
References

- [1] E. H. L. Aarts and J. Korst. *Simulated Annealing and Boltzmann Machines*. John Wiley & Sons Ltd., 1989.
- [2] J. Abello, A. L. Buchsbaum, and J. R. Westbrook. A functional approach to external graph algorithms. *Algorithmica*, 32(3):437–458, 2002.
- [3] I. Abraham, D. Delling, A. V. Goldberg, and R. F. F. Werneck. A hub-based labeling algorithm for shortest paths in road networks. In *10th Symposium on Experimental Algorithms (SEA)*, volume 6630 of *LNCS*, pages 230–241. Springer, 2011.
- [4] W. Ackermann. Zum hilbertschen Aufbau der reellen Zahlen. *Mathematische Annalen*, 99:118–133, 1928.
- [5] G. M. Adel'son-Vel'skii and E. M. Landis. An algorithm for the organization of information. *Soviet Mathematics Doklady*, 3:1259–1263, 1962.
- [6] A. Aggarwal and J. S. Vitter. The input/output complexity of sorting and related problems. *Communications of the ACM*, 31(9):1116–1127, 1988.
- [7] A. V. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.
- [8] A. V. Aho, B. W. Kernighan, and P. J. Weinberger. *The AWK Programming Language*. Addison-Wesley, 1988.
- [9] R. K. Ahuja, R. L. Magnanti, and J. B. Orlin. *Network Flows*. Prentice Hall, 1993.
- [10] R. K. Ahuja, K. Mehlhorn, J. B. Orlin, and R. E. Tarjan. Faster algorithms for the shortest path problem. *Journal of the ACM*, 3(2):213–223, 1990.
- [11] M. Ajtai, J. Komlós, and E. Szemerédi. An $O(n \log n)$ sorting network. In *15th ACM Symposium on Theory of Computing (STOC)*, pages 1–9, 1983.
- [12] Y. Akhremtsev and P. Sanders. Fast parallel operations on search trees. In *23rd IEEE Conference on High Performance Computing (HIPC)*, pages 291–300, 2016.
- [13] M. Akra and L. Bazzi. On the solution of linear recurrence equations. *Computational Optimization and Applications*, 10(2):195–210, 1998.
- [14] E. Alba and M. Tomassini. Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6(5):443–462, 2002.

- [15] N. Alon, M. Dietzfelbinger, P. B. Miltersen, E. Petrank, and E. Tardos. Linear hash functions. *Journal of the ACM*, 46(5):667–683, 1999.
- [16] A. Andersson, T. Hagerup, S. Nilsson, and R. Raman. Sorting in linear time? *Journal of Computer and System Sciences*, pages 74–93, 1998.
- [17] F. Annexstein, M. Baumslag, and A. Rosenberg. Group action graphs and parallel architectures. *SIAM Journal on Computing*, 19(3):544–569, 1990.
- [18] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, 2007.
- [19] L. Arge, M. T. Goodrich, M. Nelson, and N. Sitchinava. Fundamental parallel algorithms for private-cache chip multiprocessors. In *20th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 197–206, 2008.
- [20] N. S. Arora, R. D. Blumofe, and C. G. Plaxton. Thread scheduling for multiprogrammed multiprocessors. In *10th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 119–129, 1998.
- [21] J. Arz, D. Luxen, and P. Sanders. Transit node routing reconsidered. In *12th Symposium on Experimental Algorithms (SEA)*, volume 7933 of *LNCS*, pages 55–66. Springer, 2013.
- [22] G. Ausiello, P. Crescenzi, G. Gambosi, V. Kann, A. Marchetti-Spaccamela, and M. Protasi. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer, 1999.
- [23] M. Axtmann, T. Bingmann, P. Sanders, and C. Schulz. Practical massively parallel sorting. In *27th ACM Symposium on Parallelism in Algorithms and Architectures, (SPAA)*, pages 13–23, 2015.
- [24] M. Axtmann, A. Wiebigke, and P. Sanders. Lightweight MPI communicators with applications to perfectly balanced quicksort. In *32nd IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 254–265, 2018.
- [25] M. Axtmann, S. Witt, D. Ferizovic, and P. Sanders. In-place parallel super scalar samplesort (IPSSSSo). In *25th European Symposium on Algorithms (ESA)*, pages 9:1–9:14, 2017. full paper at arXiv:1705.02257 [cs.DC].
- [26] D. A. Bader and G. Cong. Fast shared-memory algorithms for computing the minimum spanning forest of sparse graphs. *Journal of Parallel and Distributed Computing*, 66:1366–1378, 2006.
- [27] M. Bader. *Space-filling Curves – An Introduction with Applications in Scientific Computing*, volume 9 of *Texts in Computational Science and Engineering*. Springer, 2012.
- [28] T. Balyo and P. Sanders. HordeSat: A massively parallel portfolio SAT solver. In *18th Conference on Theory and Applications of Satisfiability Testing (SAT)*, volume 9340 of *LNCS*, pages 156–172. Springer, 2015.
- [29] H. Bast. Scheduling at twilight the easy way. In *19th Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 2285 of *LNCS*, pages 166–178. Springer, 2002.
- [30] H. Bast, D. Delling, A. Goldberg, M. Müller-Hannemann, T. Pajor, P. Sanders, D. Wagner, and R. F. Werneck. Route planning in transportation networks. In

- L. Kliemann and P. Sanders, editors, *Algorithm Engineering*, volume 9220 of *LNCS*, pages 19–80. Springer, 2016.
- [31] H. Bast, S. Funke, P. Sanders, and D. Schultes. Fast routing in road networks with transit nodes. *Science*, 316(5824):566, 2007.
- [32] H. Bast and T. Hagerup. Fast parallel space allocation, estimation and integer sorting. *Information and Computation*, 123(1):72–110, 1995.
- [33] K. E. Batchler. Sorting networks and their applications. In *AFIPS Spring Joint Computing Conference*, pages 307–314, 1968.
- [34] R. Bayer and E. M. McCreight. Organization and maintenance of large ordered indexes. *Acta Informatica*, 1(3):173–189, 1972.
- [35] A. Beckmann, R. Dementiev, and J. Singler. Building a parallel pipelined external memory algorithm library. In *23rd IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, 2009.
- [36] R. Beier and B. Vöcking. Random knapsack in expected polynomial time. *Journal of Computer and System Sciences*, 69(3):306–329, 2004.
- [37] D. Belazzougui, F. C. Botelho, and M. Dietzfelbinger. Hash, displace, and compress. In *17th European Symposium on Algorithms (ESA)*, volume 5757 of *LNCS*, pages 682–693. Springer, 2009.
- [38] R. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.
- [39] M. A. Bender, E. D. Demaine, and M. Farach-Colton. Cache-oblivious B-trees. In *41st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 399–409, 2000.
- [40] M. A. Bender, M. Farach-Colton, G. Pemmasani, S. Skiena, and P. Sumazin. Lowest common ancestors in trees and directed acyclic graphs. *Journal of Algorithms*, 57(2):75–94, 2005.
- [41] J. L. Bentley and M. D. McIlroy. Engineering a sort function. *Software Practice and Experience*, 23(11):1249–1265, 1993.
- [42] J. L. Bentley and T. A. Ottmann. Algorithms for reporting and counting geometric intersections. *IEEE Transactions on Computers*, C-28(9):643–647, 1979.
- [43] J. L. Bentley and R. Sedgewick. Fast algorithms for sorting and searching strings. In *8th ACM Symposium on Discrete Algorithms (SODA)*, pages 360–369, 1997.
- [44] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, 1997.
- [45] T. Bingmann, T. Keh, and P. Sanders. A bulk-parallel priority queue in external memory with STXXL. In *14th Symposium on Experimental Algorithms (SEA)*, volume 9125 of *LNCS*, pages 28–40. Springer, 2015.
- [46] G. E. Blelloch, D. Ferizovic, and Y. Sun. Just join for parallel ordered sets. In *28th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 253–264, 2016.
- [47] G. E. Blelloch, C. E. Leiserson, B. M. Maggs, C. G. Plaxton, S. J. Smith, and M. Zagha. A comparison of sorting algorithms for the connection machine

- CM-2. In *3rd ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 3–16, 1991.
- [48] M. Blum, R. W. Floyd, V. R. Pratt, R. L. Rivest, and R. E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448, 1972.
- [49] N. Blum and K. Mehlhorn. On the average number of rebalancing operations in weight-balanced trees. *Theoretical Computer Science*, 11:303–320, 1980.
- [50] Boost.org. Boost C++ Libraries. www.boost.org.
- [51] O. Borůvka. O jistém problému minimálním. *Práce Moravské Přírodovědecké Společnosti*, 3:37–58, 1926. In Czech.
- [52] F. C. Botelho, R. Pagh, and N. Ziviani. Practical perfect hashing in nearly optimal space. *Information Systems*, 38(1):108–131, 2013.
- [53] G. S. Brodal. Worst-case efficient priority queues. In *7th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 52–58, 1996.
- [54] G. S. Brodal and J. Katajainen. Worst-case efficient external-memory priority queues. In *6th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1432 of *LNCS*, pages 107–118. Springer, 1998.
- [55] G. S. Brodal, J. L. Träff, and C. D. Zaroliagis. A parallel priority queue with constant time operations. *Journal of Parallel and Distributed Computing*, 49(1):4–21, 1998.
- [56] N. G. Bronson, J. Casper, H. Chafi, and K. Olukotun. A practical concurrent binary search tree. *ACM SIGPLAN Notices*, 45(5):257–268, 2010.
- [57] M. R. Brown and R. E. Tarjan. Design and analysis of a data structure for representing sorted lists. *SIAM Journal of Computing*, 9:594–614, 1980.
- [58] R. Brown. Calendar queues: A fast $O(1)$ priority queue implementation for the simulation event set problem. *Communications of the ACM*, 31(10):1220–1227, 1988.
- [59] A. Buluc, H. Meyerhenke, I. Safro, P. Sanders, and C. Schulz. Recent advances in graph partitioning. In L. Kliemann and P. Sanders, editors, *Algorithm Engineering*, volume 9220 of *LNCS*, pages 117–158. Springer, 2014.
- [60] C. K. Caldwell and Y. Cheng. Determining Mills’ constant and a note on Honaker’s problem. *Journal Integer Sequences*, 8(4):Article 05.4.1, 2005.
- [61] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.
- [62] S. K. Cha, S. Hwang, K. Kim, and K. Kwon. Cache-conscious concurrency control of main-memory indexes on shared-memory multiprocessor systems. In *27th Conference on Very Large Data Bases (VLDB)*, pages 181–190, 2001.
- [63] D. Chakrabarti, Y. Zhan, and C. Faloutsos. R-MAT: A recursive model for graph mining. In *SIAM International Conference on Data Mining (SDM)*, pages 442–446, 2004.
- [64] R. Chandra. *Parallel Programming in OpenMP*. Morgan Kaufmann, 2001.
- [65] Chazelle. A minimum spanning tree algorithm with inverse-Ackermann type complexity. *Journal of the ACM*, 47:1028–1047, 2000.
- [66] B. Chazelle and L. J. Guibas. Fractional cascading: I. A data structuring technique. *Algorithmica*, 1(2):133–162, 1986.

- [67] B. Chazelle and L. J. Guibas. Fractional cascading: II. Applications. *Algorithmica*, 1(2):163–191, 1986.
- [68] J.-C. Chen. Proportion extend sort. *SIAM Journal on Computing*, 31(1):323–330, 2001.
- [69] Y. Cheng. Explicit estimate on primes between consecutive cubes. *Rocky Mountain J. Math.*, 40:1, 117–153, 2010.
- [70] J. Cheriyan and K. Mehlhorn. Algorithms for dense graphs and networks. *Algorithmica*, 15(6):521–549, 1996.
- [71] B. V. Cherkassky, A. V. Goldberg, and T. Radzik. Shortest path algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73:129–174, 1996.
- [72] Y.-J. Chiang, M. T. Goodrich, E. F. Grove, R. Tamassia, D. E. Vengroff, and J. S. Vitter. External-memory graph algorithms. In *6th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 139–149, 1995.
- [73] E. G. Coffman, M. R. G. Jr., and D. S. Johnson. Approximation algorithms for bin packing: A survey. In D. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 46–93. PWS, 1997.
- [74] D. Cohen-Or, D. Levin, and O. Remez. Progressive compression of arbitrary triangular meshes. In *IEEE Conference on Visualization (VIS)*, pages 67–72, 1999.
- [75] R. Cole. Parallel merge sort. *SIAM Journal on Computing*, 17(4):770–785, 1988.
- [76] R. Cole, P. N. Klein, and R. E. Tarjan. Finding minimum spanning forests in logarithmic time and linear work using random sampling. In *8th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 243–250, 1996.
- [77] R. Cole, K. Ost, and S. Schirra. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica*, 21(1):5–12, 2000.
- [78] S. A. Cook. *On the Minimum Computation Time of Functions*. PhD thesis, Harvard University, 1966.
- [79] S. A. Cook. The complexity of theorem proving procedures. In *3rd ACM Symposium on Theory of Computing*, pages 151–158, 1971.
- [80] A. Crauser, K. Mehlhorn, U. Meyer, and P. Sanders. A parallelization of Dijkstra’s shortest path algorithm. In *23rd Symposium on Mathematical Foundations of Computer Science (MFCS)*, number 1450 in LNCS, pages 722–731. Springer, 1998.
- [81] G. B. Dantzig. Maximization of a linear function of variables subject to linear inequalities. In T. C. Koopmans, editor, *Activity Analysis of Production and Allocation*, pages 339–347. Wiley, 1951.
- [82] C. Darwin. *The Origin of Species by Means of Natural Selection: or, the Preservation of Favored Races in the Struggle for Life*. John Murray, London, 1859.
- [83] A. Davidson, D. Tarjan, M. Garland, and J. D. Owens. Efficient parallel merge sort for fixed and variable length keys. In *Innovative Parallel Computing (InPar), 2012*, pages 1–9. IEEE, 2012.

- [84] A. De, P. P. Kurur, C. Saha, and R. Satharishi. Fast integer multiplication using modular arithmetic. *SIAM Journal on Computing*, 42(2):685–699, 2013.
- [85] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry Algorithms and Applications*. Springer, 2nd edition, 2000.
- [86] J. Dean and S. Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, January 2008.
- [87] R. Dementiev, L. Kettner, J. Mehnert, and P. Sanders. Engineering a sorted list data structure for 32 bit keys. In *6th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 142–151, 2004.
- [88] R. Dementiev, L. Kettner, and P. Sanders. STXXL: standard template library for XXL data sets. *Softw., Pract. Exper.*, 38(6):589–637, 2008.
- [89] R. Dementiev and P. Sanders. Asynchronous parallel disk sorting. In *15th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 138–148, 2003.
- [90] R. Dementiev, P. Sanders, D. Schultes, and J. Sibeyn. Engineering an external memory minimum spanning tree algorithm. In *Exploring New Frontiers of Theoretical Informatics*, volume 155 of *IFIPAICT*, pages 195–208. Springer, 2004.
- [91] L. Devroye. A note on the height of binary search trees. *Journal of the ACM*, 33(3):289–498, 1986.
- [92] R. B. Dial. Shortest-path forest with topological ordering. *Communications of the ACM*, 12(11):632–633, 1969.
- [93] M. Dietzfelbinger, T. Hagerup, J. Katajainen, and M. Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1):19–51, 1997.
- [94] M. Dietzfelbinger, A. Karlin, K. Mehlhorn, F. Meyer auf der Heide, H. Rohnert, and R. E. Tarjan. Dynamic perfect hashing: Upper and lower bounds. *SIAM Journal of Computing*, 23(4):738–761, 1994.
- [95] M. Dietzfelbinger and C. Weidling. Balanced allocation and dictionaries with tightly packed constant size bins. *Theoretical Computer Science*, 380(1–2):47–68, 2007.
- [96] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.
- [97] E. A. Dinic. Economical algorithms for finding shortest paths in a network. *Transportation Modeling Systems*, pages 36–44, 1978. In Russian.
- [98] W. Domschke and A. Drexl. *Einführung in Operations Research*. Springer, 2007.
- [99] J. R. Driscoll, N. Sarnak, D. D. Sleator, and R. E. Tarjan. Making data structures persistent. *Journal of Computer and System Sciences*, 38(1):86–124, 1989.
- [100] M. Drmota and W. Szpankowski. A master theorem for discrete divide and conquer recurrences. *Journal of the ACM*, 60(3):16:1–16:49, 2013.
- [101] S. Edelkamp and A. Weiss. BlockQuicksort: Avoiding branch mispredictions in quicksort. In *24th European Symposium on Algorithms (ESA)*, volume 57 of *LIPICs*, pages 38:1–38:16, 2016.

- [102] A. Elmasry, J. Katajainen, and M. Stenmark. Branch mispredictions don't affect mergesort. In *11th Symposium on Experimental Algorithms (SEA)*, volume 7276 of *LNCS*, pages 160–171. Springer, 2012.
- [103] J. Fakcharoenphol and S. Rao. Planar graphs, negative weight edges, shortest paths, and near linear time. *Journal of Computer and System Sciences*, 72(5):868–889, 2006.
- [104] L. K. Fleischer, B. Hendrickson, and A. Pinar. On identifying strongly connected components in parallel. In *Workshop on Solving Irregularly Structured Problems in Parallel (IPDPS Workshops)*, number 1800 in *LNCS*, pages 505–511. Springer, 2000.
- [105] R. Fleischer. A tight lower bound for the worst case of Bottom-Up-Heapsort. *Algorithmica*, 11(2):104–115, 1994.
- [106] R. Floyd. Assigning meaning to programs. In J. Schwarz, editor, *Mathematical Aspects of Computer Science*, pages 19–32. AMS, 1967.
- [107] R. W. Floyd and R. L. Rivest. Expected time bounds for selection. *Communications of the ACM*, 18(3):165–172, 1975.
- [108] L. R. Ford. Network flow theory. Technical Report P-923, Rand Corporation, Santa Monica, California, 1956.
- [109] D. Fotakis, R. Pagh, P. Sanders, and P. Spirakis. Space efficient hash tables with worst case constant access time. *Theory of Computing Systems*, 38(2):229–248, 2005.
- [110] W. D. Frazer and A. C. McKellar. Samplesort: A sampling approach to minimal storage tree sorting. *Journal of the ACM*, 17(3):496–507, 1970.
- [111] E. Fredkin. Trie memory. *Communications of the ACM*, 3(9):490–499, 1960.
- [112] M. L. Fredman. On the efficiency of pairing heaps and related data structures. *Journal of the ACM*, 46(4):473–501, 1999.
- [113] M. L. Fredman, J. Komlós, and E. Szemerédi. Storing a sparse table with $O(1)$ worst case access time. *Journal of the ACM*, 31(3):538–544, 1984.
- [114] M. L. Fredman, R. Sedgwick, D. D. Sleator, and R. E. Tarjan. The pairing heap: A new form of self-adjusting heap. *Algorithmica*, 1:111–129, 1986.
- [115] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.
- [116] M. Frigo, C. E. Leiserson, H. Prokop, and S. Ramachandran. Cache-oblivious algorithms. In *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 285–298, 1999.
- [117] M. Fürer. Faster integer multiplication. *SIAM Journal on Computing*, 39(3):979–1005, 2009.
- [118] H. N. Gabow. Path-based depth-first search for strong and biconnected components. *Information Processing Letters*, 74(3–4):107–114, 2000.
- [119] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.
- [120] D. Gangal and A. Ranade. Precedence constrained scheduling in $(2 - 7/(3p - 1))$ optimal. *Journal of Computer and System Sciences*, 74(7):1139–1146, 2008.

- [121] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [122] B. Gärtner and J. Matoušek. *Understanding and Using Linear Programming*. Springer, 2006.
- [123] R. Geisberger, P. Sanders, D. Schultes, and C. Vetter. Exact routing in large road networks using contraction hierarchies. *Transportation Science*, 46(3):388–404, 2012.
- [124] J. Giacomoni, T. Moseley, and M. Vachharajani. Fastforward for efficient pipeline parallelism: A cache-optimized concurrent lock-free queue. In *13th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pages 43–52, 2008.
- [125] P. B. Gibbons. A more practical pram model. In *1st ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 158–168, 1989.
- [126] P. B. Gibbons, Y. Matias, and V. Ramachandran. The QRQW PRAM: Accounting for contention in parallel algorithms. In *5th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 638–648, 1994.
- [127] GMP (GNU Multiple Precision Arithmetic Library). <http://gmplib.org/>.
- [128] A. V. Goldberg. Scaling algorithms for the shortest path problem. *SIAM Journal on Computing*, 24(3):494–504, 1995.
- [129] A. V. Goldberg. A simple shortest path algorithm with linear average time. In *9th European Symposium on Algorithms (ESA)*, number 2161 in LNCS, pages 230–241. Springer, 2001.
- [130] A. V. Goldberg and C. Harrelson. Computing the shortest path: A* meets graph theory. In *16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 156–165, 2005.
- [131] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures: In Pascal and C*. Addison-Wesley, 2nd edition, 1991.
- [132] M. T. Goodrich. Randomized Shellsort: A simple data-oblivious sorting algorithm. *Journal of the ACM*, 58(6):27:1–27:26, 2011.
- [133] M. T. Goodrich. Zig-zag sort: A simple deterministic data-oblivious sorting algorithm running in $O(n \log n)$ time. In *46th ACM Symposium on Theory of Computing (STOC)*, pages 684–693, 2014.
- [134] G. Graefe. A survey of B-tree locking techniques. *ACM Transactions on Database Systems (TODS)*, 35(3):16, 2010.
- [135] G. Graefe and P.-A. Larson. B-tree indexes and CPU caches. In *17th International Conference on Data Engineering (ICDE)*, pages 349–358. IEEE, 2001.
- [136] R. L. Graham. Bounds on multiprocessing timing anomalies. *SIAM Journal of Applied Mathematics*, 17(2):416–429, 1969.
- [137] R. L. Graham, D. E. Knuth, and O. Patashnik. *Concrete mathematics*. Addison-Wesley, 2nd edition, 1994.
- [138] J. F. Grantham and C. Pomerance. Prime numbers. In K. H. Rosen, editor, *Handbook of Discrete and Combinatorial Mathematics*, chapter 4.4, pages 236–254. CRC Press, 2000.

- [139] R. Grossi and G. Italiano. Efficient techniques for maintaining multi-dimensional keys in linked data structures. In *26th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 1644 of *LNCS*, pages 372–381. Springer, 1999.
- [140] B. Haeupler, S. Sen, and R. E. Tarjan. Rank-pairing heaps. In *17th European Symposium on Algorithms (ESA)*, volume 5757 of *LNCS*, pages 659–670, 2009.
- [141] S. Halperin and U. Zwick. Optimal randomized EREW PRAM algorithms for finding spanning forests and for other basic graph connectivity problems. In *7th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 438–447, 1996.
- [142] Y. Han and M. Thorup. Integer sorting in $O(n\sqrt{\log \log n})$ expected time and linear space. In *42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 135–144, 2002.
- [143] G. Handler and I. Zang. A dual algorithm for the constrained shortest path problem. *Networks*, 10(4):293–309, 1980.
- [144] T. D. Hansen, H. Kaplan, R. E. Tarjan, and U. Zwick. Hollow heaps. *ACM Transactions on Algorithms (TALG)*, 13(3):42:1–42:27, 2017.
- [145] J. Hartmanis and J. Simon. On the power of multiplication in random access machines. In *5th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 13–23, 1974.
- [146] M. Held and R. Karp. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6):1138–1162, 1970.
- [147] M. Held and R. Karp. The traveling-salesman problem and minimum spanning trees, part II. *Mathematical Programming*, 1:6–25, 1971.
- [148] J. L. Hennessy and D. A. Patterson. *Computer Architecture a Quantitative Approach*. Morgan Kaufmann, 5th edition, 2011.
- [149] P. V. Hentenryck and L. Michel. *Constraint-Based Local Search*. MIT Press, 2005.
- [150] M. Herlihy and N. Shavit. *The Art of Multiprocessor Programming, Revised Reprint*. Elsevier, 2012.
- [151] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–585, 1969.
- [152] C. A. R. Hoare. Proof of correctness of data representations. *Acta Informatica*, 1(4):271–281, 1972.
- [153] R. D. Hofstadter. Metamagical themas. *Scientific American*, 248(2):16–22, 1983.
- [154] S. Hong, N. C. Rodia, and K. Olukotun. On fast parallel detection of strongly connected components (SCC) in small-world graphs. In *ACM Conference on High Performance Computing, Networking, Storage and Analysis (SC)*, pages 92:1–92:11, 2013.
- [155] J. E. Hopcroft and J. D. Ullman. Set merging algorithms. *SIAM Journal on Computing*, 2(4):294–303, 1973.

- [156] P. Høyer. A general technique for implementation of efficient priority queues. In *3rd Israeli Symposium on Theory of Computing and Systems*, pages 57–66, 1995.
- [157] L. Hübschle-Schneider and P. Sanders. Communication efficient algorithms for top- k selection problems. In *30th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 659–668, 2016.
- [158] S. Huddleston and K. Mehlhorn. A new data structure for representing sorted lists. *Acta Informatica*, 17(2):157–184, 1982.
- [159] J. Iacono. Improved upper bounds for pairing heaps. In *7th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 1851 of *LNCS*, pages 32–45. Springer, 2000.
- [160] ©Intel. *Intel® 64 and IA-32 Architectures Optimization Reference Manual*, 2016. Order No. 248966-032.
- [161] ISO/IEC. C++ programming language standard. Technical Report 14882:2014, ISO/IEC, 2014.
- [162] A. Itai, A. G. Konheim, and M. Rodeh. A sparse table implementation of priority queues. In *8th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 115 of *LNCS*, pages 417–431. Springer, 1981.
- [163] J. Jájá. *An Introduction to Parallel Algorithms*. Addison Wesley, 1992.
- [164] V. Jarník. O jistém problému minimálním (Z dopisu panu O. Borůvkovi). *Práce Moravské Přírodovědecké Společnosti*, 6:57–63, 1930. In Czech.
- [165] K. Jensen and N. Wirth. *Pascal User Manual and Report. ISO Pascal Standard*. Springer, 1991.
- [166] jgrapht.org. JGraphT Java Graph Library. <http://jgrapht.org>.
- [167] T. Jiang, M. Li, and P. Vitányi. Average-case complexity of Shellsort. In *26th International Colloquium on Automata, Languages and Programming (ICALP)*, number 1644 in *LNCS*, pages 453–462. Springer, 1999.
- [168] D. S. Johnson, C. R. Aragon, L. A. McGeoch, and C. Schevon. Optimization by simulated annealing: Experimental evaluation; part II, graph coloring and number partitioning. *Operations Research*, 39(3):378–406, 1991.
- [169] N. L. Johnson, S. Kotz, and A. W. Kemp. *Univariate Discrete Distributions*. Wiley, 1989.
- [170] S. L. Johnsson and C. T. Ho. Optimum broadcasting and personalized communication in hypercubes. *IEEE Transactions on Computers*, 38(9):1249–1268, 1989.
- [171] L. V. Kalé and A. B. Sinha. Information sharing mechanisms in parallel programs. In *8th International Parallel Processing Symposium (IPPS)*, pages 461–468. IEEE, 1994.
- [172] K. Kaligosi and P. Sanders. How branch mispredictions affect quicksort. In *14th European Symposium on Algorithms (ESA)*, volume 4168 of *LNCS*, pages 780–791. Springer, 2006.
- [173] H. Kaplan and R. E. Tarjan. New heap data structures. Technical Report TR-597-99, Princeton University, 1999.

- [174] A. Karatsuba and Y. Ofman. Multiplication of multidigit numbers on automata. *Soviet Physics Doklady*, 7(7):595–596, 1963.
- [175] D. Karger, P. N. Klein, and R. E. Tarjan. A randomized linear-time algorithm for finding minimum spanning trees. *Journal of the ACM*, 42(1):321–329, 1995.
- [176] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.
- [177] J. Katajainen and B. B. Mortensen. Experiences with the design and implementation of space-efficient deques. In *5th Workshop on Algorithm Engineering (WAE)*, volume 2141 of *LNCS*, pages 39–50. Springer, 2001.
- [178] I. Katriel, P. Sanders, and J. L. Träff. A practical minimum spanning tree algorithm using the cycle property. In *11th European Symposium on Algorithms (ESA)*, number 2832 in *LNCS*, pages 679–690. Springer, 2003.
- [179] H. Kellerer, U. Pferschy, and D. Pisinger. *Knapsack problems*. Springer, 2004.
- [180] L. Khachiyan. A polynomial time algorithm in linear programming (in Russian). *Soviet Mathematics Doklady*, 20(1):191–194, 1979.
- [181] T. Kieritz, D. Luxen, P. Sanders, and C. Vetter. Distributed time-dependent contraction hierarchies. In *9th Symposium on Experimental Algorithms (SEA)*, volume 6049 of *LNCS*, pages 83–93. Springer, 2010.
- [182] V. King. A simpler minimum spanning tree verification algorithm. *Algorithmica*, 18(2):263–270, 1997.
- [183] S. Knopp, P. Sanders, D. Schultes, F. Schulz, and D. Wagner. Computing many-to-many shortest paths using highway hierarchies. In *9th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 36–45, 2007.
- [184] D. E. Knuth. *The Art of Computer Programming—Sorting and Searching*, volume 3. Addison Wesley, 2nd edition, 1998.
- [185] D. E. Knuth. *MMIXware: A RISC Computer for the Third Millennium*, volume 1750 of *LNCS*. Springer, 1999.
- [186] P. Konecny. Introducing the Cray XMT. In *Cray User Group meeting (CUG)*, 2007.
- [187] D. König. Über Graphen und ihre Anwendung auf Determinantentheorie und Mengenlehre. *Mathematische Annalen*, 77(4):453–465, 1916.
- [188] R. E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27(1):97–109, 1985.
- [189] B. Korte and J. Vygen. *Combinatorial Optimization: Theory and Algorithms*. Springer, 2000.
- [190] J. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.
- [191] V. Kumar, A. Grama, A. Gupta, and G. Karypis. *Introduction to Parallel Computing. Design and Analysis of Algorithms*. Benjamin/Cummings, 1994.
- [192] A. LaMarca and R. E. Ladner. The influence of caches on the performance of heaps. *ACM Journal of Experimental Algorithmics (JEA)*, 1:4:1–4:32, 1996.
- [193] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys. *The Traveling Salesman Problem*. Wiley, 1985.

- [194] LEDA (Library of Efficient Data Types and Algorithms). www.algorithmic-solutions.com.
- [195] L. Q. Lee, A. Lumsdaine, and J. G. Siek. *The Boost graph library: user guide and reference manual*. Addison-Wesley, 2002.
- [196] P. L. Lehman and S. B. Yao. Efficient locking for concurrent operations on b-trees. *ACM Transactions on Database Systems (TODS)*, 6(4):650–670, 1981.
- [197] T. Leighton. *Introduction to Parallel Algorithms and Architectures*. Morgan Kaufmann, 1992.
- [198] N. Leischner, V. Osipov, and P. Sanders. GPU sample sort. In *24th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2010. see also arXiv:0909.5649.
- [199] C. E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, 100(10):892–901, 1985.
- [200] LEMON. LEMON C++ Graph Library. <http://lemon.cs.elte.hu>.
- [201] G. Lev, N. Pippenger, and L. Valiant. A fast parallel algorithm for routing in permutation networks. *IEEE Transactions on Computing*, 30(2):93–100, 1981.
- [202] L. Levin. Universal search problems (in Russian). *Problemy Peredachi Informatsii*, 9(3):265–266, 1973.
- [203] I. Lustig and J.-F. Puget. Program does not equal program: Constraint programming and its relationship to mathematical programming. *Interfaces*, 31(3):29–53, 2001.
- [204] T. Maier and P. Sanders. Dynamic Space Efficient Hashing. In *25th European Symposium on Algorithms (ESA)*, volume 87 of *LIPICs*, pages 58:1–58:14, 2017.
- [205] T. Maier, P. Sanders, and R. Dementiev. Concurrent hash tables: Fast and general?(!). *CoRR*, arXiv:1601.04017 [cs.DS], 2016. short version in PPOPP 2016.
- [206] S. Martello and P. Toth. *Knapsack Problems – Algorithms and Computer Implementations*. Wiley, 1990.
- [207] C. Martínez and S. Roura. Optimal sampling strategies in Quicksort and Quickselect. *SIAM Journal on Computing*, 31(3):683–705, 2002.
- [208] F. Mattern. Algorithms for distributed termination detection. *Distributed Computing*, 2(3):161–175, 1987.
- [209] C. McDiarmid. Concentration. In M. Habib, C. McDiarmid, and J. Ramirez-Alfonsin, editors, *Probabilistic Methods for Algorithmic Discrete Mathematics*, pages 195–247. Springer, 1998.
- [210] C. McGeoch, P. Sanders, R. Fleischer, P. R. Cohen, and D. Precup. Using finite experiments to study asymptotic performance. In *Experimental Algorithmics — From Algorithm Design to Robust and Efficient Software*, volume 2547 of *LNCS*, pages 93–126. Springer, 2002.
- [211] MCSTL: The Multi-Core Standard Template Library. <http://algo2.iti.uni-karlsruhe.de/singler/mcstl/>.
- [212] K. Mehlhorn. A faster approximation algorithm for the Steiner problem in graphs. *Information Processing Letters*, 27(3):125–128, Mar. 1988.

- [213] K. Mehlhorn. Amortisierte Analyse. In T. Ottmann, editor, *Prinzipien des Algorithmenentwurfs*, pages 91–102. Spektrum Lehrbuch, 1998.
- [214] K. Mehlhorn and U. Meyer. External-memory breadth-first search with sub-linear I/O. In *10th European Symposium on Algorithms (ESA)*, volume 2461 of *LNCS*, pages 723–735. Springer, 2002.
- [215] K. Mehlhorn and S. Näher. Bounded ordered dictionaries in $O(\log \log N)$ time and $O(n)$ space. *Information Processing Letters*, 35(4):183–189, 1990.
- [216] K. Mehlhorn and S. Näher. Dynamic fractional cascading. *Algorithmica*, 5(2):215–241, 1990.
- [217] K. Mehlhorn and S. Näher. *The LEDA Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 1999.
- [218] K. Mehlhorn, S. Näher, and P. Sanders. Engineering DFS-based graph algorithms. *arXiv preprint arXiv:1703.10023*, 2017.
- [219] K. Mehlhorn, V. Priebe, G. Schäfer, and N. Sivadasan. All-pairs shortest-paths computation in the presence of negative cycles. *Information Processing Letters*, 81(6):341–343, 2002.
- [220] K. Mehlhorn and P. Sanders. Scanning multiple sequences via cache memory. *Algorithmica*, 35(1):75–93, 2003.
- [221] K. Mehlhorn and S. Saxena. A still simpler way of introducing the interior-point method for linear programming. *Computer Science Review*, 22:1–11, 2016.
- [222] K. Mehlhorn and M. Ziegelmann. Resource constrained shortest paths. In *8th European Symposium on Algorithms (ESA)*, volume 1879 of *LNCS*, pages 326–337. Springer, 2000.
- [223] R. Mendelson, R. E. Tarjan, M. Thorup, and U. Zwick. Melding priority queues. In *9th Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 3111 of *LNCS*, pages 223–235. Springer, 2004.
- [224] *Meyers Konversationslexikon*. Bibliographisches Institut, 1888.
- [225] B. Meyer. *Object-Oriented Software Construction*. Prentice-Hall, second edition, 1997.
- [226] U. Meyer. Average-case complexity of single-source shortest-path algorithms: lower and upper bounds. *Journal of Algorithms*, 48(1):91–134, 2003. preliminary version in SODA 2001.
- [227] U. Meyer and P. Sanders. Δ -stepping: A parallelizable shortest path algorithm. *Journal of Algorithms*, 49(1):114–152, 2003.
- [228] U. Meyer, P. Sanders, and J. Sibeyn, editors. *Algorithms for Memory Hierarchies*, volume 2625 of *LNCS Tutorial*. Springer, 2003.
- [229] G. L. Miller and J. H. Reif. Parallel tree contraction and its application. In *26st IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 478–489, 1985.
- [230] B. M. E. Moret and H. D. Shapiro. An empirical analysis of algorithms for constructing a minimum spanning tree. In *2nd Workshop on Algorithms and Data Structures (WADS)*, volume 519 of *LNCS*, pages 400–411. Springer, 1991.

- [231] R. Morris. Scatter storage techniques. *Communications of the ACM*, 11(1):38–44, 1968.
- [232] S. S. Muchnick. *Advanced Compiler Design and Implementation*. Morgan Kaufmann, 1997.
- [233] I. Müller, P. Sanders, A. Lacurie, W. Lehner, and F. Färber. Cache-efficient aggregation: Hashing is sorting. In *ACM SIGMOD Conference on Management of Data*, pages 1123–1136, 2015.
- [234] I. Müller, P. Sanders, R. Schulze, and W. Zhou. Retrieval and perfect hashing using fingerprinting. In *13th Symposium on Experimental Algorithms (SEA)*, volume 8504 of *LNCS*, pages 138–149. Springer, 2014.
- [235] S. Näher and O. Zlotowski. Design and implementation of efficient data types for static graphs. In *10th European Symposium on Algorithms (ESA)*, volume 2461 of *LNCS*, pages 748–759. Springer, 2002.
- [236] M. Naor and O. Reingold. On the construction of pseudorandom permutations: Luby-Rackoff revisited. *Journal of Cryptology*, 12(1):29–66, 1999.
- [237] G. Navarro. *Compact Data Structures – A Practical Approach*. Cambridge University Press, 2016.
- [238] G. Nemhauser and Z. Ullmann. Discrete dynamic programming and capital allocation. *Management Science*, 15(9):494–505, 1969.
- [239] G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*. Wiley, 1988.
- [240] J. Nešetřil, H. Milková, and H. Nešetřilová. Otakar Borůvka on minimum spanning tree problem: Translation of both the 1926 papers, comments, history. *Discrete Mathematics*, 233:3–36, 2001.
- [241] K. S. Neubert. The Flashsort1 algorithm. *Dr. Dobb's Journal*, pages 123–125, February 1998.
- [242] J. v. Neumann. First draft of a report on the EDVAC. Technical report, University of Pennsylvania, 1945.
- [243] J. Nievergelt and E. Reingold. Binary search trees of bounded balance. *SIAM Journal on Computing*, 2(1):33–43, 1973.
- [244] K. Noshita. A theorem on the expected complexity of Dijkstra's shortest path algorithm. *Journal of Algorithms*, 6(3):400–408, 1985.
- [245] V. Osipov, P. Sanders, and J. Singler. The filter-Kruskal minimum spanning tree algorithm. In *10th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 52–61, 2009.
- [246] A. Pagh, R. Pagh, and M. Ružić. Linear probing with constant independence. *SIAM Journal on Computing*, 39(3):1107–1120, 2009.
- [247] R. Pagh and F. Rodler. Cuckoo hashing. *Journal of Algorithms*, 51(2):122–144, 2004.
- [248] M. Patrascu and M. Thorup. On the k -independence required by linear probing and minwise independence. In *37th International Colloquium on Automata, Languages and Programming (ICALP, Part I)*, volume 6198 of *LNCS*, pages 715–726, 2010.

- [249] W. J. Paul, P. Bach, M. Bosch, J. Fischer, C. Lichtenau, and J. Röhrig. Real PRAM programming. In *8th Euro-Par*, volume 2400 of *LNCS*, pages 522–531. Springer, 2002.
- [250] H.-O. Peitgen and P. H. Richter. *The Beauty of Fractals*. Springer-Verlag, 1986.
- [251] W. W. Peterson. Addressing for random access storage. *IBM Journal of Research and Development*, 1(2), Apr. 1957.
- [252] S. Pettie. Towards a final analysis of pairing heaps. In *46th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 174–183, 2005.
- [253] S. Pettie and V. Ramachandran. An optimal minimum spanning tree algorithm. In *27th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 1853 of *LNCS*, pages 49–60. Springer, 2000.
- [254] J. Pinkerton. *Voyages and Travels*, volume 2. 1808.
- [255] P. J. Plauger, A. A. Stepanov, M. Lee, and D. R. Musser. *The C++ Standard Template Library*. Prentice-Hall, 2000.
- [256] R. C. Prim. Shortest connection networks and some generalizations. *Bell Systems Technical Journal*, 36(6):1389–1401, 1957.
- [257] W. Pugh. Skip lists: A probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6):668–676, 1990.
- [258] M. Pătraşcu and M. Thorup. The power of simple tabulation hashing. *Journal of the ACM*, 59(3):14:1–14:50, 2012.
- [259] M. Rahn, P. Sanders, and J. Singler. Scalable distributed-memory external sorting. In *26th IEEE International Conference on Data Engineering (ICDE)*, pages 685–688, 2010.
- [260] S. Rajasekaran and J. H. Reif. Optimal and sublogarithmic time randomized parallel sorting algorithms. *SIAM Journal on Computing*, 18(3):594–607, 1989.
- [261] O. Ramaré and Y. Saouter. Short effective intervals containing primes. *Journal on Number Theory*, 98(1):10–33, 2003.
- [262] A. Ranade, S. Kothari, and R. Udupa. Register efficient mergesorting. In *7th Conference on High Performance Computing (HIPC)*, volume 1970 of *LNCS*, pages 96–103. Springer, 2000.
- [263] J. H. Reif. Depth-first search is inherently sequential. *Information Processing Letters*, 20(5):229–234, 1985.
- [264] N. Robertson, D. P. Sanders, P. Seymour, and R. Thomas. Efficiently four-coloring planar graphs. In *28th ACM Symposium on Theory of Computing (STOC)*, pages 571–575. ACM Press, 1996.
- [265] G. Robins and A. Zelikowsky. Improved Steiner tree approximation in graphs. In *11th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 770–779, 2000.
- [266] S. Roura. Improved master theorems for divide-and-conquer recurrences. *Journal of the ACM*, 48(2):170–205, 2001.
- [267] P. Sanders. *Lastverteilungsalgorithmen für parallele Tiefensuche*. PhD thesis, University of Karlsruhe, 1996.

- [268] P. Sanders. On the competitive analysis of randomized static load balancing. In S. Rajasekaran, editor, *First Workshop on Randomized Parallel Algorithms*, Honolulu, Hawaii, 16 April, 1996. <http://algo2.iti.kit.edu/sanders/papers/rand96.pdf>.
- [269] P. Sanders. Random permutations on distributed, external and hierarchical memory. *Information Processing Letters*, 67(6):305–310, 1998.
- [270] P. Sanders. Randomized priority queues for fast parallel access. *Journal Parallel and Distributed Computing, Special Issue on Parallel and Distributed Data Structures*, 49(1):86–97, 1998.
- [271] P. Sanders. Fast priority queues for cached memory. *ACM Journal of Experimental Algorithmics*, 5, 2000.
- [272] P. Sanders. Randomized receiver initiated load balancing algorithms for tree shaped computations. *The Computer Journal*, 45(5):561–573, 2002.
- [273] P. Sanders, S. Lamm, L. Hübschle-Schneider, E. Schrade, and C. Dachsbacher. Efficient random sampling – parallel, vectorized, cache-efficient, and online. *ACM Transactions on Mathematical Software*, 44(3), 2018.
- [274] P. Sanders, S. Schlag, and I. Müller. Communication efficient algorithms for fundamental big data problems. In *IEEE Conference on Big Data*, pages 15–23, 2013.
- [275] P. Sanders and D. Schultes. Highway hierarchies hasten exact shortest path queries. In *13th European Symposium on Algorithms (ESA)*, volume 3669 of *LNCS*, pages 568–597. Springer, 2005.
- [276] P. Sanders and C. Schulz. Distributed evolutionary graph partitioning. In *14th Workshop on Algorithm Engineering and Experiments (ALENEX)*, pages 16–29. SIAM, 2012.
- [277] P. Sanders, J. Speck, and J. L. Träff. Two-tree algorithms for full bandwidth broadcast, reduction and scan. *Parallel Computing*, 35(12):581–594, 2009.
- [278] P. Sanders and J. Wassenberg. Engineering a multi-core radix sort. In *17th Euro-Par*, volume 6853 of *LNCS*, pages 160–169. Springer, 2011.
- [279] P. Sanders and S. Winkel. Super scalar sample sort. In *12th European Symposium on Algorithms (ESA)*, volume 3221 of *LNCS*, pages 784–796. Springer, 2004.
- [280] P. Sanders and T. Worsch. *Parallele Programmierung mit MPI – ein Praktikum*. Logos Verlag Berlin, 1997. ISBN 3-931216-76-4.
- [281] R. Santos and F. Seidel. A better upper bound on the number of triangulations of a planar point set. *Journal of Combinatorial Theory Series A*, 102(1):186–193, 2003.
- [282] N. Satish, M. Harris, and M. Garland. Designing efficient sorting algorithms for manycore GPUs. In *23rd IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, 2009.
- [283] R. Schaffer and R. Sedgwick. The analysis of heapsort. *Journal of Algorithms*, 15(1):76–100, 1993.
- [284] A. Schönhage. Storage modification machines. *SIAM Journal on Computing*, 9(3):490–508, 1980.

- [285] A. Schönhage and V. Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7(3–4):281–292, 1971.
- [286] A. Schrijver. *Combinatorial Optimization (3 Volumes)*. Springer Verlag, 2003.
- [287] R. Sedgwick. Analysis of Shellsort and related algorithms. In *4th European Symposium on Algorithms (ESA)*, volume 1136 of *LNCS*, pages 1–11. Springer, 1996.
- [288] R. Sedgwick and P. Flajolet. *An Introduction to the Analysis of Algorithms*. Addison-Wesley, 1996.
- [289] R. Seidel and C. Aragon. Randomized search trees. *Algorithmica*, 16(4–5):464–497, 1996.
- [290] R. Seidel and M. Sharir. Top-down analysis of path compression. *SIAM Journal of Computing*, 34(3):515–525, 2005.
- [291] J. Sewall, J. Chhugani, C. Kim, N. Satish, and P. Dubey. PALM: Parallel Architecture-Friendly Latch-Free Modifications to B+ Trees on Many-Core Processors. *PVLDB*, 4(11):795–806, 2011.
- [292] M. Sharir. A strong-connectivity algorithm and its applications in data flow analysis. *Computers and Mathematics with Applications*, 7(1):67–72, 1981.
- [293] J. C. Shepherdson and H. E. Sturgis. Computability of recursive functions. *Journal of the ACM*, 10(2):217–255, 1963.
- [294] J. Shun and G. E. Blelloch. Phase-concurrent hash tables for determinism. In *26th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 96–107, 2014.
- [295] J. Shun, G. E. Blelloch, J. T. Fineman, and P. B. Gibbons. Reducing contention through priority updates. In *25th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 152–163, 2013.
- [296] J. Shun, G. E. Blelloch, J. T. Fineman, P. B. Gibbons, A. Kyrola, H. V. Simhadri, and K. Tangwongsan. The problem based benchmark suite. In *24th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 68–70, 2012. <http://www.cs.cmu.edu/~pbbs>.
- [297] J. Singler, P. Sanders, and F. Putze. MCSTL: The multi-core standard template library. In *13th Euro-Par*, volume 4641 of *LNCS*, pages 682–694. Springer, 2007.
- [298] M. Sipser. *Introduction to the Theory of Computation*. MIT Press, 1998.
- [299] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences*, 26(3):362–391, 1983.
- [300] D. D. Sleator and R. E. Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3):652–686, 1985.
- [301] D. Spielman and S.-H. Teng. Smoothed analysis of algorithms: why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004.
- [302] M. Stephan and J. Docter. Jülich Supercomputing Centre. JUQUEEN: IBM Blue Gene/Q Supercomputer System at the Jülich Supercomputing Centre. *Journal of large-scale research facilities*, A1, 1 2015.

- [303] A. Stivala, P. J. Stuckey, M. G. de la Banda, M. Hermenegildo, and A. Wirth. Lock-free parallel dynamic programming. *Journal of Parallel and Distributed Computing*, 70(8):839–848, 2010.
- [304] G. L. Taboada, S. Ramos, R. R. Expósito, J. Touriño, and R. Doallo. Java in the high performance computing arena: Research, practice and experience. *Science of Computer Programming*, 78(5):425–444, 2013.
- [305] R. E. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.
- [306] R. E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, 22(2):215–225, 1975.
- [307] R. E. Tarjan. Shortest paths. Technical report, AT&T Bell Laboratories, 1981.
- [308] R. E. Tarjan. Amortized computational complexity. *SIAM Journal on Algebraic and Discrete Methods*, 6(2):306–318, 1985.
- [309] R. E. Tarjan and U. Vishkin. An efficient parallel biconnectivity algorithm. *SIAM Journal on Computing*, 14(4):862–874, 1985.
- [310] M. Thorup. Undirected single source shortest paths in linear time. *Journal of the ACM*, 46(3):362–394, 1999.
- [311] M. Thorup. Even strongly universal hashing is pretty fast. In *11th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 496–497, 2000.
- [312] M. Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *Journal of the ACM*, 51(6):993–1024, 2004.
- [313] M. Thorup. Integer priority queues with decrease key in constant time and the single source shortest paths problem. *Journal of Computer and System Sciences*, 69(3):330–353, 2004.
- [314] M. Thorup and U. Zwick. Approximate distance oracles. *Journal of the ACM*, 52(1):1–24, 2005.
- [315] A. Toom. The complexity of a scheme of functional elements realizing the multiplication of integers. *Soviet Mathematics Doklady*, 150(3):496–498, 1963.
- [316] P. Tsigas and Y. Zhang. A simple, fast parallel implementation of quicksort and its performance evaluation on SUN enterprise 10000. In *11th Euromicro Conference on Parallel, Distributed and Network-Based Processing (PDP)*, pages 372–381. IEEE, 2003.
- [317] Unknown. *Der Handlungsreisende – wie er sein soll und was er zu thun hat, um Auftraege zu erhalten und eines gluecklichen Erfolgs in seinen Geschaeften gewiss zu sein – Von einem alten Commis-Voyageur*. 1832.
- [318] L. Valiant. A bridging model for parallel computation. *Communications of the ACM*, 33(8):103–111, 1990.
- [319] P. van Emde Boas. Preserving order in a forest in less than logarithmic time. *Information Processing Letters*, 6(3):80–82, 1977.
- [320] R. Vanderbei. *Linear Programming: Foundations and Extensions*. Springer, 2001.
- [321] P. J. Varman, S. D. Scheuffler, B. R. Iyer, and G. R. Ricard. Merging multiple lists on hierarchical-memory multiprocessors. *Journal on Parallel & Distributed Computing*, 12(2):171–177, 1991.

- [322] V. Vazirani. *Approximation Algorithms*. Springer, 2000.
- [323] C. Vetter. Fast and exact mobile navigation with openstreetmap data. Master's thesis, KIT, 2010.
- [324] J. Vuillemin. A data structure for manipulating priority queues. *Communications of the ACM*, 21(4):309–314, 1978.
- [325] L. Wall, T. Christiansen, and J. Orwant. *Programming Perl*. O'Reilly, 3rd edition, 2000.
- [326] J. Wassenberg, W. Middelmann, and P. Sanders. An efficient parallel algorithm for graph-based image segmentation. In *13th Conference on Computer Analysis of Images and Patterns (CAIP)*, pages 1003–1010, 2009.
- [327] I. Wegener. BOTTOM-UP-HEAPSORT, a new variant of HEAPSORT beating, on an average, QUICKSORT (if n is not very small). *Theoretical Computer Science*, 118(1):81–98, 1993.
- [328] I. Wegener. *Complexity Theory: Exploring the Limits of Efficient Algorithms*. Springer, 2005.
- [329] R. Wickremesinghe, L. Arge, J. S. Chase, and J. S. Vitter. Efficient sorting using registers and caches. *ACM Journal of Experimental Algorithmics*, 7(9), 2002.
- [330] R. Wilhelm and D. Maurer. *Compiler Design*. Addison-Wesley, 1995.
- [331] J. W. J. Williams. Algorithm 232: Heapsort. *Communications of the ACM*, 7(6):347–348, 1964.
- [332] A. Yasin. A top-down method for performance analysis and counters architecture. *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 35–44, 2014.
- [333] W. Zhou. A practical scalable shared-memory parallel algorithm for computing minimum spanning trees. Master's thesis, Karlsruhe Institute of Technology, 2017.
- [334] A. L. Zobrist. A new hashing method with application for game playing. Technical Report TR88, Computer Sciences Department, University of Wisconsin, Madison, 1970.

Index

- p*, **38**
 - @, 45
 - 15-puzzle, 381

 - Aarts, E. H. L., 390
 - (*a, b*)-tree, *see under* sorted sequence
 - Abello, J., 360
 - Ackermann, W., 348
 - Ackermann function (inverse), 348
 - active message, 427
 - addition, **2**
 - address, 33
 - Adel'son-Vel'skii, G. M., 262
 - adjacency array, *see under* graph
 - adjacency list, *see under* graph
 - adjacency matrix, *see under* graph
 - adjacent, **69**
 - Aggarwal, A., 193
 - aggregation, 139
 - Aho, A. V., 263
 - Ahuja, R. K., 319, 361
 - Ajtai, M., 215
 - Akhremtsev, Y., 259
 - Akra, M., 56
 - al-Khwarizmi, Muhammad ibn Musa, **1**, 10
 - ALD, *see under* shortest path
 - algorithm, **1**
 - algorithm analysis, **52**, *see also*
 - running time, 53
 - amortized, 84, 227, 254, 321
 - accounting method, **101**
 - binary counter, 104
 - deamortization, 103
 - general definition, **104**
 - operation sequence, **105**
 - parallel, 108
 - potential method, **101**
 - token, **101**
 - unbounded array, 101
 - universality of potential method, **107**
- approximation algorithm, 373
 - average case, **57**, 125, 162, 171, 172, 183,
189, 214, 243, 317, 323, 377
 - global, **57**
 - master theorem, **54**, 164
 - parallel, **62**
 - randomized, **63**, 98, 171, 172, 183, 194
 - recursion, 13, 21, 54, 164
 - recursive, 13, 18
 - smoothed analysis, 398
 - sum, 7, **53**
 - worst case, 173
- algorithm design, 1
 - "make the common case fast", 101
 - algebraic, 13, 128, 131, 160, 269, 274
 - black-box solvers, 365, 381, 398
 - certificate, **49**, 52, 71, 297
 - deterministic, **66**, 158
 - diversification, 371
 - divide-and-conquer, **11**, 49, 54, 164
 - building a heap, 222
 - mergesort, 164
 - MSD radix sort, 188
 - multiplication, 11
 - multiway mergesort, 193
 - parallel, 404

- parallel multiplication, 16
- quicksort, 172, 183
- dynamic programming, **374**, 397, 398
 - Bellman–Ford algorithm, 324
 - changing money, 377
 - edit distance, 378
 - knapsack, 375, 377
 - matrix products, chained, 377
 - parallel, 378
 - principle of optimality, **374**, 378
 - shortest paths, 309
- evolutionary algorithm, **395**, 398
 - parallel, 397
- greedy, 160, **371**, 393, 398
 - changing money, 377
 - cycle detection, 71
 - Dijkstra’s algorithm, 313
 - Jarník–Prim algorithm, 342
 - knapsack, 369, 371
 - Kruskal’s algorithm, 344
 - machine scheduling, 373
 - parallel, 300, 303, 374
- local search, **385**, 398
 - hill climbing, 386
 - relaxing constraints, 392
 - restarts, 394
 - simplex algorithm, 386
 - simulated annealing, 388
 - tabu search, 394
 - threshold acceptance, 394
- lookup table, 321
- multilevel algorithm, **96**
- parallel recursion, 16
- portfolio approach, 371
- preprocessing, 49, 158
- random sampling, 194, 360
- randomized, **63**, 135, 214, 263, 351, 398
 - Las Vegas, **68**, 126, 172, 183
 - load balancing, 436
 - Monte Carlo, **68**, 98, 160
- randomized algorithm
 - load balancing, 440
- recursion, 11, 13, 73, 165, 172, 177, 183, 188, 222, 288, 379
- result checking, 10, **49**, 160, 315
- systematic search, **379**, 382, 398
 - constraint programming, 381, **399**
 - ILP solving, 382
 - iterative deepening, 381
 - knapsack, 379
 - parallel, 383
 - use of sorting, 49, 157–159, 215, 271, 369
- algorithm engineering, **1**, 9, 14, 17, 134, 151, 175, 193, 213, 214, 261, 316, 328, 393, 397
- alignment, 11, 261, 460, 461, 471
- all-gather, 163, 196, **420**, 476
 - on hypercube, 421
- all-pairs shortest path, *see under* shortest path
- all-reduce, 142, **412**, 476
 - on hypercube, 412
- all-to-all, 80, 141, 196, **421**, 476
 - MPI, 202
 - nonuniform, 142, 197, 425
 - two-phase algorithm, 425
 - on hypercube, 422
- all-to-all broadcast, **420**
- allocate**, **33**
- Alon, N., 154
- amortized, *see under* algorithm analysis
- analysis, *see* algorithm analysis
- ancestor, **73**
- AND, **30**
- Andersson, A, 214
- antisymmetric, **448**
- Applegate, D. L., 358
- approximation algorithm, 339, **372**, 434, 444
- approximation ratio, **372**
- Aragon, C. R., 393
- Aragon, S. R., 263
- arbitrage, 326
- Arge, L., 81, 212
- arithmetical unit, 459
- arithmetics, 30
- Arora, N. S., 116, 438, 445
- array, **33**, 83
 - access `[·]`, **99**
 - associative, 121
 - build*, **122**
 - find*, **122**
 - forall*, **122**
 - insert*, **122**
 - remove*, **122**
- circular, **110**, 112, 319
- growing, 100
- popBack, **99**
- pushBack, **99**

- reallocate*, 100
- shrinking, 100
- size, **99**
- sorting, 175
- unbounded, **99**, 267
- assertion, **47**
- assignment, 34
- associative operator, 410
- asymptotic, 15, **27**, 31
- atomic operation, 39, 463, 467, 468
 - hashing, 122
- atomic task, 439
- atomic variable, 469
- Ausiello, G., 75
- average case, *see under* running time
- AVL tree, *see under* sorted sequence
- AWK, 122
- Axtmann, M., 163, 181, 215

- B* (block size), 32
- B-tree, 261
- backoff, 415
- Bader, D. A., 360
- Balyo, T., 371
- bandwidth, 32
- barrier, 198, 207, **416**, 476
 - binomial tree, 416
- base, **2**
- Bast, H., 215
- batch, 122
- Batcher, K. E., 215
- Bayer, R., 261
- Bazzi, L., 56
- Beckmann, A., 233
- Beier, R., 377
- Bellman, R., 324
- Bellman–Ford algorithm, *see under* shortest path
- Bender, M. A., 263, 342
- Bentley, J. L., 177, 214
- Bertsekas, D. P., 399
- best case, *see under* running time
- best-first branch-and-bound, 218
- BFS, *see under* graph
- bin packing, 240, 374
- binary heap, *see under* priority queue
- binary operation, 30
- binary search, *see under* searching, 167
- binary search tree, *see under* sorted sequence
- binary tree, 403, 406
 - distributed construction, 407
- Bingmann, T., 163, 215, 233
- binomial coefficient, **455**
- binomial heap, *see under* priority queue
- binomial tree, **228**, 403, 420, 421
 - broadcast, 404
- bisection method, **51**
- bit operation, **30**
- Bixby, E. E., 358
- Blelloch, G. E., 143, 195, 259, 335, 360
- block, *see* memory block
- BlueGene/Q, 205
- Blum, N., 213, 263
- Blumofe, R. D., 116, 438, 445
- Boolean formula, 370
- Boolean value, **33**
- Boost, 78
 - Bellman–Ford algorithm, 337
 - Dijkstra’s algorithm, 337
 - graph, 274
 - graph traversal, 305, 337
 - MST, 360
 - union–find, 360
- boost, 470
- Borůvka, O., 354
- Botelho, F., 154
- bottleneck shortest path, 339
- bottom-up heap operation, 222
- bounded array, **84**
- branch, 30, 460
- branch prediction, 214, 260
- branch-and-bound, 218, 379
 - parallel, 383
- branch-and-cut, 383
- Brent’s principle, **63**, 434
- Bro Miltersen, P., 154
- broadcast, 163, 179, 196, **403**, 476
 - asynchronous, 427
 - binomial tree, **404**
 - by hardware, 403
 - by two trees, 409
 - hypercube algorithm, 409
 - linear pipeline, 408
 - lower bound, 405, 406
 - naive, 404
 - on a mesh, 410

- pipelined, 406
 - using a tree, 405
- Brodal, G. S., 232, 237
- Bronson, N. G., 259
- Brown, M. R., 118
- Brown, R., 237
- BSP, *see under* machine model
- Buchsbaum, A., 360
- bucket, 194
- bucket sort, *see under* sorting
- bulk operation
 - FIFO, 113
 - hashing, 140
 - priority queue, 233
 - sorted sequence, 259
- Buluc, A., 272

- C, 32
- C++, 23, 32, 38, 78, 117, 152, 212, 236, 262, 274, 337, 360, 467
- C++11/14, 467
- cache, 31
 - block, 460
 - coherence, 461
 - footprint, 471
 - instruction, 460
 - levels, 460
 - limited associativity, 212
 - replacement strategy, 461
 - unified, 460
 - write-back, 461
- cache hierarchy, 403
- cache line, 460
- cache miss, 460
- cache-oblivious, 237, 263
- Caldwell, C. K., 154
- calendar queue, *see under* priority queue
- call by reference, **35**
- call by value, **35**
- carry, 2
- Carter, J., 154
- CAS, **40**, 143, 148, 182, 211, 335, 415, 463, 469, 471
 - 16 byte, 145
- cascading cut, 229
- Casper, J., 259
- casting out nines, 10
- Cayley, A., 274
- census, 157

- certificate, *see* algorithm design
- certification
 - MST, 342
 - strongly connected components, 297
- certifying algorithm, **48**
- Cha, S. K., 259
- Chandra, R., 469
- changing money, 377
- characteristic function, 75
- Chase, S., 212
- Chazelle, B., 264, 360
- checksum, 10
- Cheng, Y., 154
- Cheriyān, J., 305
- Cherkassky, B., 338
- Chernoff bound, 141, 195, 208, 233, 441, **454**
- chess, 121
- Chiang, Y-J, 301
- child, **73**
- Chvátal, V., 358
- Cilk, 79, 470
- CISC, 459
- class, 33, **37**
- clique, *see under* graph
- clock cycle, 32
- clustering, 339
- Coffman, E. G., 240
- Cohen-Or, D., 275
- coherence, 461
- Cole, R., 360, 427
- collective communication, **401**, 476
 - asynchronous, 427, 476
- collision, 123
- combinatorial search, 121
- communication network, 41
- communication pattern
 - one-to- p , 403
 - p -to-one, 403
 - p -to- p , 403
- communication time, **401**
- communication topology, 403
- communication volume, 80
- communicator, 473
- commutative operator, 410
- compare-and-swap, **40**
- comparison, **30**
 - three-way, 50, 172, 173
 - two-way, 51

- comparison-based algorithm, 49, 169
- competitive ratio, **374**
- compiler, 7, 32, 81, 121, 213, 466
 - symbol table, 121
- compiler pragma, 469
- complex number, 37, 158
- complexity, *see also* running time, 31
- complexity theory, 75
- composite data structure, 34
- composite type, 33
- computation, model of, 30
- computer architecture, 459
- concave function, 318, **449**
- conditional branch instruction, 214
- conditional statement, **34**
- cone, 387
- Cong, G., 360
- congruent, 448
- consistency, 462
- constant, **30**
- constant factor, 27, 31
- constraint, 365
- constraint programming, *see under*
 - algorithm design, systematic search
- contention, 40, 139, 150, 151, 462, 465
- contract, **47**
- contraction, 96
- convex, **449**
- convex polytope, 387
- Cook, S.A., 76
- Cook, W. J., 23, 358
- cooling schedule, 390
- coprocessor, 44
- core, 43, 459
- correctness, **46**
- cost vector, 365
- counting votes, 401
- Crauser, A., 335
- critical item, 369
- critical path length, 444
- critical section, **39**
- crossover operation, 397
- C#, 32
- cuneiform script, **83**
- cycle, **70**
 - Hamiltonian, **70**, 74
 - simple, **70**
 - testing for, **71**
- DAG, *see* graph, directed, acyclic
- Dantzig, G. B., 365
- Darwin, C., 397
- data dependency, 31, 460
- data distribution
 - block cyclic, 85
 - blocked, 84
 - cyclic, 84
 - round robin, 84
- data parallelism, 178
- data structure, **VII**
 - circular array, 112
 - (a, b) -tree, 244
 - abstraction, 37
 - augmenting search trees, 256
 - binary search tree, 242
 - deque, 109
 - FIFO queue, 109
 - distributed-memory, 114
 - parallel, 112
 - relaxed, 113
 - shared-memory, 113
 - graph, 265–275
 - hash table, 121–155
 - implementations, 78
 - invariant, *see under* invariant
 - linked list, 88
 - lock, *see* lock
 - lock-free, 46
 - non-blocking, 46
 - persistent, 264
 - priority queue, 217–237
 - addressable, 224
 - binary heap, 219
 - bucket queue, 319
 - external, 231
 - Fibonacci heap, 227
 - pairing heap, 226
 - parallel, 232
 - radix heap, 319
 - red–black tree, 262
 - skip list, 263
 - sorted sequence, 239–264
 - parallel, 258
 - stack, 109
 - unbounded array, 99
 - union–find, 346
 - wait-free, 46
- data structure

- priority queue
 - monotone, 318
- data type, *see* type
- database, 124, 139, 144, 241, 261
- database join, 121
- Davidson, A., 213
- de Berg, M., 361
- De, A., 24
- deadlock, 46, 468
- Dean, J., 142
- decision problem, 75
- declaration, 32, 35
 - implicit, 35
- decrement ($--$), 34
- degree, 69
- Delaunay triangulation, 361
- Demaine, E. D., 263
- Dementiev, R., 143, 213, 214, 233, 264, 350
- deque, 109, 118
 - first*, 109
 - last*, 109
 - popBack*, 109
 - pushFront*, 109
 - pushBack*, 109
 - pushFront*, 109
- dereference, 33
- descendant, 73
- design by contract, 47
- deterministic algorithm, *see under*
 - algorithm design
- Devroye, L., 243
- dictionary, 121, 157
- diet problem, 365
- Dietzfelbinger, M., 154, 155
- digit, 1
- digraph, *see* graph, directed
- Dijkstra's algorithm, *see under*
 - shortest path
- Dijkstra, E., 313, 342
- discrete-event simulation, 218
- disk, *see* hard disk
- dispose**, 33
- distributed memory, 41
- distributed system, 44
- div**, 30
- divide-and-conquer, *see under*
 - algorithm design
- division (integer), 10
- DMA, 465
- dopar, 45
- dot product, 364
- Driscoll, J., 264
- Drmotá, M., 56
- dynamic programming, *see under*
 - algorithm design
- dynamic tree, 345
- edge, 68
 - associated information, 265
 - backward, 277, 291
 - contraction, 305
 - cost, 70
 - cross, 277, 291, 292
 - crossing, 71
 - forward, 277, 291
 - parallel, 265, 274
 - reduced cost, *see also* node potential, 326
 - tree, 277, 291
 - weight, 70, 265
- edge coloring, 426
- edge contraction, 351
- edge query, 266, 269
- edgeArray*, 266
- edit distance, 378
- efficiency, 62, *see* running time
- Eiffel, 77
- eight-queens problem, 381, 392
- element, 33, 157
- embarrassingly parallel, 45
- empty sequence $\langle \rangle$, 34
- equals ($=$), 30
- equivalence relation, 449
- Eratosthenes, 37
- event, 451
- evolutionary algorithm, *see under*
 - algorithm design
- exchange argument, 341, 370
- exclusive OR (\oplus), 30
- execution time, *see* running time
- execution trace, 466
- existence problem, 363
- expected value, 57, 451
- exponential backoff, 415
- exponential search, 51
- external memory, *see also* machine model
 - building heap, 223
 - lower bound, 193
 - merging, 192

- MST, 350
- parallel, 81
- parallel disks, 194
- priority queue, **231**
- queue, 111
- scanning, 192
- semiexternal algorithm, 351
- sorting, 192, 194, 213
- stack, 111

- factorial, 455
- Fakcharoenphol, J., 338
- false*, **30**
- false sharing, **462**, 471
- Farach-Colton, M., 263, 342
- fast memory, 32
- fat-tree, 464
- Feistel permutation, 438
- fence, 417, 462, 469
- Ferizovic, D., 259
- ferry connections, 339
- fetch-and-add, **40**, 301, 410, 436, 463, 469
- Fibonacci, L., 227
- Fibonacci heap, *see under* priority queue
- field (algebraic), **128**, **449**
- field (of variable), 33
- FIFO queue, **109**, 278
 - distributed-memory, 114
 - epoch, 115
 - external-memory, 111
 - first*, **109**
 - parallel, 112
 - popFront*, **109**
 - pushBack*, **109**
 - relaxed, 113
 - shared-memory, 113
 - using circular array, 110
 - using two stacks, 110
- file, 34
- filing card, 239
- Flajolet, P., 56
- Fleischer, R., 236
- floating-point, **30**, 78, 321
- flow, 367
- Floyd, R. W., 81, 185, 213
- “folklore” (result), 118
- for, **35**
- Ford, L. R., Jr., 324
- forest, **71**

- Fortran, 79
- Fotakis, D., 154
- Fredkin, E., 264
- Fredman, M. L., 154, 227, 237
- frequency allocation, 393
- Frigo, M., 237
- Fürer, M., 24
- full-duplex, 422
- function object, 153
- function pointer, 212
- future, 468

- Gabow, H., 305
- Gärtner, B., 399
- Gangal, D., 445
- garbage collection, **79**
- Garey, M. R., 75, 240
- Garland, M., 213
- gather, 196, **420**, 476
 - binomial tree, 420
 - naive, 420
- Geisberger, R., 332
- generic methods, 363
- generic programming, **38**, 236, 274
- genome, 395
- geometric series, *see under* sum, 456
- geometry, 387
- Ghemawat, S., 142
- Giacomoni, J., 112
- Gibbons, P. B., 41, 195
- GMP, 23
- Goldberg, A., 323, 331, 338
- golden ratio, 409
- Goodrich, M. T., 81, 215, 301
- gossiping, **420**
- Graefe, G., 259, 261
- Graham, R. L., 56, 81, 373
- Gramma, A., 62
- Grantham, J. F., 154
- graph, **68**
 - 2-edge-connected components, 297
 - adjacency array, **267**
 - adjacency list, **267**
 - adjacency matrix, **269**
 - undirected, 269
 - average degree, 353
 - BFS, **278**, 308, 443
 - implementation, 304
 - load balancing, 432

- biconnected components, 298, 305
- bidirected, **69**, 265, 268
- bipartite, **49**, 274, 426
- breadth-first search, *see* BFS
- Cayley, 274
- citation network, 265
- clique, **74**, 75
- coloring, 49, **74**, 75, 390, 393
 - fixed- K annealing, 393
 - Kempe chain annealing, 391
 - penalty function annealing, 392
 - XRLF greedy algorithm, 393
- communication network, 277
- complete, 74
- component, **70**
- compression, 275
- connected components, **70**, 279
- construction, 266
- conversion, 266, 267
- counting paths, 270
- cut, 270, 340
- cycle detection, 268
- DAG, *see* graph, directed, acyclic (DAG)
- dense, 269
- depth-first search, *see* DFS
- DFS, 277, **288**, 325
 - backtrack*, 289
 - init*, 289
 - root*, 289
 - traverseNonTreeEdge*, 289
 - traverseTreeEdge*, 289
- diameter, **327**
- directed, **69**
 - acyclic (DAG), **70**, 71, 72, 291, 443
- dynamic, 266, 267
- ear decomposition, 305
- edge, *see under* edge
- edge sequence, 266, 344
- exploration, *see* graph traversal
- face, 274
- grid, 270
- hypergraph, 274
- input, 266
- interval graph, 158, 270
- Kempe chain, 391
- layer, 278
- LEMON, 236, 274
- linked edge objects, 268
- minimum spanning tree, *see*
 - MST
- MST, *see* MST
- multigraph, 265, 274
- navigation, 266, 267
- negative cycle, *see under* shortest path
- network design, 339
- node, *see* node
- open ear decomposition, 299
- output, 266
- parallel
 - Bellman–Ford, 335
 - BFS, 287, 443
 - coloring, 302, 303
 - Dijkstra’s algorithm, 335
 - independent sets, 302
 - representation, 271
 - shortest paths, 334
 - shortest paths in DAGs, 313
 - topological sorting, 300
- partitioning, **271**
 - by sorting, 272
 - by space-filling curve, 272
 - evolutionary, 397
 - parallel, 397
 - web graph, 272
- planar, **71**, 274
 - 4-coloring, 391
 - 5-coloring, 392
 - embedding, 305
 - testing planarity, 305
- random, 326, 393
- random geometric graph, 393
- representation, **265**
- reversal information, 266
- SCC, *see* graph, strongly connected
 - components
- shortest path, *see* shortest path
 - Δ -stepping, 335
- shrunk graph, 292
- sparse, 267
- static, 267
- Steiner tree, 357
 - 2-approximation, 357
- street network, 71
- strongly connected components, **70**, 277, **292**
 - certificate, 297
 - closed, 294

- implementation, 304
- invariant, 294
- more algorithms, 305
- open, 294
- subgraph (induced), **70**
- topological sorting, 292, 300, 312
- transitive closure, 279
- traversal, **277**
- triconnected components, 305
- undirected, **69**
 - parallel traversal, 302
 - vertex, *see* node
 - visitor, 305, 337
- graph partitioning, *see under* graph
- graph500 benchmark, 305
- greater than (>), **30**
- greedy algorithm, *see under* algorithm design
- grid network, 425
- Grossi, R., 264
- group, 274
- grouping, 158
- growth rate, 27
- Guibas, L. J., 264
- Gupta, A., 62

- h*-relation, 80, **425**
- H-tree, 410
- Haeupler, B., 237
- Hagerup, T., 214, 215
- half-space, 387
- Halperin, S., 360
- Hamilton, W. R., 70
- Han, Y., 214, 237
- handle, **33**, 88, 218, 240
- Handler, G., 338
- hard disk, 32
- hardware thread, 459, 460
- harmonic numbers, 456
- harmonic sum, *see under* sum
- Harrelson, C., 331
- Harris, M., 213
- Hartmanis, J., 80
- hash function, 122
- hash table, *see* hashing
- hashing, **121**, 158
 - available implementations, 151
 - closed, **132**
 - concurrent, 143
 - insertOrUpdate, **138**
 - large elements, 152
 - large keys, 152
 - linear probing, 123, 132
 - cyclic, 134
 - find*, **132**
 - insert*, **132**
 - parallel, 143
 - remove*, **133**
 - unbounded, 134
 - open, **132**
 - perfect, 135
 - perfect (dynamic), 137
 - realistic analysis, 127
 - shared memory, 143
 - shared memory implementation, 145
 - universal, 126
 - bit strings, 128
 - by integer multiplication, 131
 - by shifting, 131
 - by table lookup, 131
 - simple linear, 131
 - using bit matrix multiplication, 130
 - using scalar products, 128
 - universal family, 127
 - unrealistic analysis, 125
 - update, **138**
 - use of, 158, 160, 171, 266
 - with chaining, 123, **124**
 - average case, 126
 - fat, **152**
 - find*, **124**
 - implementation, 151
 - insert*, **124**
 - parallel, 144
 - remove*, **124**
 - slim, **152**
 - unbounded, 126
- heap property, **220**, *see also* priority queue
- heapsort, *see under* sorting
- Held, M., 359
- Held–Karp lower bound, *see under* MST
- Hennessy, J. L., 24, 459
- Herlihy, M., 113
- heuristic, 60, 303
- high-performance computing, 44

- hill climbing, *see under* algorithm design,
 - local search
- H_n , *see* sum, harmonic
- Ho, C. T., 409, 411, 414
- Hoare, C. A. R., 81
- Hollerith, H., 157
- Hopcroft, J., 263, 349
- Huddleston, S., 118, 263
- Hübschle-Schneider, L., 185, 187, 235
- Hwang, S., 259
- hyper-threading, 151
- hypercube, 403, 409, **412**, 421–423, 464
 - prefix sum, 412
 - subcube, 412
- hyperplane, 387
- Høyer, P., 237

- I/O step, 32
- Iacono, J., 237
- IBM, 157
- IEEE floating-point, 78
- if**, **34**
- iff, **449**
- ILP, *see* linear program, integer
- imperative programming, 32
- implementation note, 31
- incident, **69**
- increment (++), **34**
- incumbent, 380
- indentation, 34
- independent random variable, **453**
- independent set, 96, 98, **302**
- index, 33, 83
- indicator random variable, 57, 174
- inequality
 - Chernoff, **454**
 - Jensen's, **456**
 - Markov's, 68, **454**
- infinity (∞), **33**, 78
- initialization, **32**
- inlining, **35**
- input, 30
- input size, 26, 29
- inserting into a sequence, 84
- insertion sort, *see under* sorting
- instance, 26
- instruction, **30**, 31
- instruction parallelism, 460
- integer, **33**
- integer arithmetics, **1**
- internal memory, 32
- invariant, **47**, 294
 - data structure invariant, 48, 88, 219, 224, 244, 255, 262, 320, 346
 - loop invariant, **48**, 50, 133, 161
- inverse, **450**
- isoefficiency function, **62**, 167
- Itai, A., 263
- Italiano, G., 264
- item, 88
- iteration, 35
- iterative deepening search, 381
- iterator, *see under* STL

- Jájá, J., 80
- Jarník, V., 342
- Jarník-Prim algorithm, *see under*
 - MST
- Java, 23, 32, 38, 79, 118, 153, 213, 262, 337
 - deque, 118
 - hashCode*, 153
 - HashMap*, 153
 - linked list, 118
 - memory management, 118
 - PriorityQueue*, 236
 - SortedMap*, 262
 - SortedSet*, 262
 - sorting, 213
 - stack, 118
 - TreeMap*, 262
 - TreeSet*, 262
 - vector, 118
- JDSL
 - graph traversal, 305
- Jensen's inequality, 456
- Jensen, K., 32
- JGraphT, 236, 274, 305, 337, 360
 - graph, 236, 274
 - MST, 360
 - union-find, 360
- Jiang, T., 214
- job, 429
- Johnson, D. S., 75, 240, 393
- Johnson, N. L., 438
- Johnsson, S. L., 409, 411, 414
- join
 - hash-, 124
 - parallel, 144, 169

- jump, 30
- Kaligosi, K., 214
- Kaplan, H., 237
- Karatsuba, A., 13
- Karger, D., 360
- Karlin, A., 154
- Karmarkar, N., 368
- Karp, R., 359
- Karypis, G., 62
- Katajainen, J., 119, 232
- Katriel, I., 360
- Keh, T., 233
- Kellerer, H., 363
- Kemp, A. W., 438
- Kempe, A. B., 392
- Kempe chain, *see under* graph
- Kettner, L., 213, 264
- key, 122, 157, 217
- Khachiyan, L., 368
- Kieritz, T., 335
- King, V., 360
- Klein, P. N., 360
- knapsack, **74**, 307
- knapsack problem, 363
 - 2-approximation (*round*), 372
 - as an ILP, 368
 - average case, 377
 - branch-and-bound algorithm, 379
 - dynamic programming, 375
 - by profit, 377
 - evolutionary algorithm, 397
 - fractional, 369, 380
 - fractional solver, 370
 - greedy algorithm, 371
 - local search, 385
 - parallel, 439, 443
 - simulated annealing, 390
 - use of, 363
- knot, 83
- Knuth, D., 56, 81, 153, 214
- Knuth, D. E., 80
- Komlós, J., 154, 215
- Konecny, P., 211
- Konheim, A. G., 263
- König, D., 427
- Korf, R. E., 381
- Korst, J., 390
- Korte, B., 361
- Kosaraju, S. R., 305
- Kothari, S., 212
- Kotz, S., 438
- Kruskal, J., 344
- Kumar, V., 62
- Kurur, P. P., 24
- Ladner, R. E., 237
- LaMarca, A., 237
- Lamm, S., 196
- Landis, E. M., 262
- Larsen, P.-A., 261
- Las Vegas algorithm, *see under*
 - algorithm design, randomized
- latency, 32
- Lawler, E. L., 358, 361
- leading term, 28
- leaf, **73**
- LEDA, 23, 78, 304
 - Bellman–Ford algorithm, 337
 - bounded stack, 118
 - Dijkstra’s algorithm, 337
 - graph, 274
 - graph traversal, 304
 - h_array*, 153
 - list, 117
 - map, 153
 - MST, 360
 - node_pq*, 337
 - priority queue, 236
 - queue, 118
 - sortseq*, 262
 - stack, 118
 - static graph, 274
 - union–find, 360
- Lee, L. W., 274
- Lee, M., 78
- left-to-right maximum, **58**, 64, 174, 318
- Lehman, P.L., 259
- Leighton, T., 80, 414, 465
- Leischner, N., 213
- Leiserson, C. E., 237
- LEMON, 236, 274, 360
 - union–find, 360
 - graph traversal, 305, 337
 - MST, 360
- Lenstra, J. K., 358, 361
- less than (<), **30**
- Lev, G., 427

- Levenshtein distance, 378
- Levin, D., 275
- Levin, L., 76
- lexicographic order, 158, **449**
- Li, M., 214
- linear algebra, 270, 387
- linear order, 158, 338, **449**
- linear preorder, **450**
- linear program (LP), **365**
 - fractional solution, 369
 - integer (ILP), 366, **368**
 - 0 -1 ILP, 369
 - 0 -1 ILP, 382
 - branch-and-cut, 383
 - knapsack, 368
 - pigeonhole principle, 371
 - set covering, 370
 - maximum flow, 368
 - minimum-cost flow, 368
 - mixed integer (MILP), **368**
 - relaxation of ILP, 369
 - rounding, 369
 - shortest path, 366
 - simplex algorithm, 386
 - smoothed analysis, 398
 - solver, 398
 - strict inequality, 387
 - tight inequality, 387
- linearity of expectations, 57, 126, 127, 174, 353, **452**
- Linux, 199
- list, 34, 83, 124, 267
 - blocked, 111, 166, 190
 - bulk insert, **166**
 - circular, 227, 267
 - concat*, 92, 93
 - concatenate, 88, 93
 - doubly linked, **88**, 240
 - dummy item, **89**, 267
 - empty, 89
 - find*, 91, 93
 - findNext*, 92, 93
 - first*, 92, 93
 - head*, 92, 93
 - insert*, **90**, 92, 93
 - interference between ops., 92
 - invariant, 88
 - isEmpty*, 92, 93
 - last*, 92, 93
 - linked, 88
 - makeEmpty*, 92, 93
 - memory management, 89, 92
 - move item, **89**
 - popBack*, 92
 - popFront*, 92, 93
 - pushBack*, 92, 93
 - pushFront*, 92, 93
 - remove*, 89, 92, 93
 - rotation, 92
 - singly linked, **93**, 151
 - size, **92**
 - sorting, 166
 - splice*, **89**, 93
 - swapping sublists, **92**
- list ranking, **94**
 - by doubling, **94**
 - by independent set removal, 96
- load balancing, 45, 63, 115, 178, 383, **429**, 468
 - by prefix sums, **433**
 - dependent tasks, 443
 - master worker
 - by fetch-and-add, 436
 - hierarchical, 435
 - master-worker, 431, **434**
 - randomized static, 431, 436
 - scalability comparison, 430
 - work stealing, 115, 383, 431, **438**, 470
 - randomized, 440
- load instruction, **30**
- local search, *see under* algorithm design
 - parallel, 395
- locate*, *see under* sorted sequence
- lock, **46**, 415, 463, 467, 468
 - binary, **46**
 - queue, 416
 - spin, 415
 - timeout, 468
- lock-free, 46
- logarithm, **448**
- logical operations, **30**
- loop, **35**, 53
- loop fusion, 7
- loop invariant, *see under* invariant, 412
- lower bound, 373
 - “breaking”, 187
 - broadcast, 405, 406
 - element uniqueness, **171**

- external sorting, 193
- minimum, **171**
- pairing heap priority queue, 237
- sorting, **169**
- lower-order term, **28**
- lowest common ancestor, 342
- LP, *see* linear program
- Lucas, É., 110
- Lumsdaine, A., 274
- Lustig, I. J., 399
- Luxen, D., 335
- M* (size of fast memory), 32
- machine instruction, *see* instruction
- machine model, 27, **29**
 - accurate, 31
 - BSP, 80, 287
 - complex, 31
 - distributed memory, **41**
 - external memory, 31
 - parallel, 30, **38**
 - PEM, 81
 - RAM, **29**, 32
 - real, 31
 - sequential, 29
 - shared memory, **38**
 - simple, 31
 - von Neumann, 29
 - word, 214
- machine program, **30**, 32
- machine scheduling, **372**
 - online algorithm, **373**
 - shortest-queue algorithm, **373**
- machine word, 29, 31
- Magnanti, R. L., 361
- Maier, T., 143, 154
- main memory, 461
- makespan, 372, 444
- Mandelbrot set, 431, 436
- map coloring, 391
- MapReduce, 142
- Markov, A., 68
- Markov's inequality, *see under*
 - inequality
- Martello, S., 363
- Martinez, C., 214
- master, 434
- master theorem, *see under* algorithm
 - analysis
- master-worker, *see under* load balancing
- Matias, Y., 41, 195
- mating, 397
- Matoušek, J., 399
- matrix, 269
- matrix products, chained, 377
- Mattern, F., 443
- Mauer, D., 81
- maximization problem, **363**
- maximum flow, 368
- McCreight, E. M., 261
- McDiarmid, C. J. H., 99
- McGeoch, L. A., 393
- McIlroy, M. D., 214
- median, 182, *see also* selection, **450**
- Mehlhorn, K., 78, 118, 154, 263, 264, 304, 305, 319, 327, 335, 338, 358
- Mehnert, J., 264
- member variable, **38**
- memcpy*, 117
- memory
 - allocator, 470
 - initialization, 470
- memory access, 31
- memory block, **32**
- memory cell, *see also* machine word, 29
- memory fence, 415, 462
- memory hierachy, 460
- memory management, **33**, **463**, 470
- memory model, 462
- memory ordering, 469
- memory size, 31
- Mendelson, R., 237
- mergesort, *see under* sorting
- merging, 163, 164, 376
 - external, 192
 - multiway, **192**
 - parallel, 167, 378
- mesh, 461, 464
- Meyer auf der Heide, F., 154
- Meyer, B., 77
- Meyer, U., 305, 323, 336
- Meyerhenke, H., 272
- Michel, L., 399
- microinstruction, 460
- minimization problem, **363**
- minimum spanning forest, *see*
 - MST

- minimum spanning tree, *see*
 - MST
- mobile device, 44
- mod**, **30**
- modulo, 10, **448**
- Monte Carlo algorithm, *see under*
 - algorithm design, randomized
- Moret, B., 359
- Morris, R., 153
- Morton-ordering, 272
- Moseley, T., 112
- most significant distinguishing index, 320
- move-to-front, 60
- MPI, 202, **473**
- msd, *see* most significant distinguishing index
- MST, **339**
 - 2-approximation of TSP, 358
 - Borůvka’s algorithm, 354
 - bottleneck paths, 342
 - certification, 342
 - clustering, 339, 361
 - cut property, **341**, 345
 - cycle property, 341, 360
 - Euclidean, 361
 - external memory, 350
 - Held–Karp lower bound, 359
 - Jarník–Prim algorithm, **342**
 - maximum-cost spanning tree, 340
 - parallel
 - Kruskal, 360
 - semiexternal Kruskal algorithm, 351
 - streaming algorithm, 345
 - uniqueness conditions, 342
 - use of, 339, 357, 361
- Müller, I., 80, 139
- multicore processor, **43**
- multigraph, 265, 274, 426
- multikey quicksort, 177
- multilevel algorithm, 96, 272
- multiplication (integer)
 - Karatsuba, **13**
 - refined, 17
 - recursive, **11**
 - school method, 1, **6**
 - use of, 1
- multithreading, **43**, 467
- Musser, D. R., 78
- mutation, 395
- mutex, 468
- Näher, S., 264, 269
- Nadathur, S., 213
- Näher, S., 78, 304
- Naor, M., 438
- Nemhauser, G., 375, 382
- network, 44, *see also* graph, 464
 - communication network, 68
 - contention, 465
 - design, 339
 - fat-tree, 464
 - hypercube, 464
 - mesh, 464
 - torus, 464
- Neubert, K. S., 214
- Nilsson, S., 214
- node, **68**
 - active, 289
 - associated info., 265
 - depth, 73, 278
 - dfsNum*, 290
 - finishing time, 290
 - interior, **73**
 - marked, 289
 - numbering, 266
 - ordering relation (\prec), 290
 - potential, **326**, 330, 359
 - reached, 278, 314
 - representative, 279, 294
 - scanned, 313
- NodeArray*, **266**, 273
- non-blocking, 46
- nonblocking, 475
- Noshita, K., 317
- NOT, **30**
- NP**, **74**
- NP**-complete, **75**
- NP**-hard, **76**, 369, 444
- NUMA, 198, 203, 205, **461**, 474
 - allocation, 471
- numeric type, 33
- $O(\cdot)$, **27**
- $o(\cdot)$, **27**
- object, 33
- object-oriented, **37**
- objective function, **363**
- of** (in type declaration), 33, 34, 38

- Ofman, Y., 13
 $\Omega(\cdot)$, 27
 $\omega(\cdot)$, 27
 online algorithm, 61, **373**
 OpenMP, 79, 469
 optimization, **363**
 optimization problem, 77, **363**
 OR, **30**
 Orlin, J. B., 319, 361
 Osipov, V., 213, 360
 Ost, K., 427
 oversampling, 194
 owner computes, **85**, 209, 271, 287
- P, 74**
 packet, 406
 padding, 462
 page, **464**
 Pagh, A., 155
 Pagh, R., 154, 155
 pair, **34**
 pairing heap, *see under* priority queue
 parallel, 3, 15, 24, 38, 44, 62, 78–80, 108,
 112, 122, 138, 159, 162, 166, 178,
 184, 190, 232, 258, 271, 300, 374, 378
 addition, 3
 array processing, 84
 DAG traversal, 300
 divide and conquer, 16
 evolutionary algorithm, 397
 following pointers, 94
 graph construction, 271
 graph partitioning, 397
 graph representation, 271
 hardware multiplication, 24
 hashing, 138
 independent sets, 302
 knapsack problem, 439, 443
 linear programming, 368
 local search, 395
 loop, 469
 machine model, **38**
 merging, 167
 multiplication, 15
 partitioning, 180
 recursion, 16
 sampling, 179
 sorted sequence, 258
 sorting, *see* sorting
- parallel assignment, 34
 parallel external memory, 81
 parallel processing, 31, 194, 398
 parameter, **35**
 actual, 35
 formal, 35
 parameterized class, **38**
 parent, **73**
 Pareto, V., 375
 Pareto-optimal, 375, 397
 parser, 73
 partition, 346
 partitioning, 85
 Pascal, 32
 Patashnik, O., 56, 81
 path, **70**
 simple, **70**
 Pătrașcu, M., 142, 155
 Patterson, D. A., 24, 459
 Paul, W. J., 211
 pause instruction, 415
 PE, **38**, 460
 Peitgen, H.-O., 432
 Pemmasani, G., 342
 perf profiler, 465
 performance analysis, 465
 Perl, 122
 permutation, 58, 158, 160
 random, 59, 64
 persistent data structure, 264
 Peru, 83
 Peterson, W. W., 132
 Petrank, E., 154
 Pettie, S., 237, 360
 Pferschy, U., 363
 pigeonhole principle, 370
 pipe, 112
 pipeline, 459
 pipeline stage, 460
 pipelining, 9, 406
 Pippenger, N., 427
 Pisinger, D., 363
 pivot, **172**, 194
 selection, 175, 214
 Plauger, J.J., 78
 Plaxton, C. G., 116, 438, 445
 point-to-point communication, 41
 pointer, **33**

- polynomial, 28, *see also under*
 - running time, 160
- polytope, 387
- Pomerance, C., 154
- population, 395
- postcondition, **47**
- potential function, *see* node, potential
- powers (of numbers), 47
- PRAM, 81, 98
 - aCRQW, **41**, 144
 - CRCW, **38**, 211, 301, 336
 - CREW, **38**
 - CRQW, **41**
 - EREW, **38**
 - QRQW, **40**
- Pratt, V. R., 213
- precedence relation, 69
- precondition, **47**
- predecessor, **84**, 88
- prefix sum, 6, 97, 115, 180, 190, 288, 301, 374, 378, **412**, 433, 476
 - by two trees, 414
 - exclusive, **412**
 - on hypercube, 412
 - on tree, 414
- Priebe, V., 327
- Prim, R. C., 342
- Prim's algorithm, *see* MST, Jarník–Prim alg.
- prime number, 37, 128, 160, **450**
 - abundance, 129
- primitive operation
 - full adder, 2
 - product, 2
- principle of optimality, **374**, 378
- priority queue, **217**
 - addressable, **218**, 224, 315
 - binary heap, **219**, 316
 - addressable, 219, 224
 - bottom-up *deleteMin*, 236
 - building, 222
 - bulk insertion, 224
 - deleteMin*, **221**
 - insert*, **221**
 - invariant, 219
 - siftDown*, **221**
 - siftUp*, **221**
 - binomial heap, 229
 - bounded, 219
 - bucket queue, 237, **319**
 - invariant, 320
 - calendar queue, 237
 - decrease key, **218**, 315
 - deleteMin*, **218**
 - double-ended, 252
 - external, 231
 - fat heap, 237
 - Fibonacci heap, **227**, 316, *see also*
 - priority queue, heap-ordered forest
 - decreaseKey*, **229**
 - deleteMin*, **227**
 - item, **227**
 - rank, **227**
 - heap-ordered forest, **224**
 - cut, **224**
 - decreaseKey*, **226**
 - deleteMin*, **224**
 - insert*, **224**
 - invariant, 224
 - link, **224**
 - merge*, **226**
 - new tree, **224**
 - remove*, **226**
 - hollow heap, 237
 - insert*, **218**
 - integer, 236, 237, **318**
 - item, 224
 - memory management, 235
 - merge*, **218**
 - minimum, **218**, 220, 224
 - monotone, 218, 237, 315, **318**
 - naive, 219, 316
 - pairing heap, **226**, *see also*
 - priority q., heap-ordered forest
 - complexity, 237
 - three-pointer items, 226
 - two-pointer items, 226
 - parallel, 232
 - bulk operation, 233
 - radix heap, **319**
 - base b , 322
 - remove*, **218**
 - thin heap, 237
 - unbounded, 219
 - use of, 161, 193, 214, 218, 219, 315, 351
- priority update principle, 335
- probability, **451**
- probability space, 58, **451**
- problem instance, 26

- procedure, **35**
- process, 474
- processor core, 459
- profiling, 199, 465
- profit vector, *see* cost vector
- program, **30**
- program analysis, *see* algorithm analysis
- programming language, 32, 34, 81
 - functional, 166
 - logical, 166
- programming model, *see* machine model
- Prokop, H., 237
- pseudo-polynomial algorithm, 377
- pseudocode, **32**, 77
- pseudorandom permutation, 437
- Puget, J.-F., 399
- Pugh, W., 263
- Putze, F., 201

- quartile, *see also* selection, 182
- queue, 34, 268, *see also* FIFO
 - parallel, 112
- quickselect, *see under* selection
- quicksort, *see under* sorting
- quipu, **83**

- radix sort, *see under* sorting
- Radzik, T., 338
- Rahn, M., 215
- Rajasekaran, S., 215, 271
- RAM model, *see under* machine model
- Ramachandran, S., 237
- Ramachandran, V., 41, 360
- Raman, R., 214
- Ramaré, O., 154
- Ranade, A., 212, 445
- random access, 122
- random experiment, **451**
- random number, **64**
- random source, 78
- random variable, 57, **451**
 - independent, **453**
 - indicator, **451**
 - product, **453**
- randomized algorithm, *see under*
 - algorithm design; algorithm analysis
- rank, 162, **450**
- ranking, 163
- Rao, S., 338

- realloc*, 117
- receive, 475
- recombination, 395
- record, *see* composite type
- recurrence relation, 13, 21, 51, **54**, 81
- recursion, **36**, *see also under*
 - alg. design; alg. analysis
 - elimination, 177, 235
- red-black tree, *see under* sorted sequence
- reduction, **76**, 163, 234, 401, **410**, 476
 - asynchronous, 427, 443
 - by two trees, 411
 - on vectors, 210
 - pipelined, 411
 - shared-memory, 410
 - tree, 410
- reflexive, **450**
- register, **29**, 31, 43, 459
- Reif, J. H., 215, 271, 305
- Reingold, O., 438
- relation, **450**
 - antisymmetric, **448**
 - equivalence, **449**
 - reflexive, **450**
 - symmetric, **450**
 - total, **450**
 - transitive, **450**
- relaxation, 392, *see also under*
 - linear program
- remainder, **30**
- Remez, O., 275
- removing from a sequence, 84
- repeat**, **35**
- resource, 429
- result checking, *see under* algorithm design
- return**, 36
- Richter, P. H., 432
- ring, 461
- Rivest, R. L., 185, 213
- road map, 68
- Robertson, N., 391
- Robins, G., 358
- Rodeh, M., 263
- Rodler, F., 154
- roll back, 463
- root, *see under* tree
- root PE, 403
- round-robin, 437
- Roura, S., 214

- routing, 465
- run, *see under* sorting
- running time, 26, **31**, 35, *see also*
 - algorithm analysis, 53
 - average case, **26**, 57
 - best case, **26**, 31
 - polynomial, **74**
 - worst case, **26**
- Ružić, M., 155

- Sabela, R., 80
- Saha, C., 24
- sample space, **450**
- Sanders, D. P., 391
- Sanders, P., 80, 139, 143, 154, 181, 185, 187, 196, 201, 213–215, 232, 233, 235, 237, 259, 264, 304, 331, 332, 336, 350, 360, 371, 397, 409, 411, 414, 431, 437, 438, 462
- Santos, R., 275
- Saptharishi, R., 24
- Sarnak, N., 264
- SAT solver, 370
- satisfiability problem, **74**
- satisfiable, 370
- scalability, 63
 - of load balancing, 430
- scan, **412**
- scatter, **421**, 476
 - binomial tree, 421
- Schäfer, G., 327
- Schaffer, R., 236
- scheduling, 218, 307, **372**, *see also under*
 - load balancing
 - threads, 471
- Scheufler, S. D., 208
- Schevon, C., 393
- Schirra, S., 427
- Schlag, S., 80
- Schönhage, A., 23
- Schrijver, A., 399
- Schultes, D., 331, 350
- Schulz, C., 397
- search problem, **363**
- search tree, *see sorted* sequence
- searching, **239**, *see also sorted* sequence
 - binary search, **49**, 77, 158, 194, 245
 - dynamic, **60**
 - exponential, 51, 77
 - linear, 60
 - range, 158
 - shortest path, *see under* shortest path
- Sedgewick, R., 56, 177, 214, 236, 237
- Seidel, R., 263, 275, 348
- selection, **182**
 - deterministic, 213
 - from sorted sequences, 167, 206
 - parallel, 184
 - quickselect, **183**, 210
 - streaming, **184**
 - using two pivots, 185
- self-loop, **69**
- semicolon (in pseudocode), 35
- Sen, S., 237
- send, 474
 - asynchronous, **41**, 233, 427
- sentinel, **91**, 152, 161, 166, 235
- sequence, **34**, 83, 158
 - overview of operations, 117
 - space efficiency, 117
- sequential consistency, 469
- series, *see* sum
- server, 44
- set, **34**
- set covering, 370
- Seymour, P., 391
- Shapiro, H. D., 359
- shared memory, **38**
- Sharir, M., 305
- Shavit, N., 113
- Shell sort, *see under* sorting
- Shepherdson, J. C., 29, 80
- shift, **30**
- Shmoys, D. B., 358
- shortest path, **307**
 - acyclic, 308
 - ALD (average linear Dijkstra), **323**, 338
 - all-pairs, 307
 - all-pairs with negative costs, 326
 - arbitrary edge costs, 324
 - as a linear program, 366
 - A*-search, 330
 - Bellman–Ford algorithm, 324
 - refined, 337
 - bidirectional search, 328
 - bottleneck, 339, 361
 - constrained, 338, 378
 - correctness criterion, 311

- DAG, 312
- Dijkstra's algorithm, 233, **313**
 - invariant, 319
- edge relaxation, **310**
- geometric, 338
- goal-directed search, 330
- hierarchical search, 331, 335
- integer edge cost, 308
- linear average time, 323
- multicriteria, 338
- negative cycle, 309
- nonnegative edge cost, 308
- parallel
 - hierarchical search, 335
- parent pointer, 310
- public transportation, 312
- query, 328
- relaxing of edges, **310**
- single-source, 307
- subpath, 309
- tentative distance, 310
- tree, 309
 - uniqueness, 310
 - unit edge cost, 308
 - use of, 307, 326
- shrunken graph, 292
- Shun, J., 143, 335, 360
- Sibeyn, J., 350
- sibling, **73**
- sibling pointer, 227
- Siek, J. G., 274
- sieve of Eratosthenes, 37
- SIMD, **43**, 151
- Simon, J., 80
- simplex algorithm, *see under*
 - linear programming
- simulated annealing, *see under*
 - algorithm design, local search
- Singler, J., 201, 213, 215, 233, 360
- Sipser, M., 75
- Sivadasan, N., 327
- Skiena, S., 342
- Sleator, D., 118, 237, 263, 264, 345
- slow memory, 32
- snow plow heuristic, 214
- socket, **461**, 464
- solution
 - feasible, **363**
 - potential, **363**
- sorted sequence, 49, **239**
 - (a, b) -tree, **244**
 - split* (node), **247**
 - amortized update cost, **254**
 - augmentation, **256**
 - balance*, **250**
 - build/rebuild*, **252**
 - concatenation, **252**
 - fusing, **250**
 - height, **244**
 - insert*, **245**
 - invariant, **244**
 - item, **246**
 - locate*, **246**
 - parent pointer, **257**
 - reduction, **256**
 - remove*, **249**
 - removing a range, **254**
 - splitter, **244**
 - splitting, **253**
- adaptable, 263
- AVL tree, 262
- binary search tree, **242**
 - degenerate, 242
 - expected height, 243
 - implicit, 244
 - insert*, 242
 - locate*, **242**
 - perfect balance, 242
 - rotation, 244, 251
 - selection, 257
- cache-oblivious, 263
- concatenation, 259
- concurrent access, 259
- finger search, 257
- first*, **240**, 252
- insert*, **239**
- integer, 263
- last*, **240**, 252
- locate*, **239**, 241
- merging, 257
- navigation, 240
- persistent, 264
- pred*, **240**
- randomized search tree, 263
- range searching, 252
- red-black tree, 251
- remove*, **239**
- skip list, 263

- sparse table, 263
- splay tree, 263
- split, 259
- strings, 264
- succ*, **240**
- trie, 264
- use of, **240, 241**
- weight-balanced tree, 263
- sorting, **157**
 - almost sorted inputs, 162
 - bottom-up heapsort, 236
 - bucket, **187**
 - by ranking, 163
 - comparison-based, 187
 - dynamic, 161
 - external, **192**
 - flash, 214
 - heapsort, 218, **223**
 - in-place, **160**, 175
 - insertion, 53, **161**, 166
 - large elements, 212
 - list, 166
 - lower bound, 187
 - mechanical, 157
 - mergesort, **164**, 213
 - multiway merge, **192**
 - numbers, 187, 212, 267
 - parallel, 194
 - bucket, 190
 - by merging, 166
 - by multiway merging, 205
 - distr. memory multiway merging, 209
 - distributed-memory mergesort, 169
 - distributed-memory quicksort, 178
 - fast inefficient, **162**, 196, 211
 - log latency, 210
 - MPI, 202
 - MSD radix, 191
 - overview, 159
 - radix, 191
 - sample (implementation), **198**
 - shared-memory quicksort, 181
 - quicksort, **172**, 212, 214, 243
 - radix, **187**
 - LSD, **188**
 - MSD, **188**, 191, 212
 - random numbers, **189**
 - run formation, 168, 192, 214
 - sample, **194**
 - selection, **160**, 218
 - Shell sort, 214
 - small inputs, 161, 172
 - small subproblems, 175
 - stable algorithm, **187**
 - strings, 177, 187
 - use of, 49, 157–159, 215, 271, 351, 369
 - word model, 214
- source node, **69**
- space-filling curve, 272
 - Morton ordering, 272
 - z-order, 272
- span, 16, **63**, 97
- Speck, J., 409, 411, 414
- speed of light, 460
- speedup, **62**
 - superlinear, 385
- spellchecking, 215
- Spielmann, D., 398
- spin lock, 415
- Spirakis, P., 154
- splitter, 194, 242
- SPMD, **45**, 401, 469, 473
- stack, 34, 36, 109, 110
 - bounded, 110
 - external-memory, 111
 - pop, **109**
 - push, **109**
 - top, **109**
 - unbounded, 110
- statement, 34
- static array, **33**, 83
- statistics, 182
- stencil, 88
- Stepanov, A. A., 78
- Stirling's approximation, 170, 190, **455**
- Stivala, A., 143
- STL, 18, 78, 262
 - deque*, 118
 - iterator, 118, 213
 - list*, 117
 - map*, 262
 - multimap*, 262
 - multiset*, 262
 - parallel, 470
 - priority_queue*, 236
 - set*, 262
 - sort*, 212
 - stack*, 118

- unordered_map*, 152
- unordered_multimap*, 152
- unordered_multiset*, 152
- unordered_set*, 152
- store instruction, **30**
- Strassen, V., 23
- streaming algorithm, 184, 345
- string, 34, 83, 158, 448
- striping, 214
- struct, *see* composite type
- Stuckey, P. J., 143
- Sturgis, H. E., 29, 80
- STXXL, 213, 232, 236
- subcube, 412
- subroutine, **35**
- subtask, 429
- successor, **84**, 88
- succinct data structure, 155
- Sudoku, 391
- sum, 81, *see also under* algorithm analysis
 - estimation by integral, 457
 - geometric, 55, **456**
 - harmonic, 59, 64, 130, 174, 318, 353, 448, **456**
- Sumerian, 83
- Sun, Y., 259
- superscalar, 460
- survival of the fittest, 395
- swap, 34
- sweep-line algorithm, 241
- symmetric, **450**
- synchronization, **415**
- syntax, 32
- Szemerédi, E., 154, 215
- Szpankowski, W., 56

- table, 83
- tablet, 83
- Taboada, L., 80
- tabu list, *see* tabu search
- tabu search, *see under* algorithm design,
 - local search
- tabulation hashing, 142
- tail bound, **454**
- tail recursion, *see* recursion, elimination
- Tardos, E., 154
- target node, **69**
- Tarjan, D., 213

- Tarjan, R. E., 118, 154, 213, 227, 237, 263, 264, 305, 319, 337, 345, 348, 360
- task, 45, 429
 - atomic, 434, 439
 - based programming, 469
 - bundle, 435
 - DAG, 443
 - dependencies, 430
 - graph, 16
 - independent, 430, 439
 - parallelism, 166, 178
 - splitting, 438
- Taylor series, 456
- TBB, 79, 150, 470
- telephone book, 157
- telephone model, 422
- template programming, 38, 213
- Teng, S. H., 398
- termination, **48**, 50
- termination detection, 427, 442
- $\Theta(\cdot)$, **27**
- Thomas, R., 391
- Thompson, K., 379
- Thorup, M., 142, 151, 155, 214, 237, 338
- thread, **43**, 45, 459
 - pinning, 471
- thread pool, 468
- threshold acceptance, *see under*
 - algorithm design, local search
- tie breaking, 207
- tiling, **87**
- time, *see* running time
- time forward processing, 301
- time step, **31**
- Toom, A., 23
- topological sorting, *see under*
 - graph
- torus, 464
- total order, 158, **450**
- Toth, P., 363
- tournament tree, 214
- Tower of Hanoi, 110
- Träff, J. L., 232, 360, 409, 411, 414
- transaction, **39**
 - hardware support, 145
 - roll back, 463
- transactional memory, **463**
- transitive, **450**
- translation, 33–37

- traveling salesman problem, **74**, 75, 77, **358**
 - 2-exchange, 386
 - 3-exchange, 386
 - Held–Karp lower bound, 359
 - hill climbing, 386
- tree, **71**, 242
 - binary, 439
 - binomial, **228**, *see* binomial tree
 - depth, 73
 - dynamic, 345
 - expression tree, **73**
 - H, 410
 - height, **73**
 - implicitly defined, 219
 - in-order numbering, 407, 411
 - interior node, **73**
 - ordered, **73**, **241**
 - reduction, 410
 - representation, 227
 - root, **72**
 - sorting tree, **170**
 - traversal, 73
- tree-shaped computation, **438**
- triangle inequality, **358**, 386
- trie, *see under* sorted sequence
- triple, **34**
- true*, **30**
- truth value, **30**
- Tsigas, P., 181
- Tsitsiklis, J. N., 399
- TSP, *see* traveling salesman problem
- tuple, **34**, 158
- type, **32**

- Udupa, R., 212
- Ullman, J. D., 263, 349
- Ullmann, Z., 375
- unary operation, 30
- unbounded array, 84, **99**
- undefined value (\perp), **33**
- uniform memory, 29
- union–find, 346
 - path compression, 347
 - union by rank, 347
- universe (\mathcal{U}), 363
- upper bound, *see* worst case

- Vöcking, B., 377
- Vachharajani, M., 112
- Valiant, L. G., 80, 427
- van Emde Boas layout, 263
- van Emde Boas, P., 264
- Van Hentenryck, P., 399
- van Kreveld, M., 361
- Vanderbei, R. J., 399
- variable, **32**, 365
- Varman, P. J., 208
- Vazirani, V., 361
- vector (in C++), 117
- verification, 47, 162
- vertex, *see* node
- Vetter, C., 335
- virtual memory, **464**, 470
 - allocation strategy, 464
- Vishkin, U., 305
- visitor, *see under* graph, *see under* graph
- Vitányi, P., 214
- Vitter, J. S., 193, 212
- volatile, 417
- von Neumann, J., 29
- von Neumann machine, *see under* machine model
- Vuillemin, J., 229
- Vygen, J., 361

- wait-free, 46
- Wassenberg, J., 360, 462
- weak scaling, 199
- Wegener, I., 75, 236
- Wegman, M., 154
- Weidling, C., 154, 155
- Westbrook, J., 360
- while**, **35**
- Wickremsinghe, R., 212
- Wilhelm, R., 81
- Williams, J. W. J., 219
- Winkel, S., 196, 214, 237
- Wirth, N., 32
- witness, *see* algorithm design, certificate
- Wolsey, L., 382
- word, *see* machine word, 448
- work, **63**, 97
- work stealing, *see under* load balancing
- worker, 434
- Worsch, T., 431
- worst case, *see under* running time
- write combining, **462**

write contention, **40**

XOR (\oplus), **30**, 321

Yao, S. B., 259

Yasin, A., 465

z-order, 272

Zang, I., 338

Zaroliagis, C., 232

Zelikowski, A., 358

Zhang, Y., 181

Zhou, W., 360

Ziegelmann, M., 338

Ziviani, N., 154

Zlotowski, O., 269

Zobrist, A. L., 131, 155

Zwick, U., 237, 360