L. Clarkson and P. W. Shor, "Applications of random sampling in computational geometry, II," *Discrete and Computational Geometry*, vol. 4, pp. 387–421, 1989.

[112] F. Claude and G. Navarro, "A fast and compact Web graph representation," in *Proceedings of the International Symposium on String Processing and Information Retrieval*, pp. 105–116, Springer-Verlag, 2007.

[113] A. Colvin and T. H. Cormen, "ViC*: A compiler for virtual-memory C*," in *Proceedings of the International Workshop on High-Level Programming Models and Supportive Environments*, pp. 23–33, 1998.

[114] D. Comer, "The ubiquitous B-Tree," *ACM Computing Surveys*, vol. 11, no. 2, pp. 121–137, 1979.

[115] P. Corbett, D. Feitelson, S. Fineberg, Y. Hsu, B. Nitzberg, J.-P. Prost, M. Snir, B. Traversat, and P. Wong, "Overview of the MPI-IO parallel I/O interface," in *Input/Output in Parallel and Distributed Computer Systems*, (R. Jain, J. Werth, and J. C. Browne, eds.), ch. 5, pp. 127–146, Kluwer Academic Publishers, 1996.

[116] P. F. Corbett and D. G. Feitelson, "The Vesta parallel file system," *ACM Transactions on Computer Systems*, vol. 14, pp. 225–264, August 1996.

[117] T. H. Cormen and E. R. Davidson, "FG: A framework generator for hiding latency in parallel programs running on clusters," in *Proceedings of the 17th International Conference on Parallel and Distributed Computing Systems*, pp. 137–144, Sep. 2004.

[118] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*. Cambridge, Mass.: MIT Press, 2nd ed., September 2001.

[119] T. H. Cormen and D. M. Nicol, "Performing out-of-core FFTs on parallel disk systems," *Parallel Computing*, vol. 24, pp. 5–20, January 1998.

[120] T. H. Cormen, T. Sundquist, and L. F. Wisniewski, "Asymptotically tight bounds for performing BMMC permutations on parallel disk systems," *SIAM Journal on Computing*, vol. 28, no. 1, pp. 105–136, 1999.

[121] A. Crauser and P. Ferragina, "A theoretical and experimental study on the construction of suffix arrays in external memory," *Algorithmica*, vol. 32, no. 1, pp. 1–35, 2002.

[122] A. Crauser, P. Ferragina, K. Mehlhorn, U. Meyer, and E. A. Ramos, "I/O-optimal computation of segment intersections," in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 131–138, Providence, Rhode Island: American Mathematical Society Press, 1999.

[123] A. Crauser, P. Ferragina, K. Mehlhorn, U. Meyer, and E. A. Ramos, "Randomized external-memory algorithms for line segment intersection and other geometric problems," *International Journal of Computational Geometry and Applications*, vol. 11, no. 3, pp. 305–337, 2001.

[124] A. Crauser and K. Mehlhorn, "LEDA-SM: Extending LEDA to secondary memory," in *Proceedings of the Workshop on Algorithm Engineering*, (J. S. Vitter and C. Zaroliagis, eds.), pp. 228–242, London: Springer-Verlag, July 1999.

[125] K. Curewitz, P. Krishnan, and J. S. Vitter, "Practical Prefetching Via Data Compression," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 257–266, Washington, DC: ACM Press, May 1993.

[126] R. Cypher and G. Plaxton, "Deterministic sorting in nearly logarithmic time on the hypercube and related computers," *Journal of Computer and System Sciences*, vol. 47, no. 3, pp. 501–548, 1993.

[127] E. R. Davidson, *FG: Improving Parallel Programs and Parallel Programming Since 2003*. PhD thesis, Dartmouth College Department of Computer Science, Aug. 2006.

[128] M. de Berg, J. Gudmundsson, M. Hammar, and M. H. Overmars, "On R-trees with low query complexity," *Computational Geometry*, vol. 24, no. 3, pp. 179–195, 2003.

[129] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, *Computational Geometry Algorithms and Applications*. Berlin: Springer-Verlag, 1997.

[130] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," in *Proceedings of the Symposium on Operating Systems Design and Implementation*, pp. 137–150, USENIX, December 2004.

[131] F. K. H. A. Dehne, W. Dittrich, and D. A. Hutchinson, "Efficient External Memory Algorithms by Simulating Coarse-Grained Parallel Algorithms," *Algorithmica*, vol. 36, no. 2, pp. 97–122, 2003.

[132] F. K. H. A. Dehne, W. Dittrich, D. A. Hutchinson, and A. Maheshwari, "Bulk synchronous parallel algorithms for the external memory model," *Theory of Computing Systems*, vol. 35, no. 6, pp. 567–597, 2002.
[133] R. Dementiev, *Algorithm Engineering for Large Data Sets*. PhD thesis, Saarland University, 2006.
[134] R. Dementiev, J. Kärkkäinen, J. Mehnert, and P. Sanders, "Better external memory suffix array construction," *ACM Journal of Experimental Algorithmics*, in press.
[135] R. Dementiev, L. Kettner, and P. Sanders, "STXXL: Standard Template Library for XXL Data Sets," *Software — Practice and Experience*, vol. 38, no. 6, pp. 589–637, 2008.
[136] R. Dementiev and P. Sanders, "Asynchronous parallel disk sorting," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, pp. 138–148, ACM Press, 2003.
[137] R. Dementiev, P. Sanders, D. Schultes, and J. Sibeyn, "Engineering an external memory minimum spanning tree algorithm," in *Proceedings of IFIP International Conference on Theoretical Computer Science*, Toulouse: Kluwer Academic Publishers, 2004.
[138] H. B. Demuth, *Electronic data sorting*. PhD thesis, Stanford University, 1956.
[139] P. J. Denning, "Working sets past and present," *IEEE Transactions on Software Engineering*, vol. SE-6, pp. 64–84, 1980.
[140] D. J. DeWitt, J. F. Naughton, and D. A. Schneider, "Parallel sorting on a shared-nothing architecture using probabilistic splitting," in *Proceedings of the International Conference on Parallel and Distributed Information Systems*, pp. 280–291, December 1991.
[141] W. Dittrich, D. A. Hutchinson, and A. Maheshwari, "Blocking in parallel multisearch problems," *Theory of Computing Systems*, vol. 34, no. 2, pp. 145–189, 2001.
[142] J. R. Driscoll, N. Sarnak, D. D. Sleator, and R. E. Tarjan, "Making data structures persistent," *Journal of Computer and System Sciences*, vol. 38, pp. 86–124, 1989.
[143] M. C. Easton, "Key-sequence data sets on indelible storage," *IBM Journal of Research and Development*, vol. 30, pp. 230–241, 1986.
[144] H. Edelsbrunner, "A new approach to rectangle intersections, Part I," *International Journal of Computer Mathematics*, vol. 13, pp. 209–219, 1983.
[145] H. Edelsbrunner, "A New approach to rectangle intersections, Part II," *International Journal of Computer Mathematics*, vol. 13, pp. 221–229, 1983.
[146] M. Y. Eltabakh, W.-K. Hon, R. Shah, W. Aref, and J. S. Vitter, "The SBC-tree: An index for run-length compressed sequences," in *Proceedings of the International Conference on Extending Database Technology*, Nantes, France: Springer-Verlag, March 2008.
[147] R. J. Enbody and H. C. Du, "Dynamic hashing schemes," *ACM Computing Surveys*, vol. 20, pp. 85–113, June 1988.
[148] "NASA's Earth Observing System (EOS) web page, NASA Goddard Space Flight Center," http://eospso.gsfc.nasa.gov/.

[149] D. Eppstein, Z. Galil, G. F. Italiano, and A. Nissenzweig, "Sparsification — a technique for speeding up dynamic graph algorithms," *Journal of the ACM*, vol. 44, no. 5, pp. 669–696, 1997.

[150] J. Erickson, "Lower bounds for external algebraic decision trees," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 755–761, ACM Press, 2005.

[151] G. Evangelidis, D. B. Lomet, and B. Salzberg, "The hB$^{\Pi}$-tree: A multi-attribute index supporting concurrency, recovery and node consolidation," *VLDB Journal*, vol. 6, pp. 1–25, 1997.

[152] R. Fagerberg, A. Pagh, and R. Pagh, "External string sorting: Faster and cache oblivious," in *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, pp. 68–79, Springer-Verlag, 2006.

[153] R. Fagin, J. Nievergelt, N. Pippinger, and H. R. Strong, "Extendible hashing — a fast access method for dynamic files," *ACM Transactions on Database Systems*, vol. 4, no. 3, pp. 315–344, 1979.

[154] M. Farach-Colton, P. Ferragina, and S. Muthukrishnan, "On the sorting-complexity of suffix tree construction," *Journal of the ACM*, vol. 47, no. 6, pp. 987–1011, 2000.

[155] J. Feigenbaum, S. Kannan, M. Strauss, and M. Viswanathan, "An approximate L1-difference algorithm for massive data streams," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 131–151, 2002.

[156] W. Feller, *An Introduction to Probability Theory and its Applications.* Vol. 1, New York: John Wiley & Sons, 3rd ed., 1968.

[157] P. Ferragina and R. Grossi, "Fast string searching in secondary storage: Theoretical developments and experimental results," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 373–382, Atlanta: ACM Press, June 1996.

[158] P. Ferragina and R. Grossi, "The String B-tree: A new data structure for string search in external memory and its applications," *Journal of the ACM*, vol. 46, pp. 236–280, March 1999.

[159] P. Ferragina, R. Grossi, A. Gupta, R. Shah, and J. S. Vitter, "On searching compressed string collections cache-obliviously," in *Proceedings of the ACM Conference on Principles of Database Systems*, Vancouver: ACM Press, June 2008.

[160] P. Ferragina, N. Koudas, S. Muthukrishnan, and D. Srivastava, "Two-dimensional substring indexing," *Journal of Computer and System Sciences*, vol. 66, no. 4, pp. 763–774, 2003.

[161] P. Ferragina and F. Luccio, "Dynamic dictionary matching in external memory," *Information and Computation*, vol. 146, pp. 85–99, November 1998.

[162] P. Ferragina and G. Manzini, "Indexing compressed texts," *Journal of the ACM*, vol. 52, no. 4, pp. 552–581, 2005.

[163] P. Ferragina, G. Manzini, V. Mäkinen, and G. Navarro, "Compressed representations of sequences and full-text indexes," *ACM Transaction on Algorithms*, vol. 3, p. 20, May 2007.

[164] P. Flajolet, "On the performance evaluation of extendible hashing and trie searching," *Acta Informatica*, vol. 20, no. 4, pp. 345–369, 1983.

[165] R. W. Floyd, "Permuting information in idealized two-level storage," in *Complexity of Computer Computations*, (R. Miller and J. Thatcher, eds.), pp. 105–109, Plenum, 1972.

[166] W. Frakes and R. A. Baeza-Yates, eds., *Information Retrieval: Data Structures and Algorithms*. Prentice-Hall, 1992.

[167] G. Franceschini, R. Grossi, J. I. Munro, and L. Pagli, "Implicit B-Trees: A new data structure for the dictionary problem," *Journal of Computer and System Sciences*, vol. 68, no. 4, pp. 788–807, 2004.

[168] M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran, "Cache-oblivious algorithms," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 285–298, 1999.

[169] T. A. Funkhouser, C. H. Sequin, and S. J. Teller, "Management of large amounts of data in interactive building walkthroughs," in *Proceedings of the ACM Symposium on Interactive 3D Graphics*, pp. 11–20, Boston: ACM Press, March 1992.

[170] V. Gaede and O. Günther, "Multidimensional access methods," *ACM Computing Surveys*, vol. 30, pp. 170–231, June 1998.

[171] G. R. Ganger, "Generating representative synthetic workloads: An unsolved problem," in *Proceedings of the Computer Measurement Group Conference*, pp. 1263–1269, December 1995.

[172] M. Gardner, ch. 7, *Magic Show*. New York: Knopf, 1977.

[173] I. Gargantini, "An effective way to represent quadtrees," *Communications of the ACM*, vol. 25, pp. 905–910, December 1982.

[174] T. M. Ghanem, R. Shah, M. F. Mokbel, W. G. Aref, and J. S. Vitter, "Bulk operations for space-partitioning trees," in *Proceedings of IEEE International Conference on Data Engineering*, Boston: IEEE Computer Society Press, April 2004.

[175] G. A. Gibson, J. S. Vitter, and J. Wilkes, "Report of the working group on storage I/O issues in large-scale computing," *ACM Computing Surveys*, vol. 28, pp. 779–793, December 1996.

[176] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proceedings of the International Conference on Very Large Databases*, pp. 78–89, Edinburgh, Scotland: Morgan Kaufmann, 1999.

[177] R. Goldman, N. Shivakumar, S. Venkatasubramanian, and H. Garcia-Molina, "Proximity search in databases," in *Proceedings of the International Conference on Very Large Databases*, pp. 26–37, August 1998.

[178] R. González and G. Navarro, "A compressed text index on secondary memory," in *Proceedings of the International Workshop on Combinatorial Algorithms*, (Newcastle, Australia), pp. 80–91, College Publications, 2007.

[179] M. T. Goodrich, J.-J. Tsay, D. E. Vengroff, and J. S. Vitter, "External-memory computational geometry," in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pp. 714–723, Palo Alto: IEEE Computer Society Press, November 1993.

[180] "Google Earth online database of satellite images," Available on the World-Wide Web at http://earth.google.com/.

[181] S. Govindarajan, P. K. Agarwal, and L. Arge, "CRB-tree: An efficient indexing scheme for range-aggregate queries," in *Proceedings of the International Conference on Database Theory*, pp. 143–157, Springer-Verlag, 2003.

[182] S. Govindarajan, T. Lukovszki, A. Maheshwari, and N. Zeh, "I/O-efficient well-separated pair decomposition and its applications," *Algorithmica*, vol. 45, pp. 385–614, August 2006.

[183] D. Greene, "An implementation and performance analysis of spatial data access methods," in *Proceedings of IEEE International Conference on Data Engineering*, pp. 606–615, 1989.

[184] J. L. Griffin, S. W. Schlosser, G. R. Ganger, and D. F. Nagle, "Modeling and performance of MEMS-based storage devices," in *Procedings of ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 56–65, Santa Clara, Cal.: ACM Press, June 2000.

[185] R. Grossi, A. Gupta, and J. S. Vitter, "High-order entropy-compressed text indexes," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, ACM Press, January 2003.

[186] R. Grossi and G. F. Italiano, "Efficient cross-trees for external memory," in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 87–106, Providence, Rhode Island: American Mathematical Society Press, 1999.

[187] R. Grossi and G. F. Italiano, "Efficient splitting and merging algorithms for order decomposable problems," *Information and Computation*, vol. 154, no. 1, pp. 1–33, 1999.

[188] S. K. S. Gupta, Z. Li, and J. H. Reif, "Generating efficient programs for two-level memories from tensor-products," in *Proceedings of the IASTED/ISMM International Conference on Parallel and Distributed Computing and Systems*, pp. 510–513, October 1995.

[189] D. Gusfield, *Algorithms on Strings, Trees, and Sequences.* Cambridge, UK: Cambridge University Press, 1997.

[190] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 47–57, ACM Press, 1984.

[191] H. J. Haverkort and L. Toma, "I/O-efficient algorithms on near-planar graphs," in *Proceedings of the Latin American Theoretical Informatics Symposium*, pp. 580–591, 2006.

[192] T. Hazel, L. Toma, R. Wickremesinghe, and J. Vahrenhold, "Terracost: A versatile and scalable approach to computing least-cost-path surfaces for massive grid-based terrains," in *Proceedings of the ACM Symposium on Applied Computing*, pp. 52–57, ACM Press, 2006.

[193] J. M. Hellerstein, E. Koutsoupias, and C. H. Papadimitriou, "On the analysis of indexing schemes," in *Proceedings of the ACM Symposium on Principles of Database Systems*, pp. 249–256, Tucson: ACM Press, May 1997.

[194] L. Hellerstein, G. Gibson, R. M. Karp, R. H. Katz, and D. A. Patterson, "Coding techniques for handling failures in large disk arrays," *Algorithmica*, vol. 12, no. 2–3, pp. 182–208, 1994.

[195] M. R. Henzinger, P. Raghavan, and S. Rajagopalan, "Computing on data streams," in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 107–118, Providence, Rhode Island: American Mathematical Society Press, 1999.

[196] K. Hinrichs, "Implementation of the grid file: Design concepts and experience," *BIT*, vol. 25, no. 4, pp. 569–592, 1985.

[197] W.-K. Hon, T.-W. Lam, R. Shah, S.-L. Tam, and J. S. Vitter, "Cache-oblivious index for approximate string matching," in *Proceedings of the Symposium on Combinatorial Pattern Matching*, pp. 40–51, London, Ontario, Canada: Springer-Verlag, July 2007.

[198] W.-K. Hon, R. Shah, P. J. Varman, and J. S. Vitter, "Tight competitive ratios for parallel disk prefetching and caching," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, Munich: ACM Press, June 2008.

[199] J. W. Hong and H. T. Kung, "I/O complexity: The red-blue pebble game," in *Proceedings of the ACM Symposium on Theory of Computing*, pp. 326–333, ACM Press, May 1981.

[200] D. Hutchinson, A. Maheshwari, J.-R. Sack, and R. Velicescu, "Early experiences in implementing the buffer tree," in *Proceedings of the Workshop on Algorithm Engineering*, Springer-Verlag, 1997.

[201] D. A. Hutchinson, A. Maheshwari, and N. Zeh, "An external memory data structure for shortest path queries," *Discrete Applied Mathematics*, vol. 126, no. 1, pp. 55–82, 2003.

[202] D. A. Hutchinson, P. Sanders, and J. S. Vitter, "Duality between prefetching and queued writing with parallel disks," *SIAM Journal on Computing*, vol. 34, no. 6, pp. 1443–1463, 2005.

[203] P. Indyk, R. Motwani, P. Raghavan, and S. Vempala, "Locality-preserving hashing in multidimensional spaces," in *Proceedings of the ACM Symposium on Theory of Computing*, pp. 618–625, El Paso: ACM Press, May 1997.

[204] M. Kallahalla and P. J. Varman, "Optimal prefetching and caching for parallel I/O systems," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, Crete, Greece: ACM Press, July 2001.

[205] M. Kallahalla and P. J. Varman, "Optimal read-once parallel disk scheduling," *Algorithmica*, vol. 43, no. 4, pp. 309–343, 2005.

[206] I. Kamel and C. Faloutsos, "On packing R-trees," in *Proceedings of the International ACM Conference on Information and Knowledge Management*, pp. 490–499, 1993.

[207] I. Kamel and C. Faloutsos, "Hilbert R-tree: An improved R-tree using fractals," in *Proceedings of the International Conference on Very Large Databases*, pp. 500–509, 1994.

[208] I. Kamel, M. Khalil, and V. Kouramajian, "Bulk insertion in dynamic R-trees," in *Proceedings of the International Symposium on Spatial Data Handling*, pp. 3B, 31–42, 1996.

[209] P. C. Kanellakis, G. M. Kuper, and P. Z. Revesz, "Constraint query languages," *Journal of Computer and System Sciences*, vol. 51, no. 1, pp. 26–52, 1995.

[210] P. C. Kanellakis, S. Ramaswamy, D. E. Vengroff, and J. S. Vitter, "Indexing for data models with constraints and classes," *Journal of Computer and System Sciences*, vol. 52, no. 3, pp. 589–612, 1996.

[211] K. V. R. Kanth and A. K. Singh, "Optimal dynamic range searching in non-replicating index structures," in *Proceedings of the International Conference on Database Theory*, pp. 257–276, Springer-Verlag, January 1999.

[212] J. Kärkkäinen and S. S. Rao, "Full-text indexes in external memory," in *Algorithms for Memory Hierarchies*, (U. Meyer, P. Sanders, and J. Sibeyn, eds.), ch. 7, pp. 149–170, Berlin: Springer-Verlag, 2003.

[213] I. Katriel and U. Meyer, "Elementary graph algorithms in external memory," in *Algorithms for Memory Hierarchies*, (U. Meyer, P. Sanders, and J. Sibeyn, eds.), ch. 4, pp. 62–84, Berlin: Springer-Verlag, 2003.

[214] R. Khandekar and V. Pandit, "Offline Sorting Buffers On Line," in *Proceedings of the International Symposium on Algorithms and Computation*, pp. 81–89, Springer-Verlag, December 2006.

[215] S. Khuller, Y. A. Kim, and Y.-C. J. Wan, "Algorithms for data migration with cloning," *SIAM Journal on Computing*, vol. 33, no. 2, pp. 448–461, 2004.

[216] M. Y. Kim, "Synchronized disk interleaving," *IEEE Transactions on Computers*, vol. 35, pp. 978–988, November 1986.

[217] T. Kimbrel and A. R. Karlin, "Near-optimal parallel prefetching and caching," *SIAM Journal on Computing*, vol. 29, no. 4, pp. 1051–1082, 2000.

[218] D. G. Kirkpatrick and R. Seidel, "The ultimate planar convex hull algorithm?," *SIAM Journal on Computing*, vol. 15, pp. 287–299, 1986.

[219] S. T. Klein and D. Shapira, "Searching in compressed dictionaries," in *Proceedings of the Data Compression Conference*, Snowbird, Utah: IEEE Computer Society Press, 2002.

[220] D. E. Knuth, *Sorting and Searching*. Vol. 3 of *The Art of Computer Programming*, Reading, MA: Addison-Wesley, 2nd ed., 1998.

[221] D. E. Knuth, *MMIXware*. Berlin: Springer-Verlag, 1999.

[222] D. E. Knuth, J. H. Morris, and V. R. Pratt, "Fast pattern matching in strings," *SIAM Journal on Computing*, vol. 6, pp. 323–350, 1977.

[223] G. Kollios, D. Gunopulos, and V. J. Tsotras, "On indexing mobile objects," in *Proceedings of the ACM Symposium on Principles of Database Systems*, pp. 261–272, ACM Press, 1999.

[224] E. Koutsoupias and D. S. Taylor, "Tight bounds for 2-dimensional indexing schemes," in *Proceedings of the ACM Symposium on Principles of Database Systems*, pp. 52–58, Seattle: ACM Press, June 1998.

[225] M. Kowarschik and C. Weiß, "An overview of cache optimizaiton techniques and cache-aware numerical algorithms," in *Algorithms for Memory Hierarchies*, (U. Meyer, P. Sanders, and J. Sibeyn, eds.), ch. 10, pp. 213–232, Berlin: Springer-Verlag, 2003.

[226] P. Krishnan and J. S. Vitter, "Optimal prediction for prefetching in the worst case," *SIAM Journal on Computing*, vol. 27, pp. 1617–1636, December 1998.

[227] V. Kumar and E. Schwabe, "Improved algorithms and data structures for solving graph problems in external memory," in *Proceedings of the IEEE Symposium on Parallel and Distributed Processing*, pp. 169–176, October 1996.

[228] K. Küspert, "Storage utilization in B*-trees with a generalized overflow technique," *Acta Informatica*, vol. 19, pp. 35–55, 1983.
[229] P.-A. Larson, "Performance analysis of linear hashing with partial expansions," *ACM Transactions on Database Systems*, vol. 7, pp. 566–587, December 1982.
[230] R. Laurini and D. Thompson, *Fundamentals of Spatial Information Systems,*. Academic Press, 1992.
[231] P. L. Lehman and S. B. Yao, "Efficient locking for concurrent operations on B-trees," *ACM Transactions on Database Systems*, vol. 6, pp. 650–570, December 1981.
[232] F. T. Leighton, "Tight bounds on the complexity of parallel sorting," *IEEE Transactions on Computers*, vol. C-34, pp. 344–354, Special issue on sorting, E. E. Lindstrom, C. K. Wong, and J. S. Vitter, eds., April 1985.
[233] C. E. Leiserson, S. Rao, and S. Toledo, "Efficient out-of-core algorithms for linear relaxation using blocking covers," *Journal of Computer and System Sciences*, vol. 54, no. 2, pp. 332–344, 1997.
[234] Z. Li, P. H. Mills, and J. H. Reif, "Models and resource metrics for parallel and distributed computation," *Parallel Algorithms and Applications*, vol. 8, pp. 35–59, 1996.
[235] W. Litwin, "Linear hashing: A new tool for files and tables addressing," in *Proceedings of the International Conference on Very Large Databases*, pp. 212–223, October 1980.
[236] W. Litwin and D. Lomet, "A new method for fast data searches with keys," *IEEE Software*, vol. 4, pp. 16–24, March 1987.
[237] D. Lomet, "A simple bounded disorder file organization with good performance," *ACM Transactions on Database Systems*, vol. 13, no. 4, pp. 525–551, 1988.
[238] D. B. Lomet and B. Salzberg, "The hB-tree: A multiattribute indexing method with good guaranteed performance," *ACM Transactions on Database Systems*, vol. 15, no. 4, pp. 625–658, 1990.
[239] D. B. Lomet and B. Salzberg, "Concurrency and recovery for index trees," *VLDB Journal*, vol. 6, no. 3, pp. 224–240, 1997.
[240] T. Lukovszki, A. Maheshwari, and N. Zeh, "I/O-efficient batched range counting and its applications to proximity problems," *Foundations of Software Technology and Theoretical Computer Science*, pp. 244–255, 2001.
[241] A. Maheshwari and N. Zeh, "I/O-efficient algorithms for graphs of bounded treewidth," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 89–90, Washington, DC: ACM Press, January 2001.
[242] A. Maheshwari and N. Zeh, "I/O-Optimal Algorithms for Planar Graphs Using Separators," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 372–381, ACM Press, 2002.
[243] A. Maheshwari and N. Zeh, "A survey of techniques for designing I/O-efficient algorithms," in *Algorithms for Memory Hierarchies*, (U. Meyer, P. Sanders, and J. Sibeyn, eds.), ch. 3, pp. 36–61, Berlin: Springer-Verlag, 2003.
[244] A. Maheshwari and N. Zeh, "I/O-optimal algorithms for outerplanar graphs," *Journal of Graph Algorithms and Applications*, vol. 8, pp. 47–87, 2004.

[245] V. Mäkinen, G. Navarro, and K. Sadakane, "Advantages of backward searching — efficient secondary memory and distributed implementation of compressed suffix arrays," in *Proceedings of the International Symposium on Algorithms and Computation*, pp. 681–692, Springer-Verlag, 2004.

[246] U. Manber and G. Myers, "Suffix arrays: A new method for on-line string searches," *SIAM Journal on Computing*, vol. 22, pp. 935–948, October 1993.

[247] U. Manber and S. Wu, "GLIMPSE: A tool to search through entire file systems," in *Proceedings of the Winter USENIX Conference*, (USENIX Association, ed.), pp. 23–32, San Francisco: USENIX, January 1994.

[248] G. N. N. Martin, "Spiral storage: Incrementally augmentable hash addressed storage," Technical Report CS-RR-027, University of Warwick, March 1979.

[249] Y. Matias, E. Segal, and J. S. Vitter, "Efficient bundle sorting," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 839–848, San Francisco: ACM Press, January 2000.

[250] E. M. McCreight, "A space-economical suffix tree construction algorithm," *Journal of the ACM*, vol. 23, no. 2, pp. 262–272, 1976.

[251] E. M. McCreight, "Priority Search Trees," *SIAM Journal on Computing*, vol. 14, pp. 257–276, May 1985.

[252] K. Mehlhorn and U. Meyer, "External-memory breadth-first search with sublinear I/O," in *Proceedings of the European Symposium on Algorithms*, pp. 723–735, Springer-Verlag, 2002.

[253] H. Mendelson, "Analysis of extendible hashing," *IEEE Transactions on Software Engineering*, vol. SE-8, pp. 611–619, November 1982.

[254] U. Meyer, "External memory BFS on undirected graphs with bounded degree," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 87–88, Washington, DC: ACM Press, January 2001.

[255] U. Meyer, "On dynamic breadth-first search in external-memory," in *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, (Schloss Dagstuhl, Germany), pp. 551–560, Internationales Begegnungs- und Forschungszentrum für Informatik, 2008.

[256] U. Meyer, "On trade-offs in external-memory diameter approximation," in *Proceedings of the Scandinavian Workshop on Algorithm Theory*, (Gothenburg, Sweden), Springer-Verlag, July 2008.

[257] U. Meyer, P. Sanders, and J. Sibeyn, eds., *Algorithms for Memory Hierarchies*. Berlin: Springer-Verlag, 2003.

[258] U. Meyer and N. Zeh, "I/O-efficient undirected shortest paths," in *Proceedings of the European Symposium on Algorithms*, pp. 435–445, Springer-Verlag, 2003.

[259] U. Meyer and N. Zeh, "I/O-efficient undirected shortest paths with unbounded weights," in *Proceedings of the European Symposium on Algorithms*, Springer-Verlag, 2006.

[260] C. Mohan, "ARIES/KVL: A key-value locking method for concurrency control of multiaction transactions on B-tree indices," in *Proceedings of the International Conference on Very Large Databases*, pp. 392–405, August 1990.

[261] D. R. Morrison, "Patricia: Practical algorithm to retrieve information coded in alphanumeric," *Journal of the ACM*, vol. 15, pp. 514–534, 1968.

[262] S. A. Moyer and V. Sunderam, "Characterizing concurrency control performance for the PIOUS parallel file system," *Journal of Parallel and Distributed Computing*, vol. 38, pp. 81–91, October 1996.

[263] J. K. Mullin, "Spiral storage: Efficient dynamic hashing with constant performance," *The Computer Journal*, vol. 28, pp. 330–334, July 1985.

[264] K. Munagala and A. Ranade, "I/O-complexity of graph algorithms," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 687–694, Baltimore: ACM Press, January 1999.

[265] S. Muthukrishnan, *Data Streams: Algorithms and Applications.* Vol. 1, issue 2 of *Foundations and Trends in Theoretical Computer Science*, Hanover, Mass.: now Publishers, 2005.

[266] G. Navarro, "Indexing text using the Ziv–Lempel trie," *Journal of Discrete Algorithms*, vol. 2, no. 1, pp. 87–114, 2004.

[267] G. Navarro and V. Mäkinen, "Compressed full-text indexes," *ACM Computing Surveys*, vol. 39, no. 1, p. 2, 2007.

[268] J. Nievergelt, H. Hinterberger, and K. C. Sevcik, "The grid file: An adaptable, symmetric multi-key file structure," *ACM Transactions on Database Systems*, vol. 9, pp. 38–71, 1984.

[269] J. Nievergelt and E. M. Reingold, "Binary search tree of bounded balance," *SIAM Journal on Computing*, vol. 2, pp. 33–43, March 1973.

[270] J. Nievergelt and P. Widmayer, "Spatial data structures: Concepts and design choices," in *Algorithmic Foundations of GIS*, (M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, eds.), pp. 153–197, Springer-Verlag, 1997.

[271] M. H. Nodine, M. T. Goodrich, and J. S. Vitter, "Blocking for external graph searching," *Algorithmica*, vol. 16, pp. 181–214, August 1996.

[272] M. H. Nodine, D. P. Lopresti, and J. S. Vitter, "I/O overhead and parallel VLSI architectures for lattice computations," *IEEE Transactions on Communications*, vol. 40, pp. 843–852, July 1991.

[273] M. H. Nodine and J. S. Vitter, "Deterministic distribution sort in shared and distributed memory multiprocessors," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, pp. 120–129, Velen, Germany: ACM Press, June–July 1993.

[274] M. H. Nodine and J. S. Vitter, "Greed Sort: An optimal sorting algorithm for multiple disks," *Journal of the ACM*, vol. 42, pp. 919–933, July 1995.

[275] P. E. O'Neil, "The SB-tree. An index-sequential structure for high-performance sequential access," *Acta Informatica*, vol. 29, pp. 241–265, June 1992.

[276] J. A. Orenstein, "Redundancy in spatial databases," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 294–305, Portland: ACM Press, June 1989.

[277] J. A. Orenstein and T. H. Merrett, "A class of data structures for associative searching," in *Proceedings of the ACM Conference on Principles of Database Systems*, pp. 181–190, ACM Press, 1984.

[278] M. H. Overmars, *The design of dynamic data structures.* 1983. Springer-Verlag.

[279] H. Pang, M. Carey, and M. Livny, "Memory-adaptive external sorts," in *Proceedings of the International Conference on Very Large Databases*, pp. 618–629, 1993.

[280] H. Pang, M. J. Carey, and M. Livny, "Partially preemptive hash joins," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, (P. Buneman and S. Jajodia, eds.), pp. 59–68, Washington, DC: ACM Press, May 1993.

[281] I. Parsons, R. Unrau, J. Schaeffer, and D. Szafron, "PI/OT: Parallel I/O templates," *Parallel Computing*, vol. 23, pp. 543–570, June 1997.

[282] J. M. Patel and D. J. DeWitt, "Partition based spatial-merge join," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 259–270, ACM Press, June 1996.

[283] D. Pfoser, C. S. Jensen, and Y. Theodoridis, "Novel approaches to the indexing of moving object trajectories," in *Proceedings of the International Conference on Very Large Databases*, pp. 395–406, 2000.

[284] F. P. Preparata and M. I. Shamos, *Computational Geometry.* Berlin: Springer-Verlag, 1985.

[285] O. Procopiuc, P. K. Agarwal, L. Arge, and J. S. Vitter, "Bkd-tree: A dynamic scalable kd-tree," in *Proceedings of the International Symposium on Spatial and Temporal Databases*, Santorini, Greece: Springer-Verlag, July 2003.

[286] S. J. Puglisi, W. F. Smyth, and A. Turpin, "Inverted files versus suffix arrays for locating patterns in primary memory," in *Proceedings of the International Symposium on String Processing Information Retrieval*, pp. 122–133, Springer-Verlag, 2006.

[287] N. Rahman and R. Raman, "Adapting radix sort to the memory hierarchy," in *Workshop on Algorithm Engineering and Experimentation*, Springer-Verlag, January 2000.

[288] R. Raman, V. Raman, and S. S. Rao, "Succinct indexable dictionaries with applications to encoding k-ary trees and multisets," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 233–242, ACM Press, 2002.

[289] S. Ramaswamy and S. Subramanian, "Path caching: A technique for optimal external searching," in *Proceedings of the ACM Conference on Principles of Database Systems*, pp. 25–35, Minneapolis: ACM Press, 1994.

[290] J. Rao and K. Ross, "Cache conscious indexing for decision-support in main memory," in *Proceedings of the International Conference on Very Large Databases*, (M. Atkinson *et al.*, eds.), pp. 78–89, Los Altos, Cal.: Morgan Kaufmann, 1999.

[291] J. Rao and K. A. Ross, "Making B^+-trees cache conscious in main memory," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, (W. Chen, J. Naughton, and P. A. Bernstein, eds.), pp. 475–486, Dallas: ACM Press, 2000.

[292] E. Riedel, G. A. Gibson, and C. Faloutsos, "Active storage for large-scale data mining and multimedia," in *Proceedings of the International Conference on Very Large Databases*, pp. 62–73, August 1998.

[293] J. T. Robinson, "The k-d-b-tree: A search structure for large multidimensional dynamic indexes," in *Proceedings of the ACM Conference on Principles of Database Systems*, pp. 10–18, ACM Press, 1981.
[294] M. Rosenblum, E. Bugnion, S. Devine, and S. A. Herrod, "Using the SimOS machine simulator to study complex computer systems," *ACM Transactions on Modeling and Computer Simulation*, vol. 7, pp. 78–103, January 1997.
[295] C. Ruemmler and J. Wilkes, "An introduction to disk drive modeling," *IEEE Computer*, pp. 17–28, March 1994.
[296] K. Salem and H. Garcia-Molina, "Disk striping," in *Proceedings of IEEE International Conference on Data Engineering*, pp. 336–242, Los Angeles, 1986.
[297] S. Šaltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, "Indexing the positions of continuously moving objects," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, (W. Chen, J. Naughton, and P. A. Bernstein, eds.), pp. 331–342, Dallas: ACM Press, 2000.
[298] B. Salzberg and V. J. Tsotras, "Comparison of access methods for time-evolving data," *ACM Computing Surveys*, vol. 31, pp. 158–221, June 1999.
[299] H. Samet, *Applications of Spatial Data Structures: Computer Graphics, Image Processing, and GIS*. Addison-Wesley, 1989.
[300] H. Samet, *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1989.
[301] V. Samoladas and D. Miranker, "A lower bound theorem for indexing schemes and its application to multidimensional range queries," in *Proceedings of the ACM Symposium on Principles of Database Systems*, pp. 44–51, Seattle: ACM Press, June 1998.
[302] P. Sanders, "Fast priority queues for cached memory," *ACM Journal of Experimental Algorithmics*, vol. 5, no. 7, pp. 1–25, 2000.
[303] P. Sanders, "Reconciling simplicity and realism in parallel disk models," *Parallel Computing*, vol. 28, no. 5, pp. 705–723, 2002.
[304] P. Sanders, S. Egner, and J. Korst, "Fast concurrent access to parallel disks," *Algorithmica*, vol. 35, no. 1, pp. 21–55, 2002.
[305] J. E. Savage, "Extending the Hong-Kung model to memory hierarchies," in *Proceedings of the International Conference on Computing and Combinatorics*, pp. 270–281, Springer-Verlag, August 1995.
[306] J. E. Savage and J. S. Vitter, "Parallelism in space-time tradeoffs," in *Advances in Computing Research*, (F. P. Preparata, ed.), pp. 117–146, JAI Press, 1987.
[307] S. W. Schlosser, J. L. Griffin, D. F. Nagle, and G. R. Ganger, "Designing computer systems with MEMS-based storage," in *Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 1–12, November 2000.
[308] K. E. Seamons and M. Winslett, "Multidimensional array I/O in Panda 1.0," *Journal of Supercomputing*, vol. 10, no. 2, pp. 191–211, 1996.
[309] B. Seeger and H.-P. Kriegel, "The buddy-tree: An efficient and robust access method for spatial data base systems," in *Proceedings of the International Conference on Very Large Databases*, pp. 590–601, 1990.

[310] M. Seltzer, K. A. Smith, H. Balakrishnan, J. Chang, S. McMains, and V. Padmanabhan, "File system logging versus clustering: A performance comparison," in *Proceedings of the Annual USENIX Technical Conference*, pp. 249–264, New Orleans, 1995.

[311] S. Sen, S. Chatterjee, and N. Dumir, "Towards a theory of cache-efficient algorithms," *Journal of the ACM*, vol. 49, no. 6, pp. 828–858, 2002.

[312] R. Shah, P. J. Varman, and J. S. Vitter, "Online algorithms for prefetching and caching on parallel disks," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, pp. 255–264, ACM Press, 2004.

[313] R. Shah, P. J. Varman, and J. S. Vitter, "On competitive online read-many parallel disks scheduling," in *Proceedings of the ACM Symposium on Parallel Algorithms and Architectures*, p. 217, ACM Press, 2005.

[314] E. A. M. Shriver, A. Merchant, and J. Wilkes, "An analytic behavior model for disk drives with readahead caches and request reordering," in *Procedings of ACM SIGMETRICS Joint International Conference on Measurement and Modeling of Computer Systems*, pp. 182–191, Madison, Wisc.: ACM Press, June 1998.

[315] E. A. M. Shriver and M. H. Nodine, "An introduction to parallel I/O models and algorithms," in *Input/Output in Parallel and Distributed Computer Systems*, (R. Jain, J. Werth, and J. C. Browne, eds.), ch. 2, pp. 31–68, Kluwer Academic Publishers, 1996.

[316] E. A. M. Shriver and L. F. Wisniewski, "An API for choreographing data accesses," Tech. Rep. PCS-TR95-267, Dept. of Computer Science, Dartmouth College, November 1995.

[317] J. F. Sibeyn, "From parallel to external list ranking," Technical Report MPI–I–97–1–021, Max-Planck-Institut, September 1997.

[318] J. F. Sibeyn, "External selection," *Journal of Algorithms*, vol. 58, no. 2, pp. 104–117, 2006.

[319] J. F. Sibeyn and M. Kaufmann, "BSP-like external-memory computation," in *Proceedings of the Italian Conference on Algorithms and Complexity*, pp. 229–240, 1997.

[320] R. Sinha, S. Puglisi, A. Moffat, and A. Turpin, "Improving suffix array locality for fast pattern matching on disk," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, Vancouver: ACM Press, June 2008.

[321] B. Srinivasan, "An adaptive overflow technique to defer splitting in B-trees," *The Computer Journal*, vol. 34, no. 5, pp. 397–405, 1991.

[322] A. Srivastava and A. Eustace, "ATOM: A system for building customized program analysis tools," *ACM SIGPLAN Notices*, vol. 29, pp. 196–205, June 1994.

[323] S. Subramanian and S. Ramaswamy, "The P-range tree: A new data structure for range searching in secondary memory," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pp. 378–387, ACM Press, 1995.

[324] R. Tamassia and J. S. Vitter, "Optimal cooperative search in fractional cascaded data structures," *Algorithmica*, vol. 15, pp. 154–171, February 1996.

[325] "TerraServer-USA: Microsoft's online database of satellite images," Available on the World-Wide Web at http://terraserver.microsoft.com/.

[326] R. Thakur, A. Choudhary, R. Bordawekar, S. More, and S. Kuditipudi, "Passion: Optimized I/O for parallel applications," *IEEE Computer*, vol. 29, pp. 70–78, June 1996.

[327] "Topologically Integrated Geographic Encoding and Referencing system, TIGER/Line 1992 datafiles," Available on the World-Wide Web at http://www.census.gov/geo/www/tiger/, 1992.

[328] S. Toledo, "A survey of out-of-core algorithms in numerical linear algebra," in *External Memory Algorithms and Visualization*, (J. Abello and J. S. Vitter, eds.), pp. 161–179, Providence, Rhode Island: American Mathematical Society Press, 1999.

[329] L. Toma and N. Zeh, "I/O-efficient algorithms for sparse graphs," in *Algorithms for Memory Hierarchies*, (U. Meyer, P. Sanders, and J. Sibeyn, eds.), ch. 5, pp. 85–109, Berlin: Springer-Verlag, 2003.

[330] TPIE User Manual and Reference, "The manual and software distribution," available on the web at http://www.cs.duke.edu/TPIE/, 1999.

[331] J. D. Ullman and M. Yannakakis, "The input/output complexity of transitive closure," *Annals of Mathematics and Artificial Intelligence*, vol. 3, pp. 331–360, 1991.

[332] J. Vahrenhold and K. Hinrichs, "Planar point location for large data sets: To seek or not to seek," *ACM Journal of Experimental Algorithmics*, vol. 7, p. 8, August 2002.

[333] J. van den Bercken, B. Seeger, and P. Widmayer, "A generic approach to bulk loading multidimensional index structures," in *Proceedings of the International Conference on Very Large Databases*, pp. 406–415, 1997.

[334] M. van Kreveld, J. Nievergelt, T. Roos, and P. Widmayer, eds., *Algorithmic foundations of GIS*. Vol. 1340 of *Lecture Notes in Computer Science*, Springer-Verlag, 1997.

[335] P. J. Varman and R. M. Verma, "An efficient multiversion access structure," *IEEE Transactions on Knowledge and Data Engineering*, vol. 9, pp. 391–409, May–June 1997.

[336] D. E. Vengroff and J. S. Vitter, "Efficient 3-D range searching in external memory," in *Proceedings of the ACM Symposium on Theory of Computing*, pp. 192–201, Philadelphia: ACM Press, May 1996.

[337] D. E. Vengroff and J. S. Vitter, "I/O-efficient scientific computation using TPIE," in *Proceedings of NASA Goddard Conference on Mass Storage Systems*, pp. II, 553–570, September 1996.

[338] P. Vettiger, M. Despont, U. Drechsler, U. Dürig, W. Häberle, M. I. Lutwyche, E. Rothuizen, R. Stutz, R. Widmer, and G. K. Binnig, "The 'Millipede' — more than one thousand tips for future AFM data storage," *IBM Journal of Research and Development*, vol. 44, no. 3, pp. 323–340, 2000.

[339] J. S. Vitter, "Efficient memory access in large-scale computation," in *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, pp. 26–41, Springer-Verlag, 1991. Invited paper.

[340] J. S. Vitter, *Notes*. 1999.

[341] J. S. Vitter and P. Flajolet, "Average-case analysis of algorithms and data structures," in *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*, (J. van Leeuwen, ed.), ch. 9, pp. 431–524, Elsevier and MIT Press, 1990.

[342] J. S. Vitter and D. A. Hutchinson, "Distribution sort with randomized cycling," *Journal of the ACM*, vol. 53, pp. 656–680, July 2006.

[343] J. S. Vitter and P. Krishnan, "Optimal prefetching via data compression," *Journal of the ACM*, vol. 43, pp. 771–793, September 1996.

[344] J. S. Vitter and M. H. Nodine, "Large-scale sorting in uniform memory hierarchies," *Journal of Parallel and Distributed Computing*, vol. 17, pp. 107–114, 1993.

[345] J. S. Vitter and E. A. M. Shriver, "Algorithms for parallel memory I: Two-level memories," *Algorithmica*, vol. 12, no. 2–3, pp. 110–147, 1994.

[346] J. S. Vitter and E. A. M. Shriver, "Algorithms for parallel memory II: Hierarchical multilevel memories," *Algorithmica*, vol. 12, no. 2–3, pp. 148–169, 1994.

[347] J. S. Vitter and M. Wang, "Approximate computation of multidimensional aggregates of sparse data using wavelets," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 193–204, Philadelphia: ACM Press, June 1999.

[348] J. S. Vitter, M. Wang, and B. Iyer, "Data cube approximation and histograms via wavelets," in *Proceedings of the International ACM Conference on Information and Knowledge Management*, pp. 96–104, Washington, DC: ACM Press, November 1998.

[349] M. Wang, B. Iyer, and J. S. Vitter, "Scalable mining for classification rules in relational databases," in *Herman Rubin Festschrift*, Hayward, CA: Institute of Mathematical Statistics, Fall 2004.

[350] M. Wang, J. S. Vitter, L. Lim, and S. Padmanabhan, "Wavelet-based cost estimation for spatial queries," in *Proceedings of the International Symposium on Spatial and Temporal Databases*, pp. 175–196, Redondo Beach, Cal.: Springer-Verlag, July 2001.

[351] R. W. Watson and R. A. Coyne, "The parallel I/O architecture of the high-performance storage system (HPSS)," in *Proceedings of the IEEE Symposium on Mass Storage Systems*, pp. 27–44, September 1995.

[352] P. Weiner, "Linear pattern matching algorithm," in *Proceedings of the IEEE Symposium on Switching and Automata Theory*, pp. 1–11, 1973.

[353] K.-Y. Whang and R. Krishnamurthy, "Multilevel grid files — a dynamic hierarchical multidimensional file structure," in *Proceedings of the International Symposium on Database Systems for Advanced Applications*, pp. 449–459, World Scientific Press, 1992.

[354] D. E. Willard and G. S. Lueker, "Adding range restriction capability to dynamic data structures," *Journal of the ACM*, vol. 32, no. 3, pp. 597–617, 1985.

[355] I. H. Witten, A. Moffat, and T. C. Bell, *Managing Gigabytes: Compressing and Indexing Documents and Images*. Los Altos, Cal.: Morgan Kaufmann, 2nd ed., 1999.

[356] O. Wolfson, P. Sistla, B. Xu, J. Zhou, and S. Chamberlain, "DOMINO: Databases for moving objects tracking," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 547–549, Philadelphia: ACM Press, June 1999.

[357] D. Womble, D. Greenberg, S. Wheat, and R. Riesen, "Beyond core: Making parallel computer I/O practical," in *Proceedings of the DAGS Symposium on Parallel Computation*, pp. 56–63, June 1993.

[358] C. Wu and T. Feng, "The universality of the shuffle-exchange network," *IEEE Transactions on Computers*, vol. C-30, pp. 324–332, May 1981.

[359] Y. Xia, S. Prabhakar, S. Lei, R. Cheng, and R. Shah, "Indexing continuously changing data with mean-variance tree," in *Proceedings of the ACM Symposium on Applied Computing*, pp. 52–57, ACM Press, March 2005.

[360] A. C. Yao, "On random 2-3 trees," *Acta Informatica*, vol. 9, pp. 159–170, 1978.

[361] S. B. Zdonik and D. Maier, eds., *Readings in object-oriented database systems*. Morgan Kauffman, 1990.

[362] N. Zeh, *I/O-Efficient Algorithms for Shortest Path Related Problems*. PhD thesis, School of Computer Science, Carleton University, 2002.

[363] W. Zhang and P.-A. Larson, "Dynamic memory adjustment for external mergesort," in *Proceedings of the International Conference on Very Large Databases*, pp. 376–385, 1997.

[364] B. Zhu, "Further computational geometry in secondary memory," in *Proceedings of the International Symposium on Algorithms and Computation*, pp. 514–522, Springer-Verlag, 1994.

[365] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," *IEEE Transactions on Information Theory*, vol. 24, pp. 530–536, September 1978.