List of Corrections

Ph.D. Thesis
Doctor of Philosophy

DTU Compute
Department of Applied Mathematics and Computer Science

An adventure in data structures

Søren Vind

Kongens Lyngby 2015

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet
Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Abstract

Write English Abstract

Resume

Write Danish Resume

Preface

This xxx thesis was prepared at the department of Applied Mathematics and Computer Science at the Technical University of Denmark in fulfillment of the requirements for acquiring a yyy degree in zzz.

The thesis contains a general introduction as well as ......

This dissertation is the result of my research in data structures during my enrollment as a PhD student at DTU Compute.

About my project, collaboration, funding, etc

We vs I

Acknowledgements

- advisors
- external visit
- friends and family
- office
- section
- institute
- university
- coauthors: Philip Bille, Patrick Hagge Cording, Roberto Grossi, Inge Li Gørtz, Markus Jalsenius, Giulia Menconi, Nadia Pisanti, Benjamin Sach, Frederik Rye Skjoldjensen, Roberto Trani, Hjalte Wedel Vildhøj

Contents

CHAPTER 1

Introduction

Data Structures are the basic building blocks in software engineering; they are the method we use to organise and access information in our computers. An *Algorithm* specifies the steps we take in order to perform some task on an input in order to produce a given output, often relying on underlying data structures. Naturally, deep knowledge and understanding of algorithms and data structures is a core competence for software engineers, absolutely vital to developing efficient and predictable software.

In this chapter, I will give a brief primer to the study of data structures and an overview of the important problems in the field and introduce the included papers.

1.1 Bits of Background and Context

The academic field of Computer Science emerged from mathematics and electrical engineering during the latter half of the 20th century. The first large-scale use of computers was during the Second World War, where they were used to break encryption schemes. Ever more complex and large computer installations were built in the following years, primarily for military use, but the machines eventually found a home in industry and universities. The result was that the first academic institutions devoted to studying computer science were created almost exactly fifty years ago (e.g. The Department of Computer Science at Carnegie Mellon University was founded in 1965, as possibly the first such department in the world).

In his pioneering work *"On Computable Numbers, with an Application to the Entscheidungsproblem"* [Tur36] from 1936, Alan Turing in a single stroke became the father of the field of *theoretical computer science*. He gave the first general model of the theoretical capabilities of computers with the *Turing Machine* , and (among others) gave a formalisation of *algorithms*. In the following 80 years, the field expanded in various directions, but the design and analysis of *algorithms* and *data structures* remain at the core.

In the following, we describe the fields of algorithms and data structures as one, and give a very brief introduction and taxonomy of the basic concepts as they are now established. The modern equivalents of the concepts introduced by Turing are as follows:

Machine Model We use an abstract model of a computer that ignores most details and allows us to understand its behaviour and reason about performance. For example, computation in the very common Word RAM model resembles the capabilities of modern day CPUs: memory is modeled as a sequence of words with w bits. We can read or write a word in unit time and perform arithmetic and word-operations on a constant number of words in unit time.

Algorithms In its most general formulation, an algorithm is a method for solving a problem on an input that produces the correct output. The *problem* is a central concept which states the desired properties of the input and output. One classic example of an algorithmic problem is sorting: given as input a list of numbers, return the same numbers in non-decreasing order.

Data Structures A data structure is a method for maintaining a representation of some data while supporting a set of *operations* on the data: allow *updates*, and answer *queries* on the updated data. An example of a data structure problem is sorted list representation: store a list of numbers subject to insertions and deletions, and support queries for the k'th smallest number.

We characterise proposed solutions using several parameters. First, the output of an algorithm or query must always be (one of the possible) *correct* answers. The *performance* of a solution is the amount of resources required to execute it in our machine model, calculated relative to the size n of a finite input: the *space usage* is the number of words (or bits) required, and the *time* is the number of machine operations necessary. We generally consider *worst-case analysis*, meaning the performance obtained when given the input that cause the solution to perform as poorly as possible.

Traditional *deterministic* solutions have inherently bad worst-case performance when solving some problems. To this end, *randomized* solutions relaxes the strict requirements on correctness and worst-case analysis. Instead, *monte carlo* solutions have worst case performance guarantees but some probability of giving an incorrect output. On the contrary, *las vegas* solutions always give a correct output, but with a risk of bad performance.

Generally, research in a particular problem takes two different angles. One is to prove that it is impossible to solve the problem using less resources than some *lower bound* for *any* solution in a given machine model. The opposite direction is to show the existence of a solution where the amount of used resources can be limited by some *upper bound*.

The Classic Example of Sorting It is easy to show a lower bound of $\Omega(n \lg n)$ for sorting n numbers using comparisons and swaps of pairs of numbers: If all n numbers are distinct they have $n!$ possible permutations, only one of which is the correct ordering. Then the number of number-pair comparisons to identify a single permutation uniquely is $\Omega(\lg(n!)) = \Omega(n \lg n)$. There are several matching upper bounds in the form of algorithms solving the problem using $O(n \lg n)$ comparisons[1].

Observe that the sorting lower bound immediately implies a lower bound for any sorted list representation data structure $\mathscr{D}$. The argument is as follows: First insert the entire list of n numbers into $\mathscr{D}$, and then repeatedly get, store, and remove the smallest number in $\mathscr{D}$. Then the result is an ordered version of the original list, created in $O(n)$ operations, meaning that at least one of the operations must take time $\Omega(\lg n)$.

However, the lower bounds does not apply in the Word RAM model where there *is* a linear ($O(n)$) time solution if the numbers each fit in a word. That is, generally lower bounds in weaker models may be circumvented by a stronger machine model.

[1] mergesort, heapsort

1.2 My Particular Branch: What and Why?

Clearly, although I have only touched upon major concepts there are many possible research directions to take. My main interest is in designing data structures from a theoretical perspective in order to obtain new and improved upper bounds compared to previously known results. I have particularly enjoyed considering the simplest of questions, focusing on core data structure primitives and extremely classic problems.

A common theme in my work and a particular interest of mine is to find solutions that require relatively little space. Each of the proposed data structures exhibit this particular feature, though they span several different classic subfields of algorithmic research, namely combinatorial pattern matching, geometry, and compression.

The papers included in the latter chapters of this thesis each contribute at least one non-trivial newly-designed data structure for solving a particular problem. In some instances, the data structure is used as a cornerstone in giving a solution to a particular algorithmic problem, while in other cases the data structure is the singular contribution. All but one of the papers are purely theoretical, meaning that although the proposed solutions may be good in practice, their efficiency have only been established in theory. In a single case, the proposed data structure was implemented and tested experimentally in practical scenarios.

As initially stated, I consider data structures to be some of the most fundamental building blocks for efficiently solving problems. To see why, consider the following example from the origins of the field of combinatorial pattern matching. In 1970, a fair amount of research had gone into finding solutions to the problem of determining the *longest common substring*[2] of two strings of length n, and thus it was conjectured that no linear time solution existed[3]. However, in 1973 Peter Weiner [**weiner1973linear**] disproved the conjecture by giving a linear time algorithm for building the *suffix tree* data structure, which immediately solved the problem (and many others). In the following 40 years, the properties of the data structure and its extensions have been exploited to solve countless problems[4] in combinatorial pattern matching, where it remains a key primitive.

1.3 Background: A Data Structure Primer

- Models (Pointer Machine, Word Ram, Streaming) - Theory vs Practice - Massive Data Sets
- Basics of Data Structures - Trees and Tries - Predecessor
- Strings - Suffix Tree - Fingerprints
- Geometric - Orthogonal Range Searching - With Colors - And Motion
- Compression - DAG compression - SLPs and derivatives

[2] Given two strings of length n, what is the longest substring that is in both strings?

[3] donald knuth

[4] many papers

1.4 An Overview and Outline of Results

The remaining chapters of this dissertation include the following papers, each of them submitted to or published in peer-reviewed conference proceedings. The papers appear in their original form, meaning that notation, language and terminology has not been changed and may not be consistent across chapters. The chapter titles are equal to the original paper titles.

Fingerprints in Compressed Strings. By Philip Bille, Patrick Hagge Cording, Inge Li Gørtz, Benjamin Sach, Hjalte Wedel Vildhøj and Søren Vind. Presented at Algorithms and Data Structures Symposium (WADS), 2013.

Colored Range Searching. By Roberto Grossi and Søren Vind. Presented at Scandinavian Symposium and Workshops on Algorithm Theory (SWAT), 2014.

Indexing Motion Detection Data for Surveillance Video. By Philip Bille, Inge Li Gørtz and Søren Vind. Presented at IEEE International Symposium on Multimedia (ISM), 2014.

Output-Sensitive Pattern Extraction in Sequences. By Roberto Grossi, Giulia Menconi, Nadia Pisanti, Roberto Trani and Søren Vind. Presented at Foundations of Software Technology and Theoretical Computer Science (FSTTCS), 2014.

Compressed Data Structures for Range Searching. By Philip Bille, Inge Li Gørtz and Søren Vind. Presented at International Conference on Language and Automata Theory and Applications (LATA), 2015.

Annotated Data Streams with Multiple Queries. By Markus Jalsenius, Benjamin Sach and Søren Vind. In submission.

Dynamic Relative Compression By Philip Bille, Patrick Hagge Cording, Inge Li Gørtz, Frederik Rye Skjoldjensen, Hjalte Wedel Vildhøj and Søren Vind. In submission.

The journal version of the following paper appeared during my PhD, but as the results were obtained and the conference version published prior to starting the PhD programme the paper is omitted from this dissertation.

String Indexing for Patterns with Wildcards. By Philip Bille, Inge Li Gørtz, Hjalte Wedel Vildhøj and Søren Vind. Presented at Scandinavian Symposium and Workshops on Algorithm Theory (SWAT), 2012. Journal version in Theory of Computing Systems, 2014.

For each paper or subject considered in the later chapters:

- Our Contributions
- Future Directions

Bibliography

[] https://github.com/sorenvind/phd-motiondetectionindex.

[ABHY08] Lars Arge, Mark De Berg, Herman Haverkort, and Ke Yi. “The Priority R-tree: A practically efficient and worst-case optimal R-tree”. In: *ACM TALG* 4.1 (2008), page 9.

[ACP06] Alberto Apostolico, Matteo Comin, and Laxmi Parida. “Bridging lossy and lossless compression by motif pattern discovery”. In: *General Theory of Information Transfer and Combinatorics*. 2006, pages 793–813.

[AFM92] Amihood Amir, Martin Farach, and Yossi Matias. “Efficient randomized dictionary matching algorithms”. In: *Proc. 3rd CPM*. 1992, pages 262–275.

[AG10] Mohamed Ibrahim Abouelhoda and Moustafa Ghanem. “String Mining in Bioinformatics”. In: *Scientific Data Mining and Knowledge Discovery*. 2010, pages 207–247.

[AGM02] Pankaj K Agarwal, Sathish Govindarajan, and S Muthukrishnan. “Range searching in categorical data: Colored range searching on grid”. In: *Proc. 10th ESA*. 2002, pages 17–28.

[AH00] Stephen Alstrup and Jacob Holm. “Improved algorithms for finding level ancestors in dynamic trees”. In: *Proc. 27th ICALP*. 2000, pages 73–84.

[AI06] Alexandr Andoni and Piotr Indyk. “Efficient algorithms for substring near neighbor problem”. In: *Proc. 17th SODA*. 2006, pages 1203–1212.

[AKO04] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. “Replacing suffix trees with enhanced suffix arrays”. In: *JDA* 2.1 (2004), pages 53–86.

[And99] Arne Andersson. “General balanced trees”. In: *J. Algorithms* 30.1 (1999), pages 1–18.

[AT07] Arne Andersson and Mikkel Thorup. “Dynamic ordered sets with exponential search trees”. In: *Journal of the ACM (JACM)* 54.3 (2007), page 13.

[AU07] Hiroki Arimura and Takeaki Uno. “An efficient polynomial space and polynomial delay algorithm for enumeration of maximal motifs in a sequence”. In: *JCO* (2007).

[BÁBNP13] Guillermo de Bernardo, Sandra Álvarez-Garca, Nieves R Brisaboa, Gonzalo Navarro, and Oscar Pedreira. “Compact Querieable Representations of Raster Data”. In: *Proc. 20th SPIRE*. 2013, pages 96–108.

[Bak95] Brenda S Baker. “On finding duplication and near-duplication in large software systems”. In: *Proc. 2nd WCRE*. 1995, pages 86–95.

[BBV12] Djamal Belazzougui, Paolo Boldi, and Sebastiano Vigna. “Predecessor search with distance-sensitive query time”. In: *arXiv:1209.5441* (2012).

[BCKO08] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. 3rd. 2008.

[BCN10] Jérémy Barbay, Francisco Claude, and Gonzalo Navarro. “Compact rich-functional binary relation representations”. In: *Proc. 9th LATIN*. 2010, pages 170–183.

[BDG95] Sergey Brin, James Davis, and Hector Garcia-Molina. “Copy detection mechanisms for digital documents”. In: *Proc. ACM SIGMOD*. Volume 24. 2. 1995, pages 398–409.

[Ben75] Jon Louis Bentley. “Multidimensional binary search trees used for associative searching”. In: *Comm. ACM* 18.9 (1975), pages 509–517.

[Ben79] Jon Louis Bentley. “Multidimensional binary search trees in database applications”. In: *IEEE Trans. Softw. Eng.* 4 (1979), pages 333–340.

[BF04] M.A. Bender and M. Farach-Colton. “The level ancestor problem simplified”. In: *Theoret. Comput. Sci.* 321 (2004), pages 5–12.

[BGSV12] Philip Bille, Inge Li Gørtz, Benjamin Sach, and Hjalte Wedel Vildhøj. “Time-Space Trade-Offs for Longest Common Extensions”. In: *Proc. 23rd CPM*. 2012, pages 293–305.

[BHMM09] Prosenjit Bose, Meng He, Anil Maheshwari, and Pat Morin. “Succinct orthogonal range search structures on a grid with applications to text indexing”. In: *Proc. 11th WADS*. 2009, pages 98–109.

[BKMT95] Panayiotis Bozanis, Nectarios Kitsios, Christos Makris, and Athanasios K Tsakalidis. “New Upper Bounds for Generalized Intersection Searching Problems”. In: *Proc. 22nd ICALP*. 1995, pages 464–474.

[BLN09] Nieves R Brisaboa, Susana Ladra, and Gonzalo Navarro. “K2-trees for compact web graph representation”. In: *Proc. 16th SPIRE*. 2009, pages 18–30.

[BLRSSW11] P. Bille, G.M. Landau, R. Raman, K. Sadakane, S.R. Satti, and O. Weimann. “Random access to grammar-compressed strings”. In: *Proc. 22nd SODA*. 2011, pages 373–389.

[BM72] R Bayer and EM McCreight. “Organization and maintenance of large ordered indexes”. In: *Acta Informatica* 1.3 (1972), pages 173–189.

[BS80] Jon Louis Bentley and James B Saxe. “Decomposable searching problems I. Static-to-dynamic transformation”. In: *J. Algorithms* 1.4 (1980), pages 301–358.

[BV94] O. Berkman and U. Vishkin. “Finding level-ancestors in trees”. In: *J. Comput. System Sci.* 48.2 (1994), pages 214–230.

[CD00] Ross Cutler and Larry S. Davis. “Robust real-time periodic motion detection, analysis, and applications”. In: *IEEE Trans. PAMI* 22.8 (2000), pages 781–796.

[CFLMS04] Xin Chen, Brent Francia, Ming Li, Brian Mckinnon, and Amit Seker. “Shared information and program plagiarism detection”. In: *IEEE Trans Inf Theory* 50.7 (2004), pages 1545–1551.

[CH03] Richard Cole and Ramesh Hariharan. “Faster suffix tree construction with missing suffix links”. In: *SIAM J. Comput.* 33.1 (2003), pages 26–42.

[CHL03] Chia-Hui Chang, Chun-Nan Hsu, and Shao-Cheng Lui. “Automatic information extraction from semi-structured web pages by pattern discovery”. In: *Decis Support Syst* 34.1 (2003), pages 129–147.

[Cla83] Kenneth L Clarkson. “Fast algorithms for the all nearest neighbors problem”. In: *Proc. 24th FOCS*. Volume 83. 1983, pages 226–232.

[CLLPPSS05] Moses Charikar, Eric Lehman, Ding Liu, Rina Panigrahy, Manoj Prabhakaran, Amit Sahai, and Abbi Shelat. “The smallest grammar problem”. In: *IEEE Trans. Inf. Theory* 51.7 (2005), pages 2554–2576.

[CM05] Graham Cormode and S Muthukrishnan. “Substring compression problems”. In: *Proc. 16th SODA*. 2005, pages 321–330.

[CM07] Graham Cormode and S Muthukrishnan. “The string edit distance matching problem with moves”. In: *ACM Trans. Algorithms* 3.1 (2007), page 2.

[CN11] F. Claude and G. Navarro. “Self-indexed grammar-based compression”. In: *Fundamenta Informaticae* 111.3 (2011), pages 313–337.

[Com79] Douglas Comer. “Ubiquitous B-tree”. In: *ACM CSUR* 11.2 (1979), pages 121–137.

[DDW99] Hervé Debar, Marc Dacier, and Andreas Wespi. “Towards a taxonomy of intrusion-detection systems”. In: *Computer Networks* 31.8 (1999), pages 805–822.

[Die91] Paul F. Dietz. “Finding Level-Ancestors in Dynamic Trees”. In: *Proc. 2nd WADS*. 1991, pages 32–40.

[DRSB14] Shan Du, Choudhury A Rahman, Saika Sharmeen, and Wael Badawy. “Event Detection by Spatio-Temporal Indexing of Video Clips.” In: *IJCTE* 6.1 (2014).

[DST80] Peter J Downey, Ravi Sethi, and Robert Endre Tarjan. “Variations on the common subexpression problem”. In: *J. ACM* 27.4 (1980), pages 758–771.

[DSW09] Christian Dick, Jens Schneider, and Ruediger Westermann. “Efficient Geometry Compression for GPU-based Decoding in Realtime Terrain Rendering”. In: *CGF* 28.1 (2009), pages 67–83.

[EGS08] David Eppstein, Michael T Goodrich, and Jonathan Z Sun. “Skip quadtrees: Dynamic data structures for multidimensional point sets”. In: *IJCGA* 18.01n02 (2008), pages 131–160.

[EKZ76] P. van Emde Boas, R. Kaas, and E. Zijlstra. “Design and implementation of an efficient priority queue”. In: *Theory Comput. Syst.* 10.1 (1976), pages 99–127.

[FGN14] Arash Farzan, Travis Gagie, and Gonzalo Navarro. “Entropy-bounded representation of point grids”. In: *CGTA* 47.1 (2014), pages 1–14.

[FP09] Maria Federico and Nadia Pisanti. “Suffix tree characterization of maximal motifs in biological sequences”. In: *Theor. Comput. Sci.* 410.43 (2009), pages 4391–4401.

[FT98] Martin Farach and Mikkel Thorup. “String Matching in Lempel–Ziv Compressed Strings”. In: *Algorithmica* 20.4 (1998), pages 388–404.

[FW93] Michael L. Fredman and Dan E. Willard. “Surpassing the information theoretic bound with fusion trees”. In: *J. Comput. System Sci.* 47.3 (1993), pages 424–436.

[GBBN14] Sandra Alvarez Garcia, Nieves R Brisaboa, Guillermo de Bernardo, and Gonzalo Navarro. “Interleaved K2-Tree: Indexing and Navigating Ternary Relations”. In: *Proc. DCC*. 2014, pages 342–351.

[GG98] Volker Gaede and Oliver Günther. “Multidimensional access methods”. In: *ACM CSUR* 30.2 (1998), pages 170–231.

[GGKNP12] Travis Gagie, Pawe Gawrychowski, Juha Kärkkäinen, Yakov Nekrich, and Simon J. Puglisi. “A faster grammar-based self-index”. In: *arXiv:1209.5441* (2012).

[GJS95] P Gupta, R Janardan, and M Smid. “Further Results on Generalized Intersection Searching Problems: Counting, Reporting, and Dynamization”. In: *J. Algorithms* 19.2 (1995), pages 282–317.

[GJS97] Prosenjit Gupta, Ravi Janardan, and Michiel Smid. “A technique for adding range restrictions to generalized searching problems”. In: *Inform. Process. Lett.* 64.5 (1997), pages 263–269.

[GKNP12] Travis Gagie, Juha Kärkkäinen, Gonzalo Navarro, and Simon J Puglisi. “Colored range queries and document retrieval”. In: *TCS* (2012).

[GKPR96] Leszek Gasieniec, Marek Karpinski, Wojciech Plandowski, and Wojciech Rytter. “Randomized efficient algorithms for compressed strings: The finger-print approach”. In: *Proc. 7th CPM*. 1996, pages 39–49.

[GKPS05] Leszek Gasieniec, Roman Kolpakov, Igor Potapov, and Paul Sant. “Real-time traversal in grammar-based compressed files”. In: *Proc. 15th DCC*. 2005, page 458.

[GPPPUV11] Roberto Grossi, Andrea Pietracaprina, Nadia Pisanti, Geppino Pucci, Eli Upfal, and Fabio Vandin. “MADMX: A strategy for maximal dense motif extraction”. In: *J. Comp. Biol.* 18.4 (2011), pages 535–545.

[GSS98] Nicola Galli, Bernhard Seybold, and Klaus Simon. “Compression of Sparse Matrices: Achieving Almost Minimal Table Size”. In: *Proc. ALEX*. 1998, pages 27–33.

[Gut] Antonin Guttman. “R-trees: A dynamic index structure for spatial searching”. In: *Proc. 1984 ACM SIGMOD*. Volume 14. 2, pages 47–57.

[Hag98] Torben Hagerup. “Sorting and Searching on the Word RAM”. In: *Proc. 15th STACS*. 1998, pages 366–398.

[HT84a] D. Harel and R. E. Tarjan. “Fast algorithms for finding nearest common ancestors”. In: *SIAM J. Comput.* 13.2 (1984), pages 338–355.

[HT84b] D. Harel and R.E. Tarjan. “Fast algorithms for finding nearest common ancestors”. In: *SIAM J. Comput.* 13.2 (1984), pages 338–355.

[HTWM04] Weiming Hu, Tieniu Tan, Liang Wang, and Steve Maybank. “A survey on visual surveillance of object motion and behaviors”. In: *IEEE Trans. SMC* 34.3 (2004), pages 334–352.

[Hua11] Shih-Chia Huang. “An advanced motion detection algorithm with video quality analysis for video surveillance systems”. In: *IEEE Trans. CSVT* 21.1 (2011), pages 1–14.

[HWAVM10] Simon Haegler, Peter Wonka, Stefan Mueller Arisona, Luc Van Gool, and Pascal Mueller. “Grammar-based Encoding of Facades”. In: *CGF* 29.4 (2010), pages 1479–1487.

[JL93] Ravi Janardan and Mario Lopez. “Generalized intersection searching problems”. In: *IJCGA* 3.01 (1993), pages 39–69.

[JMS05] Joseph JáJá, Christian W Mortensen, and Qingmin Shi. “Space-efficient and fast algorithms for multidimensional dominance reporting and counting”. In: *Proc. 15th ISAAC*. 2005, pages 558–568.

[Kal02] Adam Kalai. “Efficient pattern-matching with don’t cares”. In: *Proc. 13th SODA*. 2002, pages 655–656.

[KCW] Wei Chieh Kao, Shih Hsuan Chiu, and Che Y. Wen. “An effective surveillance video retrieval method based upon motion detection”. In: *IEEE ISI 2008* (), pages 261–262.

[KMSTTZ13] Alexis Kaporis, Christos Makris, Spyros Sioutas, Athanasios Tsakalidis, Kostas Tsichlas, and Christos Zaroliagis. “Improved Bounds for Finger Search on a RAM”. In: *Algorithmica* (2013), pages 1–38.

[KO91] Marc J van Kreveld and Mark H Overmars. “Divided k-d trees”. In: *Algorithmica* 6.1-6 (1991), pages 840–858.

[KR87] Richard M Karp and Michael O Rabin. “Efficient randomized pattern-matching algorithms”. In: *IBM J. Res. Dev.* 31.2 (1987), pages 249–260.

[Kre92] Marc van Kreveld. *New results on data structures in computational geometry*. PhD thesis, Department of Computer Science, University of Utrecht, Netherlands, 1992.

[KRSV07] Haim Kaplan, Natan Rubin, Micha Sharir, and Elad Verbin. “Counting colors in boxes”. In: *Proc. 18th SODA*. 2007, pages 785–794.

[KS99] KV Ravi Kanth and Ambuj Singh. “Optimal Dynamic Range Searching in Non-replicating Index Structures”. In: *Proc. 7th ICDT*. 1999, pages 257–276.

[LP01] I. Rigoutsos L. Parida and D. E. Platt. “An Output-Sensitive Flexible Pattern Discovery Algorithm”. In: *Proc. 12th CPM*. 2001, pages 131–142.

[LP12] Kasper Green Larsen and Rasmus Pagh. “I/O-efficient data structures for colored range and prefix reporting”. In: *Proc. 23rd SODA*. 2012, pages 583–592.

[Lue78] George S Lueker. “A data structure for orthogonal range queries”. In: *Proc. 19th FOCS*. 1978, pages 28–34.

[LW13] Kasper Green Larsen and Freek van Walderveen. “Near-Optimal Range Reporting Structures for Categorical Data.” In: *Proc. 24th SODA*. 2013, pages 265–276.

[LW80] DT Lee and CK Wong. “Quintary trees: a file structure for multidimensional datbase sytems”. In: *ACM TODS* 5.3 (1980), pages 339–353.

[McC76] Edward M. McCreight. “A Space-Economical Suffix Tree Construction Algorithm”. In: *Journal of the ACM* 23.2 (April 1976), pages 262–272.

[ME10] Nizar R Mabroukeh and Christie I Ezeife. “A taxonomy of sequential pattern mining algorithms”. In: *ACM CSUR* 43.1 (2010), page 3.

[MN07] Veli Mäkinen and Gonzalo Navarro. “Rank and select revisited and extended”. In: *TCS* 387.3 (2007), pages 332–347.

[MN90] K. Mehlhorn and S. Näher. “Bounded ordered dictionaries in $O(\lg\lg N)$ time and $O(n)$ space”. In: *Inform. Process. Lett.* 35.4 (1990), pages 183–189.

[Mor03] Christian W Mortensen. *Generalized static orthogonal range searching in less space*. Technical report. TR-2003-22, The IT University of Copenhagen, 2003.

[Nek09] Yakov Nekrich. “Orthogonal Range Searching in Linear and Almost-linear Space”. In: *Comput. Geom. Theory Appl.* 42.4 (2009), pages 342–351.

[Nek12] Yakov Nekrich. “Space-efficient range reporting for categorical data”. In: *Proc. 31st PODS*. 2012, pages 113–120.

[Nek14] Yakov Nekrich. “Efficient range searching for categorical and plain data”. In: *ACM TODS* 39.1 (2014), page 9.

[NV13] Yakov Nekrich and Jeffrey Scott Vitter. “Optimal color range reporting in one dimension”. In: *Proc. 21st ESA*. 2013, pages 743–754.

[Ore82] Jack A Orenstein. “Multidimensional tries used for associative searching”. In: *Inform. Process. Lett.* 14.4 (1982), pages 150–157.

[Ove87] Mark H Overmars. *Design of Dynamic Data Structures*. 1987. ISBN: 038712330X.

[PAAV03] Octavian Procopiuc, Pankaj K Agarwal, Lars Arge, and Jeffrey Scott Vitter. “Bkd-tree: A dynamic scalable kd-tree”. In: *Proc. 8th SSTD*. 2003, pages 46–65.

[Pat07] Mihai Patrascu. “Lower bounds for 2-dimensional range counting”. In: *Proc. 39th STOC*. 2007, pages 40–46.

[PP09] Benny Porat and Ely Porat. “Exact and approximate pattern matching in the streaming model”. In: *Proc. 50th FOCS*. 2009, pages 315–323.

[PRFPG00] Laxmi Parida, Isidore Rigoutsos, Aris Floratos, Daniel E. Platt, and Yuan Gao. “Pattern discovery on character sets and real-valued data: linear bound on irredundant motifs and an efficient polynomial time algorithm”. In: *Proc. 11th SODA*. 2000, pages 297–308.

[PW00] Renato Pajarola and Peter Widmayer. “An image compression method for spatial search”. In: *IEEE Trans. Image Processing* 9.3 (2000), pages 357–365.

[PYK06] Luká Pichl, Takuya Yamano, and Taisei Kaizoji. “On the symbolic analysis of market indicators with the dynamic programming approach”. In: *Advances in Neural Networks-ISNN*. 2006, pages 432–441.

[RH04] Isidore Rigoutsos and Tien Huynh. “Chung-Kwei: a Pattern-discovery-based System for the Automatic Identification of Unsolicited E-mail Messages”. In: *CEAS*. 2004.

[Rob] John T Robinson. “The KDB-tree: a search structure for large multidimensional dynamic indexes”. In: *Proc. 1981 ACM SIGMOD*, pages 10–18.

[Ryt03] Wojciech Rytter. “Application of Lempel–Ziv factorization to the approximation of grammar-based compression”. In: *Theoret. Comput. Sci.* 302.1 (2003), pages 211–222.

[Sag98] Marie-France Sagot. “Spelling approximate repeated or common motifs using a suffix tree”. In: *Proc. 3rd LATIN*. 1998, pages 374–390.

[Sam90] Hanan Samet. *Applications of spatial data structures*. Addison-Wesley, 1990.

[SJ05] Qingmin Shi and Joseph JáJá. "Optimal and near-optimal algorithms for generalized intersection reporting on pointer machines". In: *Inform. Process. Lett.* 95.3 (2005), pages 382–388.

[SKLLD08] Grant Schindler, Panchapagesan Krishnamurthy, Roberto Lublinerman, Yanxi Liu, and Frank Dellaert. "Detecting and matching repeated patterns for automatic geo-tagging in urban environments". In: *CVPR*. 2008, pages 1–7.

[SMHKNM94] Todd S Sachs, Craig H Meyer, Bob S Hu, Jim Kohli, Dwight G Nishimura, and Albert Macovski. "Real-time motion detection in spiral MRI using navigators". In: *Magnetic resonance in medicine* 32.5 (1994), pages 639–645.

[SR06] Reza Sherkat and Davood Rafiei. "Efficiently evaluating order preserving similarity queries over historical market-basket data". In: *Proc. 22nd ICDE*. 2006, pages 19–19.

[TH] Ying-Li Tian and Arun Hampapur. "Robust salient motion detection with complex background for real-time video surveillance". In: *WACV 2005*.

[Tur36] Alan Mathison Turing. "On computable numbers, with an application to the Entscheidungsproblem". In: *J. of Math* 58 (1936), pages 345–363.

[TV01] Igor V Tetko and Alessandro EP Villa. "A pattern grouping algorithm for analysis of spatiotemporal patterns in neuronal spike trains." In: *J. Neurosci. Meth.* 105.1 (2001), pages 1–14.

[Ukk09] Esko Ukkonen. "Maximal and minimal representations of gapped and non-gapped motifs of a string". In: *Theor. Comput. Sci.* 410.43 (2009), pages 4341–4349.

[Wil12] Virginia Vassilevska Williams. "Multiplying matrices faster than Coppersmith-Winograd". In: *Proc. 44th STOC*. 2012, pages 887–898.

[Wil83] D.E. Willard. "Log-logarithmic worst-case range queries are possible in space Θ(N)". In: *Inform. Process. Lett.* 17.2 (1983), pages 81–84.

[ZL77] Jacob Ziv and Abraham Lempel. "A universal algorithm for sequential data compression". In: *Information Theory, IEEE Trans. Inf. Theory* 23.3 (1977), pages 337–343.

[ZL78] Jacob Ziv and Abraham Lempel. "Compression of individual sequences via variable-rate coding". In: *Information Theory, IEEE Trans. Inf. Theory* 24.5 (1978), pages 530–536.

[ZYHZ02] Qing Zhu, Xuefeng Yao, Duo Huang, and Yetin Zhang. "An Efficient Data Management Approach for Large Cyber-City GIS". In: *ISPRS Archives* (2002), pages 319–323.