

Bibliography

- [1] Milton Abramowitz and Irene A. Stegun, editors. *Handbook of Mathematical Functions*. Dover, 1965.
- [2] G. M. Adel'son-Vel'skiĭ and E. M. Landis. An algorithm for the organization of information. *Soviet Mathematics Doklady*, 3(5):1259–1263, 1962.
- [3] Alok Aggarwal and Jeffrey Scott Vitter. The input/output complexity of sorting and related problems. *Communications of the ACM*, 31(9):1116–1127, 1988.
- [4] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of Mathematics*, 160(2):781–793, 2004.
- [5] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *The Design and Analysis of Computer Algorithms*. Addison-Wesley, 1974.
- [6] Alfred V. Aho, John E. Hopcroft, and Jeffrey D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.
- [7] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.
- [8] Ravindra K. Ahuja, Kurt Mehlhorn, James B. Orlin, and Robert E. Tarjan. Faster algorithms for the shortest path problem. *Journal of the ACM*, 37(2):213–223, 1990.
- [9] Ravindra K. Ahuja and James B. Orlin. A fast and simple algorithm for the maximum flow problem. *Operations Research*, 37(5):748–759, 1989.
- [10] Ravindra K. Ahuja, James B. Orlin, and Robert E. Tarjan. Improved time bounds for the maximum flow problem. *SIAM Journal on Computing*, 18(5):939–954, 1989.
- [11] Miklós Ajtai, Nimrod Megiddo, and Orli Waarts. Improved algorithms and analysis for secretary problems and generalizations. *SIAM Journal on Discrete Mathematics*, 14(1):1–27, 2001.
- [12] Selim G. Akl. *The Design and Analysis of Parallel Algorithms*. Prentice Hall, 1989.
- [13] Mohamad Akra and Louay Bazzi. On the solution of linear recurrence equations. *Computational Optimization and Applications*, 10(2):195–210, 1998.
- [14] Susanne Albers. Online algorithms: A survey. *Mathematical Programming*, 97(1-2):3–26, 2003.
- [15] Noga Alon. Generating pseudo-random permutations and maximum flow algorithms. *Information Processing Letters*, 35:201–204, 1990.

- [16] Arne Andersson. Balanced search trees made simple. In *Proceedings of the Third Workshop on Algorithms and Data Structures*, volume 709 of *Lecture Notes in Computer Science*, pages 60–71. Springer, 1993.
- [17] Arne Andersson. Faster deterministic sorting and searching in linear space. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, pages 135–141, 1996.
- [18] Arne Andersson, Torben Hagerup, Stefan Nilsson, and Rajeev Raman. Sorting in linear time? *Journal of Computer and System Sciences*, 57:74–93, 1998.
- [19] Tom M. Apostol. *Calculus*, volume 1. Blaisdell Publishing Company, second edition, 1967.
- [20] Nimar S. Arora, Robert D. Blumofe, and C. Greg Plaxton. Thread scheduling for multiprogrammed multiprocessors. *Theory of Computing Systems*, 34(2):115–144, 2001.
- [21] Sanjeev Arora. *Probabilistic checking of proofs and the hardness of approximation problems*. PhD thesis, University of California, Berkeley, 1994.
- [22] Sanjeev Arora. The approximability of NP-hard problems. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 337–348, 1998.
- [23] Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM*, 45(5):753–782, 1998.
- [24] Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, 2009.
- [25] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: A meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.
- [26] Sanjeev Arora and Carsten Lund. Hardness of approximations. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 399–446. PWS Publishing Company, 1997.
- [27] Mikhail J. Atallah and Marina Blanton, editors. *Algorithms and Theory of Computation Handbook*, volume 1. Chapman & Hall/CRC Press, second edition, 2009.
- [28] Mikhail J. Atallah and Marina Blanton, editors. *Algorithms and Theory of Computation Handbook*, volume 2. Chapman & Hall/CRC Press, second edition, 2009.
- [29] 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.
- [30] Shai Avidan and Ariel Shamir. Seam carving for content-aware image resizing. *ACM Transactions on Graphics*, 26(3), article 10, 2007.
- [31] László Babai, Eugene M. Luks, and Ákos Seress. Fast management of permutation groups I. *SIAM Journal on Computing*, 26(5):1310–1342, 1997.
- [32] Eric Bach. Private communication, 1989.
- [33] Eric Bach. Number-theoretic algorithms. In *Annual Review of Computer Science*, volume 4, pages 119–172. Annual Reviews, Inc., 1990.
- [34] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory—Volume I: Efficient Algorithms*. The MIT Press, 1996.
- [35] Nikhil Bansal and Anupam Gupta. Potential-function proofs for first-order methods. *CoRR*, abs/1712.04581, 2017.

- [36] Hannah Bast, Daniel Delling, Andrew V. Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F. Werneck. Route planning in transportation networks. In *Algorithm Engineering - Selected Results and Surveys*, volume 9220 of *Lecture Notes in Computer Science*, pages 19–80. Springer, 2016.
- [37] Surender Baswana, Ramesh Hariharan, and Sandeep Sen. Improved decremental algorithms for maintaining transitive closure and all-pairs shortest paths. *Journal of Algorithms*, 62(2):74–92, 2007.
- [38] R. Bayer. Symmetric binary B-trees: Data structure and maintenance algorithms. *Acta Informatica*, 1(4):290–306, 1972.
- [39] R. Bayer and E. M. McCreight. Organization and maintenance of large ordered indexes. *Acta Informatica*, 1(3):173–189, 1972.
- [40] Pierre Beauchemin, Gilles Brassard, Claude Crépeau, Claude Goutier, and Carl Pomerance. The generation of random numbers that are probably prime. *Journal of Cryptology*, 1(1):53–64, 1988.
- [41] L. A. Belady. A study of replacement algorithms for a virtual-storage computer. *IBM Systems Journal*, 5(2):78–101, 1966.
- [42] Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of cipher block chaining message authentication code. *Journal of Computer and System Sciences*, 61(3):362–399, 2000.
- [43] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS '93, Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
- [44] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.
- [45] Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.
- [46] Michael Ben-Or. Lower bounds for algebraic computation trees. In *Proceedings of the Fifteenth Annual ACM Symposium on Theory of Computing*, pages 80–86, 1983.
- [47] Michael A. Bender, Erik D. Demaine, and Martin Farach-Colton. Cache-oblivious B-trees. *SIAM Journal on Computing*, 35(2):341–358, 2005.
- [48] Samuel W. Bent and John W. John. Finding the median requires $2n$ comparisons. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, pages 213–216, 1985.
- [49] Jon L. Bentley. *Writing Efficient Programs*. Prentice Hall, 1982.
- [50] Jon L. Bentley. *More Programming Pearls: Confessions of a Coder*. Addison-Wesley, 1988.
- [51] Jon L. Bentley. *Programming Pearls*. Addison-Wesley, second edition, 1999.
- [52] Jon L. Bentley, Dorothea Haken, and James B. Saxe. A general method for solving divide-and-conquer recurrences. *SIGACT News*, 12(3):36–44, 1980.
- [53] Claude Berge. Two theorems in graph theory. *Proceedings of the National Academy of Sciences*, 43(9):842–844, 1957.
- [54] Aditya Y. Bhargava. *Grokking Algorithms: An Illustrated Guide For Programmers and Other Curious People*. Manning Publications, 2016.

- [55] Daniel Bienstock and Benjamin McClosky. Tightening simplex mixed-integer sets with guaranteed bounds. *Optimization Online*, 2008.
- [56] Patrick Billingsley. *Probability and Measure*. John Wiley & Sons, second edition, 1986.
- [57] Guy E. Blelloch. *Scan Primitives and Parallel Vector Models*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1989. Available as MIT Laboratory for Computer Science Technical Report MIT/LCS/TR-463.
- [58] Guy E. Blelloch. Programming parallel algorithms. *Communications of the ACM*, 39(3):85–97, 1996.
- [59] Guy E. Blelloch, Jeremy T. Fineman, Phillip B. Gibbons, and Julian Shun. Internally deterministic parallel algorithms can be fast. In *17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 181–192, 2012.
- [60] Guy E. Blelloch, Jeremy T. Fineman, Yan Gu, and Yihan Sun. Optimal parallel algorithms in the binary-forking model. In *Proceedings of the 32nd Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 89–102, 2020.
- [61] Guy E. Blelloch, Phillip B. Gibbons, and Yossi Matias. Provably efficient scheduling for languages with fine-grained parallelism. *Journal of the ACM*, 46(2):281–321, 1999.
- [62] Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973.
- [63] Robert D. Blumofe and Charles E. Leiserson. Scheduling multithreaded computations by work stealing. *Journal of the ACM*, 46(5):720–748, 1999.
- [64] Robert L. Bocchino, Jr., Vikram S. Adve, Sarita V. Adve, and Marc Snir. Parallel programming must be deterministic by default. In *Proceedings of the First USENIX Conference on Hot Topics in Parallelism (HotPar)*, 2009.
- [65] Béla Bollobás. *Random Graphs*. Academic Press, 1985.
- [66] Leonardo Bonacci. *Liber Abaci*, 1202.
- [67] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. American Elsevier, 1976.
- [68] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 1998.
- [69] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
- [70] Gilles Brassard and Paul Bratley. *Fundamentals of Algorithmics*. Prentice Hall, 1996.
- [71] Richard P. Brent. The parallel evaluation of general arithmetic expressions. *Journal of the ACM*, 21(2):201–206, 1974.
- [72] Gerth Stølting Brodal. A survey on priority queues. In Andrej Brodnik, Alejandro López-Ortiz, Venkatesh Raman, and Alfredo Viola, editors, *Space-Efficient Data Structures, Streams, and Algorithms: Papers in Honor of J. Ian Munro on the Occasion of His 66th Birthday*, volume 8066 of *Lecture Notes in Computer Science*, pages 150–163. Springer, 2013.
- [73] Gerth Stølting Brodal, George Lagogiannis, and Robert E. Tarjan. Strict Fibonacci heaps. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, pages 1177–1184, 2012.

- [74] George W. Brown. Some notes on computation of games solutions. *RAND Corporation Report*, P-78, 1949.
- [75] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3-4):231–357, 2015.
- [76] Niv Buchbinder and Joseph Naor. The design of competitive online algorithms via a primal-dual approach. *Foundations and Trends in Theoretical Computer Science*, 3(2-3):93–263, 2009.
- [77] J. P. Buhler, H. W. Lenstra, Jr., and Carl Pomerance. Factoring integers with the number field sieve. In A. K. Lenstra and H. W. Lenstra, Jr., editors, *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 50–94. Springer, 1993.
- [78] M. Burrows and D. J. Wheeler. A block-sorting lossless data compression algorithm. SRC Research Report 124, Digital Equipment Corporation Systems Research Center, May 1994.
- [79] Neville Campbell. Recurrences. Unpublished treatise available at <https://nevillecampbell.com/Recurrences.pdf>, 2020.
- [80] J. Lawrence Carter and Mark N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979.
- [81] Barbara Chapman, Gabriele Jost, and Ruud van der Pas. *Using OpenMP: Portable Shared Memory Parallel Programming*. The MIT Press, 2007.
- [82] Philippe Charles, Christian Grothoff, Vijay Saraswat, Christopher Donawa, Allan Kielstra, Kemal Ebcioglu, Christoph Von Praun, and Vivek Sarkar. X10: An object-oriented approach to non-uniform cluster computing. In *ACM SIGPLAN Conference on Object-oriented Programming, Systems, Languages, and Applications (OOPSLA)*, pages 519–538, 2005.
- [83] Bernard Chazelle. A minimum spanning tree algorithm with inverse-Ackermann type complexity. *Journal of the ACM*, 47(6):1028–1047, 2000.
- [84] Ke Chen and Adrian Dumitrescu. Selection algorithms with small groups. *International Journal of Foundations of Computer Science*, 31(3):355–369, 2020.
- [85] Guang-Ien Cheng, Mingdong Feng, Charles E. Leiserson, Keith H. Randall, and Andrew F. Stark. Detecting data races in Cilk programs that use locks. In *Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 298–309, 1998.
- [86] Joseph Cheriyan and Torben Hagerup. A randomized maximum-flow algorithm. *SIAM Journal on Computing*, 24(2):203–226, 1995.
- [87] Joseph Cheriyan and S. N. Maheshwari. Analysis of preflow push algorithms for maximum network flow. *SIAM Journal on Computing*, 18(6):1057–1086, 1989.
- [88] Boris V. Cherkassky and Andrew V. Goldberg. On implementing the push-relabel method for the maximum flow problem. *Algorithmica*, 19(4):390–410, 1997.
- [89] Boris V. Cherkassky, Andrew V. Goldberg, and Tomasz Radzik. Shortest paths algorithms: Theory and experimental evaluation. *Mathematical Programming*, 73(2):129–174, 1996.
- [90] Boris V. Cherkassky, Andrew V. Goldberg, and Craig Silverstein. Buckets, heaps, lists and monotone priority queues. *SIAM Journal on Computing*, 28(4):1326–1346, 1999.
- [91] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23(4):493–507, 1952.

- [92] Brian Christian and Tom Griffiths. *Algorithms to Live By: The Computer Science of Human Decisions*. Picador, 2017.
- [93] Kai Lai Chung. *Elementary Probability Theory with Stochastic Processes*. Springer, 1974.
- [94] V. Chvátal. *Linear Programming*. W. H. Freeman and Company, 1983.
- [95] V. Chvátal, D. A. Klarnier, and D. E. Knuth. Selected combinatorial research problems. Technical Report STAN-CS-72-292, Computer Science Department, Stanford University, 1972.
- [96] Alan Cobham. The intrinsic computational difficulty of functions. In *Proceedings of the 1964 Congress for Logic, Methodology, and the Philosophy of Science*, pages 24–30. North-Holland, 1964.
- [97] H. Cohen and H. W. Lenstra, Jr. Primality testing and Jacobi sums. *Mathematics of Computation*, 42(165):297–330, 1984.
- [98] Michael B. Cohen, Aleksander Madry, Piotr Sankowski, and Adrian Vladu. Negative-weight shortest paths and unit capacity minimum cost flow in $\widetilde{O}(m^{10/7} \log w)$ time (extended abstract). In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms*, pages 752–771, 2017.
- [99] Douglas Comer. The ubiquitous B-tree. *ACM Computing Surveys*, 11(2):121–137, 1979.
- [100] Stephen Cook. The complexity of theorem proving procedures. In *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.
- [101] James W. Cooley and John W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.
- [102] Don Coppersmith. Modifications to the number field sieve. *Journal of Cryptology*, 6(3):169–180, 1993.
- [103] Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progression. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
- [104] Thomas H. Cormen. *Algorithms Unlocked*. The MIT Press, 2013.
- [105] Thomas H. Cormen, Thomas Sundquist, and Leonard F. Wisniewski. Asymptotically tight bounds for performing BMMC permutations on parallel disk systems. *SIAM Journal on Computing*, 28(1):105–136, 1998.
- [106] Don Dailey and Charles E. Leiserson. Using Cilk to write multiprocessor chess programs. In H. J. van den Herik and B. Monien, editors, *Advances in Computer Games*, volume 9, pages 25–52. University of Maastricht, Netherlands, 2001.
- [107] Sanjoy Dasgupta, Christos Papadimitriou, and Umesh Vazirani. *Algorithms*. McGraw-Hill, 2008.
- [108] Abraham de Moivre. De fractionibus algebraicis radicalitate immunibus ad fractiones simpliciores reducendis, deque summandis terminis quarundam serierum aequali intervallo a se distantibus. *Philosophical Transactions*, 32(373):162–168, 1722.
- [109] Erik D. Demaine, Dion Harmon, John Iacono, and Mihai Pătraşcu. Dynamic optimality—almost. *SIAM Journal on Computing*, 37(1):240–251, 2007.
- [110] Camil Demetrescu, David Eppstein, Zvi Galil, and Giuseppe F. Italiano. Dynamic graph algorithms. In Mikhail J. Attalah and Marina Blanton, editors, *Algorithms and Theory of Computation Handbook*, chapter 9, pages 9-1–9-28. Chapman & Hall/CRC, second edition, 2009.

- [111] Camil Demetrescu and Giuseppe F. Italiano. Fully dynamic all pairs shortest paths with real edge weights. *Journal of Computer and System Sciences*, 72(5):813–837, 2006.
- [112] Eric V. Denardo and Bennett L. Fox. Shortest-route methods: 1. Reaching, pruning, and buckets. *Operations Research*, 27(1):161–186, 1979.
- [113] Martin Dietzfelbinger, Torben Hagerup, Jyrki Katajainen, and Martti Penttonen. A reliable randomized algorithm for the closest-pair problem. *Journal of Algorithms*, 25(1):19–51, 1997.
- [114] Martin Dietzfelbinger, Anna Karlin, Kurt Mehlhorn, Friedhelm Meyer auf der Heide, Hans Rohnert, and Robert E. Tarjan. Dynamic perfect hashing: Upper and lower bounds. *SIAM Journal on Computing*, 23(4):738–761, 1994.
- [115] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.
- [116] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271, 1959.
- [117] Edsger W. Dijkstra. *A Discipline of Programming*. Prentice-Hall, 1976.
- [118] Dimitar Dimitrov, Martin Vechev, and Vivek Sarkar. Race detection in two dimensions. *ACM Transactions on Parallel Computing*, 4(4):1–22, 2018.
- [119] E. A. Dinic. Algorithm for solution of a problem of maximum flow in a network with power estimation. *Soviet Mathematics Doklady*, 11(5):1277–1280, 1970.
- [120] Brandon Dixon, Monika Rauch, and Robert E. Tarjan. Verification and sensitivity analysis of minimum spanning trees in linear time. *SIAM Journal on Computing*, 21(6):1184–1192, 1992.
- [121] John D. Dixon. Factorization and primality tests. *The American Mathematical Monthly*, 91(6):333–352, 1984.
- [122] Dorit Dor, Johan Håstad, Staffan Ulfberg, and Uri Zwick. On lower bounds for selecting the median. *SIAM Journal on Discrete Mathematics*, 14(3):299–311, 2001.
- [123] Dorit Dor and Uri Zwick. Selecting the median. *SIAM Journal on Computing*, 28(5):1722–1758, 1999.
- [124] Dorit Dor and Uri Zwick. Median selection requires $(2 + \epsilon)n$ comparisons. *SIAM Journal on Discrete Mathematics*, 14(3):312–325, 2001.
- [125] Alvin W. Drake. *Fundamentals of Applied Probability Theory*. McGraw-Hill, 1967.
- [126] James R. Driscoll, Neil Sarnak, Daniel D. Sleator, and Robert E. Tarjan. Making data structures persistent. *Journal of Computer and System Sciences*, 38(1):86–124, 1989.
- [127] Ran Duan, Seth Pettie, and Hsin-Hao Su. Scaling algorithms for weighted matching in general graphs. *ACM Transactions on Algorithms*, 14(1):8:1–8:35, 2018.
- [128] Richard Durstenfeld. Algorithm 235 (RANDOM PERMUTATION). *Communications of the ACM*, 7(7):420, 1964.
- [129] Derek L. Eager, John Zahorjan, and Edward D. Lazowska. Speedup versus efficiency in parallel systems. *IEEE Transactions on Computers*, 38(3):408–423, 1989.
- [130] Jack Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

- [131] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971.
- [132] Jack Edmonds and Richard M. Karp. Theoretical improvements in the algorithmic efficiency for network flow problems. *Journal of the ACM*, 19(2):248–264, 1972.
- [133] Jeff Edmonds. *How To Think About Algorithms*. Cambridge University Press, 2008.
- [134] Mourad Elloumi and Albert Y. Zomaya, editors. *Algorithms in Computational Molecular Biology: Techniques, Approaches and Applications*. John Wiley & Sons, 2011.
- [135] Jeff Erickson. *Algorithms*. <https://archive.org/details/Algorithms-Jeff-Erickson>, 2019.
- [136] Martin Erwig. *Once Upon an Algorithm: How Stories Explain Computing*. The MIT Press, 2017.
- [137] Shimon Even. *Graph Algorithms*. Computer Science Press, 1979.
- [138] Shimon Even and Yossi Shiloach. An on-line edge-deletion problem. *Journal of the ACM*, 28(1):1–4, 1981.
- [139] William Feller. *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, third edition, 1968.
- [140] Mingdong Feng and Charles E. Leiserson. Efficient detection of determinacy races in Cilk programs. In *Proceedings of the 9th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 1–11, 1997.
- [141] Amos Fiat, Richard M. Karp, Michael Luby, Lyle A. McGeoch, Daniel Dominic Sleator, and Neal E. Young. Competitive paging algorithms. *Journal of Algorithms*, 12(4):685–699, 1991.
- [142] Amos Fiat and Gerhard J. Woeginger, editors. *Online Algorithms, The State of the Art*, volume 1442 of *Lecture Notes in Computer Science*. Springer, 1998.
- [143] Sir Ronald A. Fisher and Frank Yates. *Statistical Tables for Biological, Agricultural and Medical Research*. Hafner Publishing Company, fifth edition, 1957.
- [144] Robert W. Floyd. Algorithm 97 (SHORTEST PATH). *Communications of the ACM*, 5(6):345, 1962.
- [145] Robert W. Floyd. Algorithm 245 (TREESORT). *Communications of the ACM*, 7(12):701, 1964.
- [146] Robert W. Floyd. Permuting information in idealized two-level storage. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 105–109. Plenum Press, 1972.
- [147] Robert W. Floyd and Ronald L. Rivest. Expected time bounds for selection. *Communications of the ACM*, 18(3):165–172, 1975.
- [148] L. R. Ford. *Network Flow Theory*. RAND Corporation, Santa Monica, CA, 1956.
- [149] Lestor R. Ford, Jr. and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.
- [150] Lestor R. Ford, Jr. and Selmer M. Johnson. A tournament problem. *The American Mathematical Monthly*, 66(5):387–389, 1959.
- [151] E. W. Forgy. Cluster analysis of multivariate efficiency versus interpretability of classifications. *Biometrics*, 21(3):768–769, 1965.

- [152] Lance Fortnow. *The Golden Ticket: P, NP, and the Search for the Impossible*. Princeton University Press, 2013.
- [153] Michael L. Fredman. New bounds on the complexity of the shortest path problem. *SIAM Journal on Computing*, 5(1):83–89, 1976.
- [154] Michael L. Fredman, János Komlós, and Endre Szemerédi. Storing a sparse table with $O(1)$ worst case access time. *Journal of the ACM*, 31(3):538–544, 1984.
- [155] Michael L. Fredman and Michael E. Saks. The cell probe complexity of dynamic data structures. In *Proceedings of the Twenty First Annual ACM Symposium on Theory of Computing*, pages 345–354, 1989.
- [156] Michael L. Fredman and Robert E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM*, 34(3):596–615, 1987.
- [157] Michael L. Fredman and Dan E. Willard. Surpassing the information theoretic bound with fusion trees. *Journal of Computer and System Sciences*, 47(3):424–436, 1993.
- [158] Michael L. Fredman and Dan E. Willard. Trans-dichotomous algorithms for minimum spanning trees and shortest paths. *Journal of Computer and System Sciences*, 48(3):533–551, 1994.
- [159] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.
- [160] Matteo Frigo, Pablo Halpern, Charles E. Leiserson, and Stephen Lewin-Berlin. Reducers and other Cilk++ hyperobjects. In *Proceedings of the 21st Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 79–90, 2009.
- [161] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005.
- [162] Hannah Fry. *Hello World: Being Human in the Age of Algorithms*. W. W. Norton & Company, 2018.
- [163] Harold N. Gabow. Path-based depth-first search for strong and biconnected components. *Information Processing Letters*, 74(3–4):107–114, 2000.
- [164] Harold N. Gabow. The weighted matching approach to maximum cardinality matching. *Fundamenta Informaticae*, 154(1-4):109–130, 2017.
- [165] Harold N. Gabow, Z. Galil, T. Spencer, and Robert E. Tarjan. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica*, 6(2):109–122, 1986.
- [166] Harold N. Gabow and Robert E. Tarjan. A linear-time algorithm for a special case of disjoint set union. *Journal of Computer and System Sciences*, 30(2):209–221, 1985.
- [167] Harold N. Gabow and Robert E. Tarjan. Faster scaling algorithms for network problems. *SIAM Journal on Computing*, 18(5):1013–1036, 1989.
- [168] Harold N. Gabow and Robert Endre Tarjan. Faster scaling algorithms for general graph-matching problems. *Journal of the ACM*, 38(4):815–853, 1991.
- [169] D. Gale and L. S. Shapley. College admissions and the stability of marriage. *American Mathematical Monthly*, 69(1):9–15, 1962.
- [170] Zvi Galil and Oded Margalit. All pairs shortest distances for graphs with small integer length edges. *Information and Computation*, 134(2):103–139, 1997.

- [171] Zvi Galil and Oded Margalit. All pairs shortest paths for graphs with small integer length edges. *Journal of Computer and System Sciences*, 54(2):243–254, 1997.
- [172] Zvi Galil and Kunsoo Park. Dynamic programming with convexity, concavity and sparsity. *Theoretical Computer Science*, 92(1):49–76, 1992.
- [173] Zvi Galil and Joel Seiferas. Time-space-optimal string matching. *Journal of Computer and System Sciences*, 26(3):280–294, 1983.
- [174] Igal Galperin and Ronald L. Rivest. Scapegoat trees. In *Proceedings of the 4th ACM-SIAM Symposium on Discrete Algorithms*, pages 165–174, 1993.
- [175] Michael R. Garey, R. L. Graham, and J. D. Ullman. Worst-case analysis of memory allocation algorithms. In *Proceedings of the Fourth Annual ACM Symposium on Theory of Computing*, pages 143–150, 1972.
- [176] Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.
- [177] Naveen Garg and Jochen Könemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. *SIAM Journal on Computing*, 37(2):630–652, 2007.
- [178] Saul Gass. *Linear Programming: Methods and Applications*. International Thomson Publishing, fourth edition, 1975.
- [179] Fănică Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972.
- [180] Alan George and Joseph W-H Liu. *Computer Solution of Large Sparse Positive Definite Systems*. Prentice Hall, 1981.
- [181] E. N. Gilbert and E. F. Moore. Variable-length binary encodings. *Bell System Technical Journal*, 38(4):933–967, 1959.
- [182] Ashish Goel, Sanjeev Khanna, Daniel H. Larkin, and Robert E. Tarjan. Disjoint set union with randomized linking. In *Proceedings of the 25th ACM-SIAM Symposium on Discrete Algorithms*, pages 1005–1017, 2014.
- [183] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995.
- [184] Michel X. Goemans and David P. Williamson. The primal-dual method for approximation algorithms and its application to network design problems. In Dorit S. Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, pages 144–191. PWS Publishing Company, 1997.
- [185] Andrew V. Goldberg. *Efficient Graph Algorithms for Sequential and Parallel Computers*. PhD thesis, Department of Electrical Engineering and Computer Science, MIT, 1987.
- [186] Andrew V. Goldberg. Scaling algorithms for the shortest paths problem. *SIAM Journal on Computing*, 24(3):494–504, 1995.
- [187] Andrew V. Goldberg and Satish Rao. Beyond the flow decomposition barrier. *Journal of the ACM*, 45(5):783–797, 1998.
- [188] Andrew V. Goldberg and Robert E. Tarjan. A new approach to the maximum flow problem. *Journal of the ACM*, 35(4):921–940, 1988.

- [189] D. Goldfarb and M. J. Todd. Linear programming. In G. L. Nemhauser, A. H. G. Rinnooy-Kan, and M. J. Todd, editors, *Handbooks in Operations Research and Management Science, Vol. 1, Optimization*, pages 73–170. Elsevier Science Publishers, 1989.
- [190] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
- [191] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal on Computing*, 17(2):281–308, 1988.
- [192] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, third edition, 1996.
- [193] G. H. Gonnet and R. Baeza-Yates. *Handbook of Algorithms and Data Structures in Pascal and C*. Addison-Wesley, second edition, 1991.
- [194] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. Addison-Wesley, 1992.
- [195] Michael T. Goodrich and Roberto Tamassia. *Algorithm Design: Foundations, Analysis, and Internet Examples*. John Wiley & Sons, 2001.
- [196] Michael T. Goodrich and Roberto Tamassia. *Data Structures and Algorithms in Java*. John Wiley & Sons, sixth edition, 2014.
- [197] Ronald L. Graham. Bounds for certain multiprocessor anomalies. *Bell System Technical Journal*, 45(9):1563–1581, 1966.
- [198] Ronald L. Graham and Pavol Hell. On the history of the minimum spanning tree problem. *Annals of the History of Computing*, 7(1):43–57, 1985.
- [199] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics*. Addison-Wesley, second edition, 1994.
- [200] David Gries. *The Science of Programming*. Springer, 1981.
- [201] M. Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*. Springer, 1988.
- [202] Leo J. Guibas and Robert Sedgewick. A dichromatic framework for balanced trees. In *Proceedings of the 19th Annual Symposium on Foundations of Computer Science*, pages 8–21, 1978.
- [203] Dan Gusfield and Robert W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. The MIT Press, 1989.
- [204] Gregory Gutin and Abraham P. Punnen, editors. *The Traveling Salesman Problem and Its Variations*. Kluwer Academic Publishers, 2002.
- [205] Torben Hagerup. Improved shortest paths on the word RAM. In *Proceedings of 27th International Colloquium on Automata, Languages and Programming, ICALP 2000*, volume 1853 of *Lecture Notes in Computer Science*, pages 61–72. Springer, 2000.
- [206] H. Halberstam and R. E. Ingram, editors. *The Mathematical Papers of Sir William Rowan Hamilton*, volume III (Algebra). Cambridge University Press, 1967.
- [207] Yijie Han. Improved fast integer sorting in linear space. *Information and Computation*, 170(1):81–94, 2001.
- [208] Frank Harary. *Graph Theory*. Addison-Wesley, 1969.

- [209] Gregory C. Harfst and Edward M. Reingold. A potential-based amortized analysis of the union-find data structure. *SIGACT News*, 31(3):86–95, 2000.
- [210] J. Hartmanis and R. E. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.
- [211] Michael T. Heideman, Don H. Johnson, and C. Sidney Burrus. Gauss and the history of the Fast Fourier Transform. *IEEE ASSP Magazine*, 1(4):14–21, 1984.
- [212] Monika R. Henzinger and Valerie King. Fully dynamic biconnectivity and transitive closure. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, pages 664–672, 1995.
- [213] Monika R. Henzinger and Valerie King. Randomized fully dynamic graph algorithms with polylogarithmic time per operation. *Journal of the ACM*, 46(4):502–516, 1999.
- [214] Monika R. Henzinger, Satish Rao, and Harold N. Gabow. Computing vertex connectivity: New bounds from old techniques. *Journal of Algorithms*, 34(2):222–250, 2000.
- [215] Nicholas J. Higham. Exploiting fast matrix multiplication within the level 3 BLAS. *ACM Transactions on Mathematical Software*, 16(4):352–368, 1990.
- [216] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, second edition, 2002.
- [217] W. Daniel Hillis and Jr. Guy L. Steele. Data parallel algorithms. *Communications of the ACM*, 29(12):1170–1183, 1986.
- [218] C. A. R. Hoare. Algorithm 63 (PARTITION) and algorithm 65 (FIND). *Communications of the ACM*, 4(7):321–322, 1961.
- [219] C. A. R. Hoare. Quicksort. *The Computer Journal*, 5(1):10–15, 1962.
- [220] Dorit S. Hochbaum. Efficient bounds for the stable set, vertex cover and set packing problems. *Discrete Applied Mathematics*, 6(3):243–254, 1983.
- [221] Dorit S. Hochbaum, editor. *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, 1997.
- [222] W. Hoeffding. On the distribution of the number of successes in independent trials. *Annals of Mathematical Statistics*, 27(3):713–721, 1956.
- [223] Micha Hofri. *Probabilistic Analysis of Algorithms*. Springer, 1987.
- [224] John E. Hopcroft and Richard M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.
- [225] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, third edition, 2006.
- [226] John E. Hopcroft and Robert E. Tarjan. Efficient algorithms for graph manipulation. *Communications of the ACM*, 16(6):372–378, 1973.
- [227] John E. Hopcroft and Jeffrey D. Ullman. Set merging algorithms. *SIAM Journal on Computing*, 2(4):294–303, 1973.
- [228] John E. Hopcroft and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.
- [229] Juraĳ Hromkoviĉ. *Algorithmics for Hard Problems: Introduction to Combinatorial Optimization, Randomization, Approximation, and Heuristics*. Springer-Verlag, 2001.

- [230] T. C. Hu and M. T. Shing. Computation of matrix chain products. Part I. *SIAM Journal on Computing*, 11(2):362–373, 1982.
- [231] T. C. Hu and M. T. Shing. Computation of matrix chain products. Part II. *SIAM Journal on Computing*, 13(2):228–251, 1984.
- [232] T. C. Hu and A. C. Tucker. Optimal computer search trees and variable-length alphabetic codes. *SIAM Journal on Applied Mathematics*, 21(4):514–532, 1971.
- [233] David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, 1952.
- [234] Oscar H. Ibarra and Chul E. Kim. Fast approximation algorithms for the knapsack and sum of subset problems. *Journal of the ACM*, 22(4):463–468, 1975.
- [235] E. J. Isaac and R. C. Singleton. Sorting by address calculation. *Journal of the ACM*, 3(3):169–174, 1956.
- [236] David S. Johnson. Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, 9(3):256–278, 1974.
- [237] David S. Johnson. The NP-completeness column: An ongoing guide—The tale of the second prover. *Journal of Algorithms*, 13(3):502–524, 1992.
- [238] Donald B. Johnson. Efficient algorithms for shortest paths in sparse networks. *Journal of the ACM*, 24(1):1–13, 1977.
- [239] Richard Johnsonbaugh and Marcus Schaefer. *Algorithms*. Pearson Prentice Hall, 2004.
- [240] Neil C. Jones and Pavel Pevzner. *An Introduction to Bioinformatics Algorithms*. The MIT Press, 2004.
- [241] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu. A local search approximation algorithm for k -means clustering. *Computational Geometry*, 28:89–112, 2004.
- [242] A. Karatsuba and Yu. Ofman. Multiplication of multidigit numbers on automata. *Soviet Physics—Doklady*, 7(7):595–596, 1963. Translation of an article in *Doklady Akademii Nauk SSSR*, 145(2), 1962.
- [243] David R. Karger, Philip N. Klein, and Robert E. Tarjan. A randomized linear-time algorithm to find minimum spanning trees. *Journal of the ACM*, 42(2):321–328, 1995.
- [244] David R. Karger, Daphne Koller, and Steven J. Phillips. Finding the hidden path: Time bounds for all-pairs shortest paths. *SIAM Journal on Computing*, 22(6):1199–1217, 1993.
- [245] Juha Kärkkäinen, Peter Sanders, and Stefan Burkhardt. Linear work suffix array construction. *Journal of the ACM*, 53(6):918–936, 2006.
- [246] Howard Karloff. *Linear Programming*. Birkhäuser, 1991.
- [247] N. Karmarkar. A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4):373–395, 1984.
- [248] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
- [249] Richard M. Karp. An introduction to randomized algorithms. *Discrete Applied Mathematics*, 34(1–3):165–201, 1991.

- [250] Richard M. Karp and Michael O. Rabin. Efficient randomized pattern-matching algorithms. *IBM Journal of Research and Development*, 31(2):249–260, 1987.
- [251] A. V. Karzanov. Determining the maximal flow in a network by the method of preflows. *Soviet Mathematics Doklady*, 15(2):434–437, 1974.
- [252] Toru Kasai, Gunho Lee, Hiroki Arimura, Setsuo Arikawa, and Kunsoo Park. Linear-time longest-common-prefix computation in suffix arrays and its applications. In *Proceedings of the 12th Annual Symposium on Combinatorial Pattern Matching*, volume 2089, pages 181–192. Springer-Verlag, 2001.
- [253] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. CRC Press, second edition, 2015.
- [254] Valerie King. A simpler minimum spanning tree verification algorithm. *Algorithmica*, 18(2):263–270, 1997.
- [255] Valerie King, Satish Rao, and Robert E. Tarjan. A faster deterministic maximum flow algorithm. *Journal of Algorithms*, 17(3):447–474, 1994.
- [256] Philip N. Klein and Neal E. Young. Approximation algorithms for NP-hard optimization problems. In *CRC Handbook on Algorithms*, pages 34-1–34-19. CRC Press, 1999.
- [257] Jon Kleinberg and Éva Tardos. *Algorithm Design*. Addison-Wesley, 2006.
- [258] Robert D. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proceedings of the 16th ACM-SIAM Symposium on Discrete Algorithms*, pages 630–631, 2005.
- [259] Donald E. Knuth. *Fundamental Algorithms*, volume 1 of *The Art of Computer Programming*. Addison-Wesley, third edition, 1997.
- [260] Donald E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, third edition, 1997.
- [261] Donald E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, second edition, 1998.
- [262] Donald E. Knuth. *Combinatorial Algorithms*, volume 4A of *The Art of Computer Programming*. Addison-Wesley, 2011.
- [263] Donald E. Knuth. *Satisfiability*, volume 4, fascicle 6 of *The Art of Computer Programming*. Addison-Wesley, 2015.
- [264] Donald E. Knuth. Optimum binary search trees. *Acta Informatica*, 1(1):14–25, 1971.
- [265] Donald E. Knuth. Big omicron and big omega and big theta. *SIGACT News*, 8(2):18–23, 1976.
- [266] Donald E. Knuth. *Stable Marriage and Its Relation to Other Combinatorial Problems: An Introduction to the Mathematical Analysis of Algorithms*, volume 10 of *CRM Proceedings and Lecture Notes*. American Mathematical Society, 1997.
- [267] Donald E. Knuth, James H. Morris, Jr., and Vaughan R. Pratt. Fast pattern matching in strings. *SIAM Journal on Computing*, 6(2):323–350, 1977.
- [268] Mykel J. Kochenderfer and Tim A. Wheeler. *Algorithms for Optimization*. The MIT Press, 2019.
- [269] J. Komlós. Linear verification for spanning trees. *Combinatorica*, 5(1):57–65, 1985.
- [270] Dexter C. Kozen. *The Design and Analysis of Algorithms*. Springer, 1992.

- [271] David W. Krumme, George Cybenko, and K. N. Venkataraman. Gossiping in minimal time. *SIAM Journal on Computing*, 21(1):111–139, 1992.
- [272] Joseph B. Kruskal, Jr. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50, 1956.
- [273] Harold W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.
- [274] William Kuszmaul and Charles E. Leiserson. Floors and ceilings in divide-and-conquer recurrences. In *Proceedings of the 3rd SIAM Symposium on Simplicity in Algorithms*, pages 133–141, 2021.
- [275] Leslie Lamport. How to make a multiprocessor computer that correctly executes multiprocess programs. *IEEE Transactions on Computers*, C-28(9):690–691, 1979.
- [276] Eugene L. Lawler. *Combinatorial Optimization: Networks and Matroids*. Holt, Rinehart, and Winston, 1976.
- [277] Eugene L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, editors. *The Traveling Salesman Problem*. John Wiley & Sons, 1985.
- [278] François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 2014 International Symposium on Symbolic and Algebraic Computation, (ISSAC)*, pages 296–303, 2014.
- [279] Doug Lea. A Java fork/join framework. In *ACM 2000 Conference on Java Grande*, pages 36–43, 2000.
- [280] C. Y. Lee. An algorithm for path connection and its applications. *IRE Transactions on Electronic Computers*, EC-10(3):346–365, 1961.
- [281] Edward A. Lee. The problem with threads. *IEEE Computer*, 39(3):33–42, 2006.
- [282] I-Ting Angelina Lee, Charles E. Leiserson, Tao B. Schardl, Zhunping Zhang, and Jim Sukha. On-the-fly pipeline parallelism. *ACM Transactions on Parallel Computing*, 2(3):17:1–17:42, 2015.
- [283] I-Ting Angelina Lee and Tao B. Schardl. Efficient race detection for reducer hyperobjects. *ACM Transactions on Parallel Computing*, 4(4):1–40, 2018.
- [284] Mun-Kyu Lee, Pierre Michaud, Jeong Seop Sim, and Daehun Nyang. A simple proof of optimality for the MIN cache replacement policy. *Information Processing Letters*, 116(2):168–170, 2016.
- [285] Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: Solving linear programs in $\widetilde{O}(\sqrt{\text{rank}})$ iterations and faster algorithms for maximum flow. In *Proceedings of the 55th Annual Symposium on Foundations of Computer Science*, pages 424–433, 2014.
- [286] Tom Leighton. Tight bounds on the complexity of parallel sorting. *IEEE Transactions on Computers*, C-34(4):344–354, 1985.
- [287] Tom Leighton. Notes on better master theorems for divide-and-conquer recurrences. Class notes. Available at <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.39.1636>, 1996.
- [288] Tom Leighton and Satish Rao. Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *Journal of the ACM*, 46(6):787–832, 1999.

- [289] Daan Leijen and Judd Hall. Optimize managed code for multi-core machines. *MSDN Magazine*, 2007.
- [290] Charles E. Leiserson. The Cilk++ concurrency platform. *Journal of Supercomputing*, 51(3):244–257, March 2010.
- [291] Charles E. Leiserson. Cilk. In David Padua, editor, *Encyclopedia of Parallel Computing*, pages 273–288. Springer, 2011.
- [292] Charles E. Leiserson, Tao B. Schardl, and Jim Sukha. Deterministic parallel random-number generation for dynamic-multithreading platforms. In *Proceedings of the 17th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pages 193–204, 2012.
- [293] Charles E. Leiserson, Neil C. Thompson, Joel S. Emer, Bradley C. Kuszmaul, Butler W. Lampson, Daniel Sanchez, and Tao B. Schardl. There’s plenty of room at the Top: What will drive computer performance after Moore’s law? *Science*, 368(6495), 2020.
- [294] Debra A. Lelewer and Daniel S. Hirschberg. Data compression. *ACM Computing Surveys*, 19(3):261–296, 1987.
- [295] A. K. Lenstra, H. W. Lenstra, Jr., M. S. Manasse, and J. M. Pollard. The number field sieve. In A. K. Lenstra and H. W. Lenstra, Jr., editors, *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 11–42. Springer, 1993.
- [296] H. W. Lenstra, Jr. Factoring integers with elliptic curves. *Annals of Mathematics*, 126(3):649–673, 1987.
- [297] L. A. Levin. Universal sequential search problems. *Problems of Information Transmission*, 9(3):265–266, 1973. Translated from the original Russian article in *Problemy Peredachi Informatsii* 9(3): 115–116, 1973.
- [298] Anany Levitin. *Introduction to the Design & Analysis of Algorithms*. Addison-Wesley, third edition, 2011.
- [299] Harry R. Lewis and Christos H. Papadimitriou. *Elements of the Theory of Computation*. Prentice Hall, second edition, 1998.
- [300] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.
- [301] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.
- [302] C. L. Liu. *Introduction to Combinatorial Mathematics*. McGraw-Hill, 1968.
- [303] Yang P. Liu and Aaron Sidford. Faster energy maximization for faster maximum flow. In *Proceedings of the 52nd Annual ACM Symposium on Theory of Computing*, pages 803–814, 2020.
- [304] S. P. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.
- [305] Panos Louridas. *Real-World Algorithms: A Beginner’s Guide*. The MIT Press, 2017.
- [306] László Lovász and Michael D. Plummer. *Matching Theory*, volume 121 of *Annals of Discrete Mathematics*. North Holland, 1986.
- [307] John MacCormick. *9 Algorithms That Changed the Future: The Ingenious Ideas That Drive Today’s Computers*. Princeton University Press, 2012.

- [308] Aleksander Madry. Navigating central path with electrical flows: From flows to matchings, and back. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science*, pages 253–262, 2013.
- [309] Bruce M. Maggs and Serge A. Plotkin. Minimum-cost spanning tree as a path-finding problem. *Information Processing Letters*, 26(6):291–293, 1988.
- [310] M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k -means problem is NP-hard. In S. Das and R. Uehara, editors, *WALCOM 2009: Algorithms and Computation*, volume 5431 of *Lecture Notes in Computer Science*, pages 274–285. Springer, 2009.
- [311] Michael Main. *Data Structures and Other Objects Using Java*. Addison-Wesley, 1999.
- [312] Udi Manber and Gene Myers. Suffix arrays: A new method for on-line string searches. *SIAM Journal on Computing*, 22(5):935–948, 1993.
- [313] David F. Manlove. *Algorithmics of Matching Under Preferences*, volume 2 of *Series on Theoretical Computer Science*. World Scientific, 2013.
- [314] Giovanni Manzini. An analysis of the Burrows-Wheeler transform. *Journal of the ACM*, 48(3):407–430, 2001.
- [315] Mario Andrea Marchisio, editor. *Computational Methods in Synthetic Biology*. Humana Press, 2015.
- [316] William J. Masek and Michael S. Paterson. A faster algorithm computing string edit distances. *Journal of Computer and System Sciences*, 20(1):18–31, 1980.
- [317] Yu. V. Matiyasevich. Real-time recognition of the inclusion relation. *Journal of Soviet Mathematics*, 1(1):64–70, 1973. Translated from the original Russian article in *Zapiski Nauchnykh Seminarov Leningradskogo Otdeleniya Matematicheskogo Instituta im. V. A. Steklova Akademii Nauk SSSR* 20: 104–114, 1971.
- [318] H. A. Maurer, Th. Ottmann, and H.-W. Six. Implementing dictionaries using binary trees of very small height. *Information Processing Letters*, 5(1):11–14, 1976.
- [319] Ernst W. Mayr, Hans Jürgen Prömel, and Angelika Steger, editors. *Lectures on Proof Verification and Approximation Algorithms*, volume 1367 of *Lecture Notes in Computer Science*. Springer, 1998.
- [320] Catherine C. McGeoch. All pairs shortest paths and the essential subgraph. *Algorithmica*, 13(5):426–441, 1995.
- [321] Catherine C. McGeoch. *A Guide to Experimental Algorithmics*. Cambridge University Press, 2012.
- [322] Andrew McGregor. Graph stream algorithms: A survey. *SIGMOD Record*, 43(1):9–20, 2014.
- [323] M. D. McIlroy. A killer adversary for quicksort. *Software—Practice and Experience*, 29(4):341–344, 1999.
- [324] Kurt Mehlhorn and Stefan Näher. *LEDA: A Platform for Combinatorial and Geometric Computing*. Cambridge University Press, 1999.
- [325] Kurt Mehlhorn and Peter Sanders. *Algorithms and Data Structures: The Basic Toolbox*. Springer, 2008.
- [326] Dinesh P. Mehta and Sartaj Sahni. *Handbook of Data Structures and Applications*. Chapman and Hall/CRC, second edition, 2018.

- [327] Gary L. Miller. Riemann's hypothesis and tests for primality. *Journal of Computer and System Sciences*, 13(3):300–317, 1976.
- [328] Marvin Minsky and Seymour A. Papert. *Perceptrons*. The MIT Press, 1969.
- [329] John C. Mitchell. *Foundations for Programming Languages*. The MIT Press, 1996.
- [330] Joseph S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k -MST, and related problems. *SIAM Journal on Computing*, 28(4):1298–1309, 1999.
- [331] Michael Mitzenmacher and Eli Upfal. *Probability and Computing*. Cambridge University Press, second edition, 2017.
- [332] Louis Monier. *Algorithmes de Factorisation D'Entiers*. PhD thesis, L'Université Paris-Sud, 1980.
- [333] Louis Monier. Evaluation and comparison of two efficient probabilistic primality testing algorithms. *Theoretical Computer Science*, 12(1):97–108, 1980.
- [334] Edward F. Moore. The shortest path through a maze. In *Proceedings of the International Symposium on the Theory of Switching*, pages 285–292. Harvard University Press, 1959.
- [335] Rajeev Motwani, Joseph (Seffi) Naor, and Prabhakar Raghavan. Randomized approximation algorithms in combinatorial optimization. In Dorit Hochbaum, editor, *Approximation Algorithms for NP-Hard Problems*, chapter 11, pages 447–481. PWS Publishing Company, 1997.
- [336] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
- [337] James Munkres. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics*, 5(1):32–38, 1957.
- [338] J. I. Munro and V. Raman. Fast stable in-place sorting with $O(n)$ data moves. *Algorithmica*, 16(2):151–160, 1996.
- [339] Yoichi Muraoka and David J. Kuck. On the time required for a sequence of matrix products. *Communications of the ACM*, 16(1):22–26, 1973.
- [340] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.
- [341] S. Muthukrishnan. Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), 2005.
- [342] Richard Neapolitan. *Foundations of Algorithms*. Jones & Bartlett Learning, fifth edition, 2014.
- [343] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*, volume 87 of *Applied Optimization*. Springer, 2004.
- [344] J. Nievergelt and E. M. Reingold. Binary search trees of bounded balance. *SIAM Journal on Computing*, 2(1):33–43, 1973.
- [345] Ivan Niven and Herbert S. Zuckerman. *An Introduction to the Theory of Numbers*. John Wiley & Sons, fourth edition, 1980.
- [346] National Institute of Standards and Technology. Hash functions. <https://csrc.nist.gov/projects/hash-functions>, 2019.
- [347] Alan V. Oppenheim and Ronald W. Schaffer, with John R. Buck. *Discrete-Time Signal Processing*. Prentice Hall, second edition, 1998.

- [348] Alan V. Oppenheim and Alan S. Willsky, with S. Hamid Nawab. *Signals and Systems*. Prentice Hall, second edition, 1997.
- [349] James B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 78(1):109–129, 1997.
- [350] James B. Orlin. Max flows in $O(nm)$ time, or better. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*, pages 765–774, 2013.
- [351] Anna Pagh, Rasmus Pagh, and Milan Ruzic. Linear probing with constant independence. <https://arxiv.org/abs/cs/0612055>, 2006.
- [352] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [353] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice Hall, 1982.
- [354] Michael S. Paterson. Progress in selection. In *Proceedings of the Fifth Scandinavian Workshop on Algorithm Theory*, pages 368–379, 1996.
- [355] Seth Pettie. A new approach to all-pairs shortest paths on real-weighted graphs. *Theoretical Computer Science*, 312(1):47–74, 2004.
- [356] Seth Pettie and Vijaya Ramachandran. An optimal minimum spanning tree algorithm. *Journal of the ACM*, 49(1):16–34, 2002.
- [357] Seth Pettie and Vijaya Ramachandran. A shortest path algorithm for real-weighted undirected graphs. *SIAM Journal on Computing*, 34(6):1398–1431, 2005.
- [358] Steven Phillips and Jeffery Westbrook. Online load balancing and network flow. *Algorithmica*, 21(3):245–261, 1998.
- [359] Serge A. Plotkin, David. B. Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.
- [360] J. M. Pollard. Factoring with cubic integers. In A. K. Lenstra and H. W. Lenstra, Jr., editors, *The Development of the Number Field Sieve*, volume 1554 of *Lecture Notes in Mathematics*, pages 4–10. Springer, 1993.
- [361] Carl Pomerance. On the distribution of pseudoprimes. *Mathematics of Computation*, 37(156):587–593, 1981.
- [362] Carl Pomerance, editor. *Proceedings of the AMS Symposia in Applied Mathematics: Computational Number Theory and Cryptography*. American Mathematical Society, 1990.
- [363] William K. Pratt. *Digital Image Processing*. John Wiley & Sons, fourth edition, 2007.
- [364] Franco P. Preparata and Michael Ian Shamos. *Computational Geometry: An Introduction*. Springer, 1985.
- [365] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C++: The Art of Scientific Computing*. Cambridge University Press, second edition, 2002.
- [366] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, third edition, 2007.
- [367] R. C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401, 1957.

- [368] Robert L. Probert. On the additive complexity of matrix multiplication. *SIAM Journal on Computing*, 5(2):187–203, 1976.
- [369] William Pugh. Skip lists: A probabilistic alternative to balanced trees. *Communications of the ACM*, 33(6):668–676, 1990.
- [370] Simon J. Puglisi, W. F. Smyth, and Andrew H. Turpin. A taxonomy of suffix array construction algorithms. *ACM Computing Surveys*, 39(2), 2007.
- [371] Paul W. Purdom, Jr. and Cynthia A. Brown. *The Analysis of Algorithms*. Holt, Rinehart, and Winston, 1985.
- [372] Michael O. Rabin. Probabilistic algorithms. In J. F. Traub, editor, *Algorithms and Complexity: New Directions and Recent Results*, pages 21–39. Academic Press, 1976.
- [373] Michael O. Rabin. Probabilistic algorithm for testing primality. *Journal of Number Theory*, 12(1):128–138, 1980.
- [374] P. Raghavan and C. D. Thompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.
- [375] Rajeev Raman. Recent results on the single-source shortest paths problem. *SIGACT News*, 28(2):81–87, 1997.
- [376] James Reinders. *Intel Threading Building Blocks: Outfitting C++ for Multi-core Processor Parallelism*. O'Reilly Media, Inc., 2007.
- [377] Edward M. Reingold, Kenneth J. Urban, and David Gries. K-M-P string matching revisited. *Information Processing Letters*, 64(5):217–223, 1997.
- [378] Hans Riesel. *Prime Numbers and Computer Methods for Factorization*, volume 126 of *Progress in Mathematics*. Birkhäuser, second edition, 1994.
- [379] Ronald L. Rivest, M. J. B. Robshaw, R. Sidney, and Y. L. Yin. The RC6 block cipher. In *First Advanced Encryption Standard (AES) Conference*, 1998.
- [380] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. See also U.S. Patent 4,405,829.
- [381] Herbert Robbins. A remark on Stirling's formula. *American Mathematical Monthly*, 62(1):26–29, 1955.
- [382] Julia Robinson. An iterative method of solving a game. *The Annals of Mathematics*, 54(2):296–301, 1951.
- [383] Arch D. Robison and Charles E. Leiserson. Cilk Plus. In Pavan Balaji, editor, *Programming Models for Parallel Computing*, chapter 13, pages 323–352. The MIT Press, 2015.
- [384] D. J. Rosenkrantz, R. E. Stearns, and P. M. Lewis. An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, 6(3):563–581, 1977.
- [385] Tim Roughgarden. *Algorithms Illuminated, Part 1: The Basics*. Soundlikeyourself Publishing, 2017.
- [386] Tim Roughgarden. *Algorithms Illuminated, Part 2: Graph Algorithms and Data Structures*. Soundlikeyourself Publishing, 2018.
- [387] Tim Roughgarden. *Algorithms Illuminated, Part 3: Greedy Algorithms and Dynamic Programming*. Soundlikeyourself Publishing, 2019.

- [388] Tim Roughgarden. *Algorithms Illuminated, Part 4: Algorithms for NP-Hard Problems*. Soundlikeyourself Publishing, 2020.
- [389] Salvador Roura. Improved master theorems for divide-and-conquer recurrences. *Journal of the ACM*, 48(2):170–205, 2001.
- [390] Y. A. Rozanov. *Probability Theory: A Concise Course*. Dover, 1969.
- [391] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, fourth edition, 2020.
- [392] S. Sahni and T. Gonzalez. P-complete approximation problems. *Journal of the ACM*, 23(3):555–565, 1976.
- [393] Peter Sanders, Kurt Mehlhorn, Martin Dietzfelbinger, and Roman Dementiev. *Sequential and Parallel Algorithms and Data Structures: The Basic Toolkit*. Springer, 2019.
- [394] Piotr Sankowski. Shortest paths in matrix multiplication time. In *Proceedings of the 13th Annual European Symposium on Algorithms*, pages 770–778, 2005.
- [395] Russel Schaffer and Robert Sedgwick. The analysis of heapsort. *Journal of Algorithms*, 15(1):76–100, 1993.
- [396] Tao B. Schardl, I-Ting Angelina Lee, and Charles E. Leiserson. Brief announcement: Open Cilk. In *Proceedings of the 30th Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 351–353, 2018.
- [397] A. Schönhage, M. Paterson, and N. Pippenger. Finding the median. *Journal of Computer and System Sciences*, 13(2):184–199, 1976.
- [398] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons, 1986.
- [399] Alexander Schrijver. Paths and flows—A historical survey. *CWI Quarterly*, 6(3):169–183, 1993.
- [400] Alexander Schrijver. On the history of the shortest paths problem. *Documenta Mathematica*, 17(1):155–167, 2012.
- [401] Robert Sedgwick. Implementing quicksort programs. *Communications of the ACM*, 21(10):847–857, 1978.
- [402] Robert Sedgwick and Kevin Wayne. *Algorithms*. Addison-Wesley, fourth edition, 2011.
- [403] Raimund Seidel. On the all-pairs-shortest-path problem in unweighted undirected graphs. *Journal of Computer and System Sciences*, 51(3):400–403, 1995.
- [404] Raimund Seidel and C. R. Aragon. Randomized search trees. *Algorithmica*, 16(4–5):464–497, 1996.
- [405] João Setubal and João Meidanis. *Introduction to Computational Molecular Biology*. PWS Publishing Company, 1997.
- [406] Clifford A. Shaffer. *A Practical Introduction to Data Structures and Algorithm Analysis*. Prentice Hall, second edition, 2001.
- [407] Jeffrey Shallit. Origins of the analysis of the Euclidean algorithm. *Historia Mathematica*, 21(4):401–419, 1994.
- [408] M. Sharir. A strong-connectivity algorithm and its applications in data flow analysis. *Computers and Mathematics with Applications*, 7(1):67–72, 1981.

- [409] David B. Shmoys. Computing near-optimal solutions to combinatorial optimization problems. In William Cook, László Lovász, and Paul Seymour, editors, *Combinatorial Optimization*, volume 20 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. American Mathematical Society, 1995.
- [410] Avi Shoshan and Uri Zwick. All pairs shortest paths in undirected graphs with integer weights. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science*, pages 605–614, 1999.
- [411] Victor Shoup. *A Computational Introduction to Number Theory and Algebra*. Cambridge University Press, second edition, 2009.
- [412] Julian Shun. *Shared-Memory Parallelism Can Be Simple, Fast, and Scalable*. Association for Computing Machinery and Morgan & Claypool, 2017.
- [413] Michael Sipser. *Introduction to the Theory of Computation*. Cengage Learning, third edition, 2013.
- [414] Steven S. Skiena. *The Algorithm Design Manual*. Springer, second edition, corrected printing, 2012.
- [415] Daniel D. Sleator and Robert E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences*, 26(3):362–391, 1983.
- [416] Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update rules. In *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*, pages 488–492, 1984.
- [417] Daniel D. Sleator and Robert E. Tarjan. Amortized efficiency of list update and paging rules. *Communications of the ACM*, 28(2):202–208, 1985.
- [418] Daniel D. Sleator and Robert E. Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3):652–686, 1985.
- [419] Michael Soltys-Kulinicz. *An Introduction to the Analysis of Algorithms*. World Scientific, third edition, 2018.
- [420] Joel Spencer. *Ten Lectures on the Probabilistic Method*, volume 64 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics, 1993.
- [421] Daniel A. Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004.
- [422] Gilbert Strang. *Introduction to Applied Mathematics*. Wellesley-Cambridge Press, 1986.
- [423] Gilbert Strang. *Linear Algebra and Its Applications*. Thomson Brooks/Cole, fourth edition, 2006.
- [424] Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 14(3):354–356, 1969.
- [425] T. G. Szymanski. A special case of the maximal common subsequence problem. Technical Report TR-170, Computer Science Laboratory, Princeton University, 1975.
- [426] Robert E. Tarjan. Depth first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2):146–160, 1972.
- [427] Robert E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, 22(2):215–225, 1975.

- [428] Robert E. Tarjan. A class of algorithms which require nonlinear time to maintain disjoint sets. *Journal of Computer and System Sciences*, 18(2):110–127, 1979.
- [429] Robert E. Tarjan. *Data Structures and Network Algorithms*. Society for Industrial and Applied Mathematics, 1983.
- [430] Robert E. Tarjan. Amortized computational complexity. *SIAM Journal on Algebraic and Discrete Methods*, 6(2):306–318, 1985.
- [431] Robert E. Tarjan. Class notes: Disjoint set union. COS 423, Princeton University, 1999. Available at <https://www.cs.princeton.edu/courses/archive/spr00/cs423/handout3.pdf>.
- [432] Robert E. Tarjan and Jan van Leeuwen. Worst-case analysis of set union algorithms. *Journal of the ACM*, 31(2):245–281, 1984.
- [433] George B. Thomas, Jr., Maurice D. Weir, Joel Hass, and Frank R. Giordano. *Thomas' Calculus*. Addison-Wesley, eleventh edition, 2005.
- [434] Mikkel Thorup. Faster deterministic sorting and priority queues in linear space. In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms*, pages 550–555, 1998.
- [435] Mikkel Thorup. Undirected single-source shortest paths with positive integer weights in linear time. *Journal of the ACM*, 46(3):362–394, 1999.
- [436] Mikkel Thorup. On RAM priority queues. *SIAM Journal on Computing*, 30(1):86–109, 2000.
- [437] Mikkel Thorup. High speed hashing for integers and strings. <http://arxiv.org/abs/1504.06804>, 2015.
- [438] Mikkel Thorup. Linear probing with 5-independent hashing. <http://arxiv.org/abs/1509.04549>, 2015.
- [439] Richard Tolimieri, Myoung An, and Chao Lu. *Mathematics of Multidimensional Fourier Transform Algorithms*. Springer, second edition, 1997.
- [440] P. van Emde Boas. Preserving order in a forest in less than logarithmic time and linear space. *Information Processing Letters*, 6(3):80–82, 1977.
- [441] P. van Emde Boas, R. Kaas, and E. Zijlstra. Design and implementation of an efficient priority queue. *Mathematical Systems Theory*, 10(1):99–127, 1976.
- [442] Charles Van Loan. *Computational Frameworks for the Fast Fourier Transform*. Society for Industrial and Applied Mathematics, 1992.
- [443] Benjamin Van Roy. A short proof of optimality for the MIN cache replacement algorithm. *Information Processing Letters*, 102(2–3):72–73, 2007.
- [444] Robert J. Vanderbei. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, 1996.
- [445] Virginia Vassilevska Williams. Multiplying matrices faster than Coppersmith-Winograd. In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing*, pages 887–898, 2012.
- [446] Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2001.
- [447] Rakesh M. Verma. General techniques for analyzing recursive algorithms with applications. *SIAM Journal on Computing*, 26(2):568–581, 1997.

- [448] Berthold Vöcking, Helmut Alt, Martin Dietzfelbinger, Rüdiger Reischuk, Christian Scheideler, Heribert Vollmer, and Dorothea Wager, editors. *Algorithms Unplugged*. Springer, 2011.
- [449] Antony F. Ware. Fast approximate Fourier transforms for irregularly spaced data. *SIAM Review*, 40(4):838–856, 1998.
- [450] Stephen Warshall. A theorem on boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.
- [451] Mark Allen Weiss. *Data Structures and Problem Solving Using C++*. Addison-Wesley, second edition, 2000.
- [452] Mark Allen Weiss. *Data Structures and Problem Solving Using Java*. Addison-Wesley, third edition, 2006.
- [453] Mark Allen Weiss. *Data Structures and Algorithm Analysis in C++*. Addison-Wesley, third edition, 2007.
- [454] Mark Allen Weiss. *Data Structures and Algorithm Analysis in Java*. Addison-Wesley, second edition, 2007.
- [455] Herbert S. Wilf. *Algorithms and Complexity*. A K Peters, second edition, 2002.
- [456] J. W. J. Williams. Algorithm 232 (HEAPSORT). *Communications of the ACM*, 7(6):347–348, 1964.
- [457] Ryan Williams. Faster all-pairs shortest paths via circuit complexity. *SIAM Journal on Computing*, 47(5):1965–1985, 2018.
- [458] David P. Williamson. *Network Flow Algorithms*. Cambridge University Press, 2019.
- [459] David P. Williamson and David B. Shmoys. *The Design of Approximation Algorithms*. Cambridge University Press, 2011.
- [460] Shmuel Winograd. On the algebraic complexity of functions. In *Actes du Congrès International des Mathématiciens*, volume 3, pages 283–288, 1970.
- [461] Yifan Xu, I-Ting Angelina Lee, and Kunal Agrawal. Efficient parallel determinacy race detection for two-dimensional dags. In *Proceedings of the 23rd ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming (PPoPP)*, pages 368–380, 2018.
- [462] Chee Yap. A real elementary approach to the master recurrence and generalizations. In M. Ogiwara and J. Tarui, editors, *Theory and Applications of Models of Computation. TAMC 2011*, volume 6648 of *Lecture Notes in Computer Science*, pages 14–26. Springer, 2011.
- [463] Yinyu Ye. *Interior Point Algorithms: Theory and Analysis*. John Wiley & Sons, 1997.
- [464] Neal E. Young. Online paging and caching. In *Encyclopedia of Algorithms*, pages 1457–1461. Springer, 2016.
- [465] Raphael Yuster and Uri Zwick. Answering distance queries in directed graphs using fast matrix multiplication. In *Proceedings of the 46th Annual Symposium on Foundations of Computer Science*, pages 389–396, 2005.
- [466] Jisheng Zhao and Vivek Sarkar. The design and implementation of the Habanero-Java parallel programming language. In *Symposium on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA)*, pages 185–186, 2011.
- [467] Uri Zwick. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *Journal of the ACM*, 49(3):289–317, 2002.
- [468] Daniel Zwillinger, editor. *CRC Standard Mathematical Tables and Formulae*. Chapman & Hall/CRC Press, 31st edition, 2003.

Index

This index uses the following conventions. Numbers are alphabetized as if spelled out; for example, “2-3-4 tree” is indexed as if it were “two-three-four tree.” When an entry refers to a place other than the main text, the page number is followed by a tag: ex. for exercise, pr. for problem, fig. for figure, and n. for footnote. A tagged page number often indicates the first page of an exercise or problem, which is not necessarily the page on which the reference actually appears.

- $\alpha(n)$, 533
- α -strongly convex function, 1041
- β -smooth function, 1041
- δ
 - (shortest-path distance), 558
 - (shortest-path weight), 604
- ϕ (golden ratio), 69
- $\hat{\phi}$ (conjugate of the golden ratio), 69
- $\phi(n)$ (Euler’s phi function), 920
- π
 - (predecessor in a breadth-first tree), 555
 - (predecessor in a shortest-paths tree), 608
- $\rho(n)$ -approximation algorithm, 1104, 1120
- o -notation, 60
- O -notation, 50, 54–55
- O' -notation, 73 pr.
- \widetilde{O} -notation, 73 pr.
- ω -notation, 61
- Ω -notation, 51, 54 fig., 55–56
- ∞ -notation, 73 pr.
- $\widetilde{\Omega}$ -notation, 73 pr.
- Θ -notation, 33, 51, 54 fig., 56
- $\widetilde{\Theta}$ -notation, 73 pr.
- $\{ \}$ (set), 1153
- \in (set member), 1153
- \notin (not a set member), 1153
- \emptyset
 - (empty language), 1052
 - (empty set), 1153
- \subseteq (subset), 1154
- \subset (proper subset), 1154
- $:$ (such that), 54 n., 1154
- \cap (set intersection), 1154
- \cup (set union), 1154
- $-$ (set difference), 1154
- $||$
 - (flow value), 672
 - (length of a string), 959
 - (set cardinality), 1156
- \times (Cartesian product), 1157
- $\langle \rangle$
 - (sequence), 1162
 - (standard encoding), 1052
- $:$ (subarray), 19, 23
- $[a, b]$ (closed interval), 1157
- (a, b) (open interval), 1157
- $[a, b)$ or $(a, b]$ (half-open interval), 1157
- $\binom{n}{k}$ (choose), 1180
- $\| \|$ (euclidean norm), 1219
- $!$ (factorial), 67–68
- $\lceil \rceil$ (ceiling), 63
- $\lfloor \rfloor$ (floor), 63

- ∂ (partial derivative), 1023
- \sum (sum), 1140
- \prod (product), 1144
- \rightarrow (adjacency relation), 1165
- \rightsquigarrow (reachability relation), 1165
- \wedge (AND), 659, 1065
- \parallel (concatenation), 291
- \neg (NOT), 1065
- \vee (OR), 659, 1065
- \lll (left shift), 305
- \ggg (logical right shift), 285
- \oplus
 - (group operator), 917
 - (semiring operator), 651 n.
 - (symmetric difference), 706
- \otimes
 - (convolution operator), 880
 - (semiring operator), 651 n.
- $*$ (closure operator), 1052
- \mid (divides relation), 904
- \nmid (does-not-divide relation), 904
- \equiv (mod n) (equivalent, modulo n), 64
- $\not\equiv$ (mod n) (not equivalent, modulo n), 64
- $[a]_n$ (equivalence class modulo n), 905
- $+_n$ (addition modulo n), 917
- \cdot_n (multiplication modulo n), 917
- $\left(\frac{a}{p}\right)$ (Legendre symbol), 954 pr.
- ε (empty string), 959, 1052
- \sqsubset (prefix relation), 959
- \sqsupset (suffix relation), 959
- $//$ (comment symbol), 22
- \gg (much-greater-than relation), 533
- \ll (much-less-than relation), 761
- \leq_P (polynomial-time reducibility relation), 1062, 1071 ex.
- AA-tree, 358
- abelian group, 917
- absent child, 1173
- absolutely convergent series, 1140
- absorption laws for sets, 1155
- abstract problem, 1048
- abuse of asymptotic notation, 55, 59–60
- acceptable pair of integers, 950
- acceptance
 - by an algorithm, 1053
 - by a finite automaton, 968
- accepting state, 967
- accounting method, 453–456
 - for binary counters, 455
 - for dynamic tables, 463
 - for stack operations, 454–455
- Ackermann's function, 544
- activity-selection problem, 418–425
- acyclic graph, 1166
- ADD-BINARY-INTEGERS, 25 ex.
- add instruction, 26
- addition
 - of matrices, 1217
 - modulo n ($+_n$), 917
 - of polynomials, 877
- additive group modulo n , 918
- addressing, open, *see* open-address hash table
- ADD-SUBARRAY, 783 pr.
- adjacency-list representation, 550–551
 - replaced by a hash table, 553 ex.
- adjacency-matrix representation, 551–552
- adjacency relation (\rightarrow), 1165
- adjacent vertices, 1165
- Advanced Encryption Standard (AES), 291
- adversary, 204, 286, 805, 807, 941
- AES, 291
- aggregate analysis, 449–453
 - for binary counters, 451–453
 - for breadth-first search, 558
 - for depth-first search, 566–567
 - for Dijkstra's algorithm, 623–624
 - for disjoint-set data structures, 525–526, 527 ex.
 - for dynamic tables, 462–463
 - for the Knuth-Morris-Pratt algorithm, 977–978
 - for Prim's algorithm, 597
 - for rod cutting, 370
 - for shortest paths in a dag, 617
 - for stack operations, 449–451
- aggregate flow, 864
- Akra-Bazzi recurrence, 115–119
 - solving by Akra-Bazzi method, 117–118
- algorithm, 1–1226
 - analysis of, 25–34
 - approximation, 1104–1136
 - compare-exchange, 222 pr.
 - correctness of, 6
 - decision, 1053

- algorithm, *continued*
 - deterministic, 135
 - lookahead, 815 *ex.*
 - nondeterministic, 765
 - oblivious, 222 *pr.*
 - offline, 791
 - online, *see* online algorithm
 - origin of word, 48
 - parallel, *see* parallel algorithm
 - push-relabel, 702
 - randomized, *see* randomized algorithm
 - recursive, 34
 - reduction, 1046, 1062
 - running time of, 29
 - scaling, 641 *pr.*, 699 *pr.*
 - streaming, 818
 - as a technology, 13
 - verification, 1058
- algorithmic recurrence, 77–78
- ALLOCATE-NODE, 506
- all-pairs shortest paths, 605, 646–669
 - in dynamic graphs, 669
 - in ϵ -dense graphs, 668 *pr.*
 - Floyd-Warshall algorithm for, 655–659
 - Johnson’s algorithm for, 662–667
 - by matrix multiplication, 648–655, 668–669
 - by repeated squaring, 652–653
- α -balanced, 472 *pr.*
- $\alpha(n)$, 533
- α -strongly convex function, 1041
- alphabet, 967, 1052
- alternating path, 705
- amortized analysis, 448–475
 - by accounting method, 453–456
 - by aggregate analysis, 370, 449–453
 - for breadth-first search, 558
 - for depth-first search, 566–567
 - for Dijkstra’s algorithm, 623–624
 - for disjoint-set data structures, 525–526, 527 *ex.*, 531 *ex.*, 534–540, 541 *ex.*
 - for dynamic tables, 460–471
 - for the Knuth-Morris-Pratt algorithm, 977–978
 - for making binary search dynamic, 472 *pr.*
 - by potential method, 456–460
 - for Prim’s algorithm, 597
 - for restructuring red-black trees, 473 *pr.*
 - for shortest paths in a dag, 617
 - for stacks on secondary storage, 517 *pr.*
 - for weight-balanced trees, 472 *pr.*
- amortized cost
 - in the accounting method, 453
 - in aggregate analysis, 449
 - in the potential method, 456
- amortized progress, 1028
- analysis of algorithms, 25–34
 - see also* amortized analysis, competitive analysis, probabilistic analysis
- ancestor, 1172
 - lowest common, 543 *pr.*
- AND function (\wedge), 659, 1065
- AND gate, 1065
- and, in pseudocode, 24
- antiparallel edges, 673–674
- antisymmetric relation, 1160
- approximation
 - by least squares, 841–845
 - of summation by integrals, 1150
- approximation algorithm, 1103–1136
 - for bin packing, 1131 *pr.*
 - for MAX-CNF satisfiability, 1124 *ex.*
 - for maximum clique, 1131 *pr.*
 - for maximum matching, 1132 *pr.*
 - for maximum spanning tree, 1134 *pr.*
 - for maximum-weight cut, 1124 *ex.*
 - for MAX-3-CNF satisfiability, 1120–1121
 - for parallel machine scheduling, 1133 *pr.*
 - randomized, 1120
 - for set cover, 1115–1119
 - for subset sum, 1124–1130
 - for traveling-salesperson problem, 1109–1115
 - for vertex cover, 1106–1109, 1121–1124
 - for weighted set cover, 1132 *pr.*
 - for 0-1 knapsack problem, 1134 *pr.*
- approximation error, 842
- approximation ratio, 1104, 1120
- approximation scheme, 1105
- APPROX-MIN-WEIGHT-VC, 1123
- APPROX-SUBSET-SUM, 1128
- APPROX-TSP-TOUR, 1111
- APPROX-VERTEX-COVER, 1107
- arbitrage, 641 *pr.*
- arc, *see* edge
- argument of a function, 1161–1162
- arithmetic instructions, 26

- arithmetic, modular, 64, 916–923
- arithmetic series, 1141
- arithmetic with infinities, 611
- arm in a disk drive, 498
- array
 - indexing into, 22–23, 26 n., 252
 - inversion in, 47 pr.
 - Monge, 123 pr.
 - passing as a parameter, 24
 - in pseudocode, 22–23
 - storage of, 26 n., 252
- articulation point, 582 pr.
- assignment
 - optimal, 723–739
 - satisfying, 1066, 1074
 - truth, 1066, 1073
- assignment problem, 723–739
- associative laws for sets, 1155
- associative operation, 917
- asymptotically larger, 62
- asymptotically nonnegative, 54
- asymptotically positive, 54
- asymptotically smaller, 62
- asymptotically tight bound, 56
- asymptotic lower bound, 55
- asymptotic notation, 53–63, 72 pr.
 - and graph algorithms, 548
 - and linearity of summations, 1141
- asymptotic running time, 49
- asymptotic upper bound, 54
- attribute
 - in clustering, 1006
 - in a graph, 552
 - of an object, 23
- augmentation of a flow, 678
- augmented primal linear program, 870
- augmenting data structures, 480–496
- augmenting path, 681–682, 705
 - widest, 700 pr.
- authentication, 309 pr., 938–939, 942
- automaton, 967–974
- auxiliary hash function, 295
- average-case running time, 32, 128
- AVL tree, 357 pr., 358
- back edge, 569, 573
- back substitution, 823
- balanced search tree
 - AA-trees, 358
 - AVL trees, 357 pr., 358
 - B-trees, 497–519
 - k -neighbor trees, 358
 - left-leaning red-black binary search trees, 358
 - red-black trees, 331–359
 - scapegoat trees, 358
 - splay trees, 359, 478
 - treaps, 358
 - 2-3-4 trees, 502, 518 pr.
 - 2-3 trees, 358, 519
 - weight-balanced trees, 358, 472 pr.
- balls and bins, 143–144, 1212 pr.
- base- a pseudoprime, 944
- base case
 - of a divide-and-conquer algorithm, 34, 76
 - of a recurrence, 41, 77–78
- base, in DNA, 393
- basis function, 841
- Bayes’s theorem, 1189
- BELLMAN-FORD, 612
- Bellman-Ford algorithm, 612–616
 - for all-pairs shortest paths, 647
 - in Johnson’s algorithm, 664–666
 - and objective functions, 632 ex.
 - to solve systems of difference constraints, 630–631
 - Yen’s improvement to, 640 pr.
- Bernoulli trial, 1196
 - and balls and bins, 143–144
 - in bucket sort analysis, 217
 - in finding prime numbers, 943
 - in randomized selection analysis, 232
 - and streaks, 144–150
- best-case running time, 34 ex.
- β -smooth function, 1041
- BFS, 556
 - see also* breadth-first search
- BIASED-RANDOM, 129 ex.
- biconnected component, 582 pr.
- big-oh notation (O), 50, 54–55
- big-omega notation (Ω), 51, 54 fig., 55–56
- bijective function, 1162
- binary character code, 431

- binary counter
 - analyzed by accounting method, 455
 - analyzed by aggregate analysis, 451–453
 - analyzed by potential method, 458–459
- binary entropy function, 1182
- binary gcd algorithm, 953 pr.
- binary heap, *see* heap
- binary logarithm (\lg), 66
- binary reflected Gray code, 471 pr.
- binary relation, 1158
- binary search, 44 ex.
 - with fast insertion, 472 pr.
 - in insertion sort, 45 ex.
 - in parallel merging, 777–778
 - in searching B-trees, 512 ex.
- binary search tree, 312–330
 - AA-trees, 358
 - AVL trees, 357 pr., 358
 - deletion from, 322–325, 326 ex.
 - with equal keys, 327 pr.
 - insertion into, 321–322
 - k -neighbor trees, 358
 - left-leaning red-black binary search trees, 358
 - maximum key of, 317–318
 - minimum key of, 317–318
 - optimal, 400–407
 - persistent, 355 pr.
 - predecessor in, 318–319
 - querying, 316–320
 - randomly built, 328 pr.
 - red-black trees, 331–359
 - right-converting of, 337 ex.
 - scapegoat trees, 358
 - searching, 316–317
 - for sorting, 326 ex.
 - splay trees, 359
 - successor in, 318–319
 - weight-balanced trees, 358
 - see also* red-black tree
- binary-search-tree property, 313–314
 - vs. min-heap property, 315 ex.
- binary tree, 1173
 - full, 433, 1174
 - number of different ones, 329 pr.
 - representation of, 265
 - see also* binary search tree
- binomial coefficient, 1181–1182
- binomial distribution, 1198–1201
 - and balls and bins, 143
 - in bucket sort analysis, 217
 - maximum value of, 1202 ex.
 - tails of, 1203–1210
- binomial expansion, 1181
- binomial theorem, 1181
- bin packing, 1131 pr.
- bipartite graph, 1167
 - complete, 716
 - corresponding flow network of, 694
 - d -regular, 716 ex., 740 pr.
 - matching in, 693–697, 704–743
- bipartite matching, 693–697, 704–743
- birthday paradox, 140–143
- bisection of a tree, 1177 pr.
- bitonic euclidean traveling-salesperson
 - problem, 407 pr.
- bitonic sequence, 644 pr.
- bitonic tour, 407 pr.
- bit operation, 904
 - in Euclid’s algorithm, 954 pr.
- bit-reversal permutation, 897
- bit vector, 274 ex.
- black-height, 332
- black vertex, 554, 564
- block
 - in a cache, 440, 802
 - on a disk, 499, 512 ex., 517 pr.
- blocking flow, 702
- blocking pair, 716
- block representation of matrices, 254
- block structure in pseudocode, 21–22
- body, 1032
- Boole’s inequality, 1190 ex.
- boolean combinational circuit, 1065
- boolean combinational element, 1065
- boolean connective, 1073
- boolean data type, 26
- boolean formula, 1043, 1060 ex., 1073–1074
- boolean function, 1182 ex.
- boolean operators, 24
- Borůvka’s algorithm, 603
- bottleneck spanning tree, 601 pr.
- bottleneck traveling-salesperson problem, 1115 ex.
- bottoming out, 76
- bottom of a stack, 254

- BOTTOM-UP-CUT-ROD, 369
- bottom-up method, for dynamic programming, 368
- bound
 - asymptotically tight, 56
 - asymptotic lower, 55
 - asymptotic upper, 54
 - on binomial coefficients, 1181–1182
 - on binomial distributions, 1201
 - polylogarithmic, 67
 - on the tails of a binomial distribution, 1203–1210
 - see also* lower bounds
- bounding a summation, 1145–1152
- box, nesting, 640 pr.
- B⁺-tree, 501
- branch instructions, 26
- breadth-first forest, 728
- breadth-first search, 554–563
 - in the Hopcroft-Karp algorithm, 711
 - in the Hungarian algorithm, 727–728
 - in maximum flow, 689–691
 - and shortest paths, 558–561, 605
 - similarity to Dijkstra's algorithm, 624, 625 ex.
- breadth-first tree, 555, 561
- bridge, 582 pr.
- B*-tree, 502 n.
- B-tree, 497–519
 - compared with red-black trees, 497, 503
 - creating, 505–506
 - deletion from, 513–516
 - full node in, 502
 - height of, 502–504
 - insertion into, 506–511
 - minimum degree of, 502
 - properties of, 501–504
 - searching, 504–505
 - splitting a node in, 506–508
 - 2-3-4 trees, 502
- B-TREE-CREATE, 506
- B-TREE-DELETE, 513
- B-TREE-INSERT, 508
- B-TREE-INSERT-NONFULL, 511
- B-TREE-SEARCH, 505, 512 ex.
- B-TREE-SPLIT-CHILD, 507
- B-TREE-SPLIT-ROOT, 509
- BUBBLESORT, 46 pr.
- bucket, 215
- bucket sort, 215–219
- BUCKET-SORT, 216
- BUILD-MAX-HEAP, 167
- BUILD-MAX-HEAP', 179 pr.
- BUILD-MIN-HEAP, 169
- Burrows-Wheeler transform (BWT), 1000 pr.
- butterfly operation, 894
- BWT (Burrows-Wheeler transform), 1000 pr.
- by, in pseudocode, 22
- cache, 27, 301, 440, 802
- cache block, 301, 440, 802
- cache hit, 440, 803
- cache line, *see* cache block
- cache miss, 440, 803
- cache obliviousness, 519
- caching
 - offline, 440–446
 - online, 802–815
- call
 - in a parallel computation, 753
 - of a subroutine, 26, 29 n.
 - by value, 23
- cancellation lemma, 886
- cancellation of flow, 679
- capacity
 - of a cut, 682
 - of an edge, 671
 - residual, 677, 681
 - of a vertex, 676 ex.
- capacity constraint, 672
- cardinality of a set ($| \cdot |$), 1156
- Carmichael number, 945, 953 ex.
- Cartesian product (\times), 1157
- Cartesian sum, 885 ex.
- Catalan numbers, 329 pr., 375
- CBC-MAC, 291, 306
- c -competitive, 793
- ceiling function ($\lceil \cdot \rceil$), 63
 - in recurrences, 116–117
- ceiling instruction, 26
- center of a cluster, 1008
- centralized scheduler, 759
- centroid of a cluster, 1009
- certain event, 1185

- certificate
 - in a cryptosystem, 942
 - for verification algorithms, 1058
- CHAINED-HASH-DELETE, 278
- CHAINED-HASH-INSERT, 278
- CHAINED-HASH-SEARCH, 278
- chaining, 277–281, 308 pr.
- changing variables, to solve a recurrence, 120 pr.
- character code, 431
- character data type, 26
- chess-playing program, 768–769
- child
 - in a binary tree, 1173
 - in a parallel computation, 753
 - in a rooted tree, 1172
- Chinese remainder theorem, 928–931
- chirp transform, 893 ex.
- choose $\binom{n}{k}$, 1180
- chord, 486 ex.
- Cilk, 750, 790
- ciphertext, 938
- circuit
 - boolean combinational, 1065
 - depth of, 894
 - for fast Fourier transform, 894–897
- CIRCUIT-SAT, 1067
- circuit satisfiability, 1064–1071
- circular, doubly linked list with a sentinel, 262
- circular linked list, 259
- class
 - complexity, 1054
 - equivalence, 1159
- classification of edges
 - in breadth-first search, 581 pr.
 - in depth-first search, 569–570, 571 ex.
- clause, 1075–1076
- clean area, 222 pr.
- climate change, 845
- clique, 1081
- CLIQUE, 1081
- clique problem
 - approximation algorithm for, 1131 pr.
 - NP-completeness of, 1081–1084
- closed convex body, 1032
- closed interval $([a, b])$, 1157
- closed semiring, 669
- closest-point heuristic, 1115 ex.
- closure
 - group property, 917
 - of a language $(*)$, 1052
 - transitive, *see* transitive closure
- cluster, 1008
 - for parallel computing, 748
- clustering, 1005–1013
 - Lloyd’s procedure for, 1011–1013
 - primary, 303
- CNF (conjunctive normal form), 1043, 1076
- CNF satisfiability, 1124 ex.
- coarsening leaves of recursion
 - in merge sort, 45 pr.
 - in quicksort, 198 ex.
 - when recursively spawning, 764
- code, 431–432
 - Huffman, 431–439
- codeword, 432
- codomain, 1161
- coefficient
 - binomial, 1181
 - of a polynomial, 65, 877
- coefficient representation, 879
 - and fast multiplication, 882–884
- cofactor, 1221
- coin changing, 446 pr.
- coin flipping, 131–132
- collection of sets, 1156
- collision, 275
 - resolution by chaining, 277–281
 - resolution by open addressing, 293–301
- collision-resistant hash function, 941
- coloring, 425 ex., 1100 pr., 1176 pr.
- color, of a red-black-tree node, 331
- column-major order, 222 pr., 253
- column rank, 1220
- columnsort, 222 pr.
- column vector, 1215
- combination, 1180
- combinational circuit, 1065
- combinational element, 1065
- combine step, in divide-and-conquer, 34, 76
- comment, in pseudocode $(//)$, 22
- commodity, 864
- common divisor, 906
 - greatest, *see* greatest common divisor
- common multiple, 916 ex.
- common subexpression, 894

- common subsequence, 394
 - longest, 393–399
- commutative laws for sets, 1154
- commutative operation, 917
- compact list, 269 pr.
- COMPACT-LIST-SEARCH, 269 pr.
- COMPACT-LIST-SEARCH', 270 pr.
- COMPARE-EXCHANGE, 222 pr.
- COMPARE-EXCHANGE-INSERTION-SORT, 223 pr.
- compare-exchange operation, 222 pr.
- comparison sort, 205
 - and binary search trees, 315 ex.
 - randomized, 219 pr.
 - and selection, 241
- compatible activities, 418
- compatible matrices, 1218
- competitive analysis, 792
- competitive ratio, 793
 - expected, 808
 - unbounded, 804
- complement
 - of an event, 1186
 - of a graph, 1085
 - of a language, 1052
 - Schur, 825, 839
 - of a set, 1155
- complementary slackness, 873 pr.
- complete graph, 1167
 - bipartite, 716
- complete k -ary tree, 1174
 - see also* heap
- completeness of a language, 1072 ex.
- complete step, 759
- completion time, 446 pr., 816 pr., 1133 pr.
- COMPLETION-TIME-SCHEDULE, 817 pr.
- complexity class, 1054
 - co-NP, 1059
 - NP, 1043, 1058, 1060 ex.
 - NPC, 1044, 1063
 - P, 1043, 1050, 1054, 1055 ex.
- complexity measure, 1054
- complex numbers
 - inverting matrices of, 838 ex.
 - multiplication of, 90 ex.
- complex root of unity, 885
 - interpolation at, 891–892
- component
 - biconnected, 582 pr.
 - connected, 1166
 - strongly connected, 1166
- component graph, 576
- composite number, 905
 - witness to, 946
- composition
 - of logarithms, 66
 - of parallel traces, 762 fig.
- compression
 - by Huffman code, 431–439
 - of images, 412 pr.
- compulsory miss, 440
- computational depth, *see* span
- computational problem, 5–6
- computation dag, 754 n.
- COMPUTE-LCP, 993
- COMPUTE-PREFIX-FUNCTION, 978
- COMPUTE-SUFFIX-ARRAY, 988
- COMPUTE-TRANSITION-FUNCTION, 974
- concatenation
 - of languages, 1052
 - operator (\parallel), 291
 - of strings, 959
- concrete problem, 1049
- conditional branch instruction, 26
- conditional independence, 1190 ex.
- conditional probability, 1187, 1189
- configuration, 1068
- conjugate of the golden ratio ($\hat{\phi}$), 69, 70 ex.
- conjugate transpose, 838 ex.
- conjunctive normal form, 1043, 1076
- connected component, 1166
 - identified using depth-first search, 572 ex.
 - identified using disjoint-set data structures, 521–523
- CONNECTED-COMPONENTS, 522
- connected graph, 1166
- connective, 1073
- co-NP (complexity class), 1059
- conquer step, in divide-and-conquer, 34, 76
- conservation of flow, 672
- consistency
 - of literals, 1082
 - sequential, 756
- constrained gradient descent, 1032–1034

- constraint
 - difference, 627
 - equality, 632 *ex.*
 - linear, 851, 853–854
 - nonnegativity, 854
- constraint graph, 628–630
- contain, in a path, 1165
- continuous master theorem, 112
 - proof of, 107–115
- continuous uniform probability distribution, 1187
- contraction
 - of a dynamic table, 465–470
 - of an undirected graph by an edge, 1168
- contraction algorithm, 701 *pr.*
- control instructions, 26
- convergence property, 611, 634–635
- convergent series, 1140
- converting binary to decimal, 910 *ex.*
- convex body, 1032
- convex function, 1025–1027, 1194
 - α -strongly convex, 1041
- convex set, 675 *ex.*
- convolution (\otimes), 880
- convolution theorem, 892
- copy instruction, 26
- correctness of an algorithm, 6
- corresponding flow network for bipartite matching, 694
- countably infinite set, 1156
- counter, *see* binary counter
- counting, 1178–1184
 - probabilistic, 153 *pr.*
- counting sort, 208–211
 - in computing suffix arrays, 992
 - in radix sort, 213
- COUNTING-SORT, 209
- coupon collector's problem, 144
- cover
 - path, 698 *pr.*
 - by a subfamily, 1116
 - vertex, 1084, 1106
- credit, 453
- critical edge, 690
- critical path
 - in a dag, 619
 - of a PERT chart, 617
 - of a task-parallel trace, 757
- cross a cut, 587, 701 *pr.*
- cross edge, 569
- cryptographic hash function, 291
- cryptosystem, 936–942
- cubic spline, 847 *pr.*
- currency exchange, 641 *pr.*
- curve fitting, 841–845
- cut
 - capacity of, 682
 - of a flow network, 682–685
 - global, 701 *pr.*
 - minimum, 682
 - net flow across, 682
 - of an undirected graph, 587
 - weight of, 1124 *ex.*
- CUT-ROD, 366
- cycle of a graph, 1165–1166
 - hamiltonian, 1043, 1056, 1085–1090
 - minimum mean-weight, 642 *pr.*
 - negative-weight, *see* negative-weight cycle
 - and shortest paths, 607–608
- cycle cover, 741 *pr.*
- cyclic group, 932
- dag, *see* directed acyclic graph
- DAG-SHORTEST-PATHS, 617
- d -ary heap, 179 *pr.*
 - in shortest-paths algorithms, 668 *pr.*
- data-movement instructions, 26
- data-parallel model, 789
- data science, 14–15
- data structure, 9, 249–359, 477–545
 - AA-trees, 358
 - augmentation of, 480–496
 - AVL trees, 357 *pr.*, 358
 - binary search trees, 312–330
 - bit vectors, 274 *ex.*
 - B-trees, 497–519
 - deques, 258 *ex.*
 - dictionaries, 249
 - direct-address tables, 273–275
 - for disjoint sets, 520–545
 - for dynamic graphs, 479
 - dynamic sets, 249–251
 - dynamic trees, 478
 - exponential search trees, 226, 478
 - Fibonacci heaps, 478
 - fusion trees, 226, 478

- data structure, *continued*
 - hash tables, 275–282
 - heaps, 161–181
 - interval trees, 489–495
 - k -neighbor trees, 358
 - left-leaning red-black binary search trees, 358
 - linked lists, 258–264
 - order-statistic trees, 480–486
 - persistent, 355 pr., 478
 - potential of, 456
 - priority queues, 172–178
 - queues, 254, 256–257
 - radix trees, 327 pr.
 - red-black trees, 331–359
 - rooted trees, 265–268
 - scapegoat trees, 358
 - on secondary storage, 498–501
 - skip lists, 359
 - splay trees, 359, 478
 - stacks, 254–255
 - treaps, 358
 - 2-3-4 trees, 502, 518 pr.
 - 2-3 trees, 358, 519
 - van Emde Boas trees, 478
 - weight-balanced trees, 358
- data type, 26
- decision by an algorithm, 1053
- decision problem, 1045, 1049
 - and optimization problems, 1045
- decision tree, 206–207, 219 pr.
- decision variable, 851
- DECREASE-KEY, 173
- decrementing, 22
- decryption, 936
- default vertex labeling, 724
- degree
 - minimum, of a B-tree, 502
 - of a node, 1173
 - of a polynomial, 65, 877
 - of a vertex, 1165
- degree-bound, 877
- DELETE, 250
- DELETE-LARGER-HALF, 460 ex.
- deletion
 - from binary search trees, 322–325, 326 ex.
 - from B-trees, 513–516
 - from chained hash tables, 278
 - from direct-address tables, 274
 - from dynamic tables, 465–470
 - from hash tables with linear probing, 302–303
 - from heaps, 178 ex.
 - from interval trees, 491
 - from linked lists, 261
 - from open-address hash tables, 294–295
 - from order-statistic trees, 484–485
 - from queues, 256
 - from red-black trees, 346–355
 - from stacks, 254
- DeMorgan's laws
 - for propositional logic, 1078
 - for sets, 1155, 1158 ex.
- dense graph, 549
 - ϵ -dense, 668 pr.
- dense matrix, 81
- density
 - of prime numbers, 943
 - of a rod, 372 ex.
- dependence
 - and indicator random variables, 131
 - linear, 1220
 - see also* independence
- depth
 - average, of a node in a randomly built binary search tree, 328 pr.
 - of a circuit, 894
 - of a node in a rooted tree, 1173
 - of quicksort recursion tree, 191 ex.
 - of a stack, 202 pr.
- depth-determination problem, 542 pr.
- depth-first forest, 564
- depth-first search, 563–572
 - in finding articulation points, bridges, and biconnected components, 582 pr.
 - in finding strongly connected components, 576–581
 - in the Hopcroft-Karp algorithm, 711
 - in topological sorting, 573–576
- depth-first tree, 564
- deque, 258 ex.
- DEQUEUE, 257
- derivative of a series, 1142
- descendant, 1172
- destination vertex, 605
- det, 1221

- determinacy race, 765–768
- determinant, 1221
- deterministic algorithm, 135
 - parallel, 765
- DETERMINISTIC-SEARCH, 154 pr.
- DFS, 565
 - see also* depth-first search
- DFS-VISIT, 565
- DFT, 888
- diagonal matrix, 1215
- diameter
 - of a network, 646
 - of a tree, 563 ex.
- dictionary, 249
- difference
 - of sets ($-$), 1154
 - symmetric, 706
- difference constraints, 626–632
- differentiation of a series, 1142
- digital signature, 938
- digraph, *see* directed graph
- DIJKSTRA, 620
- Dijkstra's algorithm, 620–626
 - for all-pairs shortest paths, 646, 666
 - with edge weights in a range, 626 ex.
 - implemented with a Fibonacci heap, 623–624
 - implemented with a min-heap, 623
 - with integer edge weights, 625–626 ex.
 - in Johnson's algorithm, 664
 - similarity to breadth-first search, 624, 625 ex.
 - similarity to Prim's algorithm, 624
- d -independent family of hash functions, 288
- DIRECT-ADDRESS-DELETE, 274
- direct addressing, 273–275
- DIRECT-ADDRESS-INSERT, 274
- DIRECT-ADDRESS-SEARCH, 274
- direct-address table, 273–275
- directed acyclic graph (dag), 1167
 - and back edges, 573
 - and component graphs, 578
 - and hamiltonian paths, 1060 ex.
 - longest simple path in, 407 pr.
 - for representing a parallel computation, 754
 - single-source shortest-paths algorithm for, 616–619
 - topological sort of, 573–576
- directed equality subgraph, 727
- directed graph, 1164
 - all-pairs shortest paths in, 646–669
 - constraint graph, 628
 - Euler tour of, 583 pr., 1043
 - hamiltonian cycle of, 1043
 - incidence matrix of, 553 ex.
 - and longest paths, 1042
 - path cover of, 698 pr.
 - PERT chart, 617, 619 ex.
 - semiconnected, 581 ex.
 - shortest path in, 604
 - single-source shortest paths in, 604–645
 - singly connected, 572 ex.
 - square of, 553 ex.
 - transitive closure of, 659
 - transpose of, 553 ex.
 - universal sink in, 553 ex.
 - see also* directed acyclic graph, graph, network
- directed version of an undirected graph, 1166
- dirty area, 222 pr.
- discovered vertex, 554, 564
- discovery time, 565
- discrete Fourier transform, 888
- discrete logarithm, 933
- discrete logarithm theorem, 933
- discrete probability distribution, 1186
- discrete random variable, 1191–1196
- disjoint-set data structure, 520–545
 - analysis of, 534–540
 - in connected components, 521–523
 - in depth determination, 542 pr.
 - disjoint-set-forest implementation of, 527–531
 - in Kruskal's algorithm, 593
 - linear-time special case of, 545
 - linked-list implementation of, 523–527
 - lower bound for, 545
 - in offline lowest common ancestors, 543 pr.
 - in offline minimum, 541 pr.
- disjoint-set forest, 527–531
 - analysis of, 534–540
 - rank properties of, 533–534, 540 ex.
 - see also* disjoint-set data structure
- disjoint sets, 1156
- disjunctive normal form, 1078

- disk drive, 498–500
 - see also* secondary storage
- DISK-READ, 500
- DISK-WRITE, 500
- dissimilarity, 1006
- distance
 - edit, 409 pr.
 - Manhattan, 244 pr.
 - of a shortest path (δ), 558
- distributed memory, 748
- distribution
 - binomial, *see* binomial distribution
 - continuous uniform, 1187
 - discrete, 1186
 - geometric, *see* geometric distribution
 - of inputs, 128, 134
 - of prime numbers, 943
 - probability, 218 ex., 1185
 - uniform, 1186
- distributive laws for sets, 1155
- divergent series, 1140
- divide-and-conquer method, 34, 76
 - analysis of, 39–41, 90–119
 - for binary search, 44 ex.
 - for conversion of binary to decimal, 910 ex.
 - for fast Fourier transform, 888–891, 895
 - for matrix inversion, 834–837
 - for matrix multiplication, 81–90, 770–775, 783 pr.
 - for merge sort, 34–44, 775–782
 - for multiplication, 899 pr.
 - for quicksort, 182–204
 - relation to dynamic programming, 362
 - for selection, 230–243
 - solving recurrences for, 90–119
 - for Strassen’s algorithm, 85–90, 773–774
- divide instruction, 26
- divides relation (\mid), 904
- divide step, in divide-and-conquer, 34, 76
- division method, 284, 292 ex.
- division theorem, 905
- divisor, 904
 - common, 906
 - see also* greatest common divisor
- DNA, 6, 393–394, 409 pr.
- DNF (disjunctive normal form), 1078
- does-not-divide relation (\nmid), 904
- Dog River, 717
- dolphins, allowing to vote, 850
- domain, 1161
- double hashing, 295–297, 301 ex.
- doubly linked list, 258–259, 264 ex.
 - circular, with a sentinel, 262
- downto**, in pseudocode, 22
- d -regular graph, 716 ex., 740 pr.
- driving function, 101
- duality, 734, 866–873, 874 pr.
 - weak, 868–869, 874 pr.
- dual linear program, 866
- dummy key, 400
- dynamic graph, 523
 - all-pairs shortest paths algorithms for, 669
 - data structures for, 479
 - minimum-spanning-tree algorithm for, 599 ex.
 - transitive closure of, 667 pr., 669
- dynamic graph algorithm, 817
- dynamic multiset, 460 ex.
- dynamic order statistics, 480–486
- dynamic-programming method, 362–416
 - for activity selection, 424 ex.
 - for all-pairs shortest paths, 648–659
 - for bitonic euclidean traveling-salesperson problem, 407 pr.
 - bottom-up, 368
 - for breaking a string, 412 pr.
 - compared with greedy method, 384–385, 393 ex., 421, 426–430
 - for edit distance, 409 pr.
 - elements of, 382–393
 - for Floyd-Warshall algorithm, 655–659
 - for inventory planning, 414 pr.
 - for longest common subsequence, 393–399
 - for longest palindrome subsequence, 407 pr.
 - for longest simple path in a weighted directed acyclic graph, 407 pr.
 - for matrix-chain multiplication, 373–382 and memoization, 390–392
 - for optimal binary search trees, 400–407
 - optimal substructure in, 382–387
 - overlapping subproblems in, 387–390
 - for printing neatly, 408 pr.
 - reconstructing an optimal solution in, 390
 - relation to divide-and-conquer, 362
 - for rod cutting, 363–373
 - for seam carving, 412 pr.

- dynamic-programming method, *continued*
 - for signing free agents, 414 pr.
 - top-down with memoization, 368
 - for transitive closure, 659–661
 - for Viterbi algorithm, 411 pr.
 - for the 0-1 knapsack problem, 430 ex.
- dynamic set, 249–251
 - see also* data structure
- dynamic table, 460–471
 - analyzed by accounting method, 463
 - analyzed by aggregate analysis, 462–463
 - analyzed by potential method, 463–470
 - load factor of, 461
- dynamic tree, 478
- $E[\]$, *see* expected value
- e (base of the natural logarithm), 65
- edge, 1164
 - antiparallel, 673–674
 - attributes of, 552
 - back, 569
 - bridge, 582 pr.
 - capacity of, 671
 - classification in breadth-first search, 581 pr.
 - classification in depth-first search, 569–570, 571 ex.
 - critical, 690
 - cross, 569
 - forward, 569
 - light, 587
 - negative-weight, 606–607
 - residual, 678
 - safe, 587
 - tree, 561, 564, 569
 - weight of, 551
- edge connectivity, 692 ex.
- edge set, 1164
- edit distance, 409 pr.
- Edmonds-Karp algorithm, 689–691
- elementary event, 1185
- elementary insertion, 461
- element of a set (\in), 1153
- ellipsoid algorithm, 857
- elliptic-curve factorization method, 956
- elseif**, in pseudocode, 22 n.
- else**, in pseudocode, 22
- empty language (\emptyset), 1052
- empty set (\emptyset), 1153
- empty set laws, 1154
- empty stack, 255
- empty string (ε), 959, 1052
- empty tree, 1173
- encoding of problem instances, 1049–1052
- encryption, 936
- endpoint of an interval, 489
- ENQUEUE, 257
- entering a vertex, 1164
- entropy function, 1182
- epoch, 805
- ϵ -dense graph, 668 pr.
- ϵ -universal family of hash functions, 287, 292 ex.
- equality
 - of functions, 1162
 - linear, 853
 - of sets, 1153
- equality constraint, 632 ex.
- equality subgraph, 724
 - directed, 727
- equations and asymptotic notation, 58–59
- equivalence class, 1159
 - modulo n ($[a]_n$), 905
- equivalence, modular ($= (\bmod n)$), 64
- equivalence relation, 1159
- error bound, 1027
- error**, in pseudocode, 24
- escape problem, 697 pr.
- EUCLID, 912
- Euclid's algorithm, 911–916, 954 pr.
- euclidean norm ($\| \ \|$), 1219
- Euler's constant, 921
- Euler's phi function, 920
- Euler's theorem, 932, 953 ex.
- Euler tour, 583 pr., 740 pr.
 - and hamiltonian cycles, 1043
- evaluation of a polynomial, 46 pr., 879, 884 ex.
 - derivatives of, 900 pr.
 - at multiple points, 900 pr.
- event, 1185
- event-driven simulation, 173, 181
- EXACT-SUBSET-SUM, 1125
- example, in clustering, 1006
- exclusion and inclusion, 1158 ex.
- execute a subroutine, 29 n.
- expansion of a dynamic table, 461–465
- expectation, *see* expected value

- expected competitive ratio, 808
- expected running time, 32, 129
- expected value, 1192–1194
 - of a binomial distribution, 1198
 - of a geometric distribution, 1197
 - of an indicator random variable, 130
- explored edge, 565
- exponential function, 65–66
- exponential search tree, 226, 478
- exponentiation
 - of logarithms, 66
 - modular, 934–935
- exponentiation instruction, 27
- EXTENDED-BOTTOM-UP-CUT-ROD, 372
- EXTENDED-EUCLID, 914
- EXTEND-SHORTEST-PATHS, 650
- external node, 1172
- external path length, 1175 ex.
- extracting the maximum key
 - from d -ary heaps, 179 pr.
 - from max-heaps, 174
- extracting the minimum key
 - from Young tableaux, 179 pr.
- EXTRACT-MAX, 173–174
- EXTRACT-MIN, 173
- factor, 904
 - twiddle, 891
- factorial function (!), 67–68
- factorization, 956
 - unique, 909
- failure, in a Bernoulli trial, 1196
- fair coin, 1186
- family of hash functions, 286–288, 292 ex.
- fan-out, 1066
- Farkas’s lemma, 869
- FASTER-APSP, 653, 655 ex.
- fast Fourier transform (FFT), 877–902
 - circuit for, 894–897
 - multidimensional, 899 pr.
 - recursive implementation of, 888–891
 - using modular arithmetic, 901 pr.
- feasibility problem, 627, 873 pr.
- feasible linear program, 854
- feasible region, 854
- feasible solution, 627, 854
- feasible vertex labeling, 724, 742 pr.
- feature vector, 1006
- Fermat’s theorem, 932
- FFT, 890
 - see also* fast Fourier transform
- FFTW, 902
- FIB, 751
- Fibonacci heap, 478
 - in Dijkstra’s algorithm, 623–624
 - in Johnson’s algorithm, 666
 - in Prim’s algorithm, 597
- Fibonacci numbers, 69, 70 ex., 121 pr.
 - computation of, 750–753, 954 pr.
- FIFO, *see* first-in, first-out; queue
- final-state function, 968
- FIND-AUGMENTING-PATH, 738
- FIND-DEPTH, 542 pr.
- find path, 528
- FIND-POM, 496 pr.
- FIND-SET, 521
 - disjoint-set-forest implementation of, 530, 544
 - linked-list implementation of, 523
- FIND-SPLIT-POINT, 778
- finished vertex, 564
- finish time
 - in activity selection, 418
 - in depth-first search, 565
 - and strongly connected components, 578
- finite automaton, 967–975
- FINITE-AUTOMATON-MATCHER, 971
- finite group, 917
- finite sequence, 1162
- finite set, 1156
- finite sum, 1140
- first-fit heuristic, 1131 pr.
- first-in, first-out (FIFO), 254, 803–804, 814 ex.
 - implemented with a priority queue, 178 ex.
 - see also* queue
- fixed-length code, 432
- floating-point data type, 26
- floor function ($\lfloor \rfloor$), 63
 - in recurrences, 116–117
- floor instruction, 26
- flow, 671–676
 - aggregate, 864
 - augmentation of, 678
 - blocking, 702
 - cancellation of, 679
 - integer-valued, 695

- flow, *continued*
 - net, across a cut, 682
 - value of, 672
- flow conservation, 672
- flow network, 671–676
 - corresponding to a bipartite graph, 694
 - cut of, 682–685
 - with multiple sources and sinks, 674
- FLOYD-WARSHALL, 657
- FLOYD-WARSHALL', 661 ex.
- Floyd-Warshall algorithm, 655–659, 661–662 ex.
- flying cars, highways for, 850
- FORD-FULKERSON, 686
- Ford-Fulkerson method, 676–693
- FORD-FULKERSON-METHOD, 676
- FORESEE, 797
- forest, 1167, 1169
 - breadth-first, 728
 - depth-first, 564
 - disjoint-set, 527–531
- for**, in pseudocode, 22
 - and loop invariants, 21 n.
- fork-join parallelism, 749–770
 - see also* parallel algorithm
- fork-join scheduling, 759–761, 769 ex.
- formal power series, 121 pr.
- formula satisfiability, 1073–1076
- forward edge, 569
- forward substitution, 822–823
- Fourier transform, *see* discrete Fourier transform
- transform, fast Fourier transform
- fractional knapsack problem, 429
- fractional matching, 741 pr.
- free tree, 1167, 1169–1171
- frequency count, 802 ex.
- frequency domain, 877
- full binary tree, 1174
 - relation to optimal code, 433
- full node, 502
- full rank, 1220
- full walk of a tree, 1112
- fully parenthesized matrix-chain product, 374
- fully polynomial-time approximation scheme, 1105
 - for subset sum, 1124–1130
- function, 1161–1163
 - Ackermann's, 544
 - α -strongly convex, 1041
 - basis, 841
 - β -smooth, 1041
 - boolean, 1182 ex.
 - convex, 1025–1027, 1194
 - driving, 101
 - final-state, 968
 - hash, *see* hash function
 - iterated, 68, 74 pr.
 - linear, 30, 853
 - objective, 626, 852, 854
 - potential, 456
 - prefix, 975–977
 - probability distribution, 218 ex.
 - quadratic, 31
 - reduction, 1062
 - suffix, 968
 - transition, 967, 973–974
 - watershed, 103
- functional iteration, 68
- fundamental theorem of linear programming, 872
- furthest-in-future, 441
- fusion tree, 226, 478
- fuzzy sorting, 203 pr.

- Gabow's scaling algorithm for single-source shortest paths, 641 pr.
- gadget, 1086, 1097
- GALE-SHAPLEY, 719
- Gale-Shapley algorithm, 718–722
- Galois field of two elements ($GF(2)$), 1224 pr.
- gap character, 961 ex., 975 ex.
- gate, 1065
- Gaussian elimination, 825
- gcd, *see* greatest common divisor
- GCD recursion theorem, 911
- general arithmetic series, 1141
- general number-field sieve, 956
- generating function, 121 pr.
- generation of partitioned sets, 234
- generator
 - of a subgroup, 922
 - of \mathbb{Z}_n^* , 932

- GENERIC-MST, 587
- geometric distribution, 1196–1198
 - and balls and bins, 143–144
 - in finding prime numbers, 943
 - in randomized selection analysis, 232
- geometric series, 1142
- $GF(2)$ (Galois field of two elements), 1224 pr.
- global cut, 701 pr.
- global minimizer, 1022, 1024 fig., 1026 fig.
- global variable, 22
- golden ratio (ϕ), 69, 70 ex.
- gossiping, 475
- gradient descent, 1022–1038
 - constrained, 1032–1034
 - in machine learning, 1035–1037
 - for solving systems of linear equations, 1034–1035
 - stochastic, 1040 pr.
 - unconstrained, 1023–1031
- GRADIENT-DESCENT, 1025
- GRADIENT-DESCENT-CONSTRAINED, 1032
- gradient of a function, 1023
- GRAFT, 542 pr.
- grain size in a parallel algorithm, 783 pr.
- graph, 1164–1169
 - adjacency-list representation of, 550–551
 - adjacency-matrix representation of, 551–552
 - and asymptotic notation, 548
 - attributes of, 548, 552
 - breadth-first search of, 554–563
 - coloring of, 1100 pr.
 - complement of, 1085
 - component, 576
 - constraint, 628–630
 - dense, 549
 - depth-first search of, 563–572
 - dynamic, 523, 817
 - ϵ -dense, 668 pr.
 - hamiltonian, 1056
 - interval, 425 ex.
 - matching in, 693–697, 704–743
 - nonhamiltonian, 1056
 - planar, 584 pr.
 - regular, 716 ex., 740 pr.
 - shortest path in, 558
 - singly connected, 572 ex.
 - sparse, 549
 - static, 522
 - subproblem, 370–371
 - tour of, 1090
 - weighted, 551
 - see also* directed acyclic graph, directed graph, flow network, undirected graph, tree
- GRAPH-ISOMORPHISM, 1060 ex.
- Gray code, 471 pr.
- gray vertex, 554, 564
- greatest common divisor (gcd), 906–907, 910 ex.
 - binary gcd algorithm for, 953 pr.
 - Euclid’s algorithm for, 911–916, 954 pr.
 - with more than two arguments, 916 ex.
 - recursion theorem for, 911
- GREEDY-ACTIVITY-SELECTOR, 424
- GREEDY-BIPARTITE-MATCHING, 726
- greedy-choice property, 427–428
 - of activity selection, 420–421
 - of Huffman codes, 436–437
 - of offline caching, 442–445
- greedy method, 417–447
 - for activity selection, 418–425
 - for coin changing, 446 pr.
 - compared with dynamic programming, 384–385, 393 ex., 421, 426–430
 - Dijkstra’s algorithm, 620–626
 - elements of, 426–431
 - for the fractional knapsack problem, 429
 - greedy-choice property in, 427–428
 - for Huffman code, 431–439
 - Kruskal’s algorithm, 592–594
 - for maximal bipartite matching, 726
 - for minimum spanning tree, 591–599
 - for offline caching, 440–446
 - optimal substructure in, 428
 - Prim’s algorithm, 594–597
 - for set cover, 1115–1119
 - for task-parallel scheduling, 759–761, 769 ex.
 - for task scheduling, 446 pr.
 - for weighted set cover, 1132 pr.
- greedy scheduler, 759
- GREEDY-SET-COVER, 1117
- grid, 697 pr.
- group, 916–923
 - cyclic, 932
 - operator (\oplus), 917

- growth step, 736
- guessing the solution, in the substitution method, 92
- Habanero-Java, 750
- half 3-CNF satisfiability, 1099 ex.
- half-open interval ($[a, b)$ or $(a, b]$), 1157
- Hall's theorem, 715 ex.
- halting, 6
- halting problem, 1042
- halving lemma, 887
- HAM-CYCLE, 1056
- hamiltonian cycle, 1043, 1056
 - NP completeness of, 1085–1090
- hamiltonian graph, 1056
- hamiltonian path, 1060 ex., 1098 ex.
- HAM-PATH, 1060 ex.
- handle, 173
- handshaking lemma, 1168 ex.
- harmonic number, 1142, 1149
- harmonic series, 1142, 1149
- HASH-DELETE, 300 ex.
- hash function, 275, 282–292
 - auxiliary, 295
 - collision-resistant, 941
 - cryptographic, 291
 - division method for, 284, 292 ex.
 - for hierarchical memory, 304–307
 - multiplication method for, 284–286
 - multiply-shift method for, 285–286
 - random, 286–290
 - static, 282, 284–286
 - universal, 286–290
 - wee, 305–307
 - see also* family of hash functions
- hashing, 272–311
 - with chaining, 277–281, 308 pr.
 - double, 295–297, 301 ex.
 - independent uniform, 276
 - with linear probing, 297, 302–304
 - in memoization, 368, 391
 - with open addressing, 293–301, 308 pr.
 - perfect, 310
 - random, 286–290
 - to replace adjacency lists, 553 ex.
 - of static sets, 308 pr.
 - of strings, 290–291, 292 ex.
 - uniform, 278
 - universal, 286–290, 309 pr.
 - of variable-length inputs, 290–291
 - of vectors, 290–291
- HASH-INSERT, 294, 300 ex.
- HASH-SEARCH, 294, 300 ex.
- hash table, 275–282
 - dynamic, 470 ex.
 - used within a priority queue, 174
 - see also* hashing
- hash value, 275
- hat-check problem, 134 ex.
- head
 - in a disk drive, 498
 - of a linked list, 259
 - of a queue, 256
- heap, 161–181
 - analyzed by potential method, 459 ex.
 - building, 167–170, 178 pr.
 - in constructing Huffman codes, 436
 - d -ary, 179 pr., 668 pr.
 - deletion from, 178 ex.
 - in Dijkstra's algorithm, 623
 - extracting the maximum key from, 174
 - Fibonacci, 478
 - height of, 163
 - increasing a key in, 174–175
 - insertion into, 175
 - in Johnson's algorithm, 666
 - max-heap, 162
 - maximum key of, 174
 - mergeable, 268 pr.
 - min-heap, 163
 - in Prim's algorithm, 597
 - as a priority queue, 172–178
- HEAP-DECREASE-KEY, 176 ex.
- HEAP-EXTRACT-MIN, 176 ex.
- HEAP-MINIMUM, 176 ex.
- heap property, 162
 - maintenance of, 164–167
 - vs. binary-search-tree property, 315 ex.
- heapsort, 161–181
 - lower bound for, 207
- HEAPSORT, 170
- HEDGE, 1039 pr.
- height
 - black-, 332
 - of a B-tree, 502–504
 - of a d -ary heap, 179 pr.

- height, *continued*
 - of a decision tree, 207
 - of a heap, 163
 - of a node in a heap, 163, 170 *ex.*
 - of a node in a tree, 1173
 - of a red-black tree, 332
 - of a tree, 1173
- height-balanced tree, 357 *pr.*
- helpful partitioning, 232
- Hermitian matrix, 838 *ex.*
- Hessian matrix, 1035
- heuristic
 - first-fit for bin packing, 1131 *pr.*
 - path compression, 528
 - in the Rabin-Karp algorithm, 965
 - for the set-covering problem, 1116, 1132 *pr.*
 - table doubling, 461
 - for the traveling-salesperson problem, 1115 *ex.*
 - union by rank, 528
 - weighted union, 525
- high endpoint of an interval, 489
- high side of a partition, 183
- HIRE-ASSISTANT, 127
- hiring problem, 126–127, 135–136
 - online, 150–152
 - probabilistic analysis of, 132–133
- hit, 965
- HOARE-PARTITION, 199 *pr.*
- HOPCROFT-KARP, 709
- Hopcroft-Karp algorithm, 709–715
- HORNER, 47 *pr.*
- Horner's rule, 46 *pr.*, 879, 963
- HUFFMAN, 434
- Huffman code, 431–439
- Human Genome Project, 6
- HUNGARIAN, 737
- Hungarian algorithm, 723–739, 740 *pr.*
- hybrid cryptosystem, 941
- hyperedge, 1167
- hypergraph, 1167
- hypotheses, 1003
- ideal parallel computer, 756
- idempotency laws, 1154
- identity, 917
- identity matrix, 1215
- identity permutation, 138 *ex.*
- if**, in pseudocode, 22
- ill-defined recurrence, 77
- image, 1162
- image compression, 412 *pr.*
- incidence, 1164–1165
- incidence matrix
 - and difference constraints, 628
 - of a directed graph, 553 *ex.*
- inclusion and exclusion, 1158 *ex.*
- incomplete step, 759
- INCREASE-KEY, 173
- increasing a key, in a max-heap, 174–175
- INCREMENT, 451
- incremental design method, 34
- incrementing, 21
- in-degree, 1165
- indentation in pseudocode, 21–22
- independence
 - of events, 1188, 1190 *ex.*
 - of random variables, 1192
 - of subproblems in dynamic programming, 386–387
- independent family of hash functions, 288
- independent set, 1099 *pr.*
- independent uniform hash function, 276
- independent uniform hashing, 276, 278
- independent uniform permutation hashing, 295
- indexing into an array, 22–23, 26 *n.*, 252
- index of an element of \mathbb{Z}_n^* , 933
- indicator random variable, 130–133
 - in approximation algorithm for MAX-3-CNF satisfiability, 1120–1121
 - in birthday paradox analysis, 142–143
 - in bounding the right tail of the binomial distribution, 1207–1208
 - in coin flipping analysis, 131–132
 - expected value of, 130
 - in hashing analysis, 279–281
 - in the hat-check problem, 134 *ex.*
 - in hiring-problem analysis, 132–133
 - and linearity of expectation, 131
 - in quicksort analysis, 197–198, 200 *pr.*
 - in randomized caching analysis, 812
 - in randomized-selection analysis, 245 *pr.*
 - in streak analysis, 148–150
- induced subgraph, 1166
- inequality, linear, 853
- infeasible linear program, 854

- infeasible solution, 854
- inference, 1004
- infinite sequence, 1162
- infinite set, 1156
- infinite sum, 1140
- infinity, arithmetic with, 611
- initialization of a loop invariant, 20
- INITIALIZE-SINGLE-SOURCE, 609
- injective function, 1162
- inner product, 1219
- inorder tree walk, 314, 320 *ex.*
- INORDER-TREE-WALK, 314
- in-place permuting, 136
- in-place sorting, 158, 220 *pr.*
- in play, 232
- input
 - to an algorithm, 5
 - to a combinational circuit, 1066
 - distribution of, 128, 134
 - to a logic gate, 1065
 - size of, 28
- input alphabet, 967
- INSERT, 173, 250, 460 *ex.*
- insertion
 - into binary search trees, 321–322
 - into B-trees, 506–511
 - into chained hash tables, 278
 - into d -ary heaps, 179 *pr.*
 - into direct-address tables, 274
 - into dynamic tables, 461–465
 - elementary, 461
 - into heaps, 175
 - into interval trees, 491
 - into linked lists, 260
 - into open-address hash tables, 293–294
 - into order-statistic trees, 484
 - into queues, 256
 - into red-black trees, 338–346
 - into stacks, 254
 - into Young tableaux, 179 *pr.*
- insertion sort, 17–21, 29–31, 51–53, 56–57
 - in bucket sort, 216–218
 - compared with merge sort, 12–13, 15 *ex.*
 - compared with quicksort, 191 *ex.*
 - decision tree for, 206 *fig.*
 - lower bound for, 52–53
 - in merge sort, 45 *pr.*
 - in quicksort, 198 *ex.*
 - using binary search, 45 *ex.*
- INSERTION-SORT, 19, 30, 51
- instance
 - of an abstract problem, 1045, 1049
 - of a problem, 6
- instructions of the RAM model, 26
- integer data type, 26
- integer linear programming, 857, 874 *pr.*, 1098 *ex.*
- integers (\mathbb{Z}), 1153
- integer-valued flow, 695
- integrality theorem, 696
- integral, to approximate summations, 1150
- integration of a series, 1142
- interior-point methods, 857
- intermediate vertex, 655
- internal node, 1172
- internal path length, 1175 *ex.*
- interpolation by a cubic spline, 847 *pr.*
- interpolation by a polynomial, 880, 885 *ex.*
 - at complex roots of unity, 891–892
- intersection
 - of chords, 486 *ex.*
 - of languages, 1052
 - of sets (\cap), 1154
- interval, 489–490, 1157
 - fuzzy sorting of, 203 *pr.*
- INTERVAL-DELETE, 490, 496 *pr.*
- interval graph, 425 *ex.*
- INTERVAL-INSERT, 490, 496 *pr.*
- INTERVAL-SEARCH, 490, 492
- INTERVAL-SEARCH-EXACTLY, 495 *ex.*
- interval tree, 489–495
- interval trichotomy, 490
- intractability, 1042
- invalid shift, 957
- inventory planning, 414 *pr.*
- inverse
 - of a bijective function, 1163
 - in a group, 917
 - of a matrix, 784 *pr.*, 833–837, 1220
 - multiplicative, modulo n , 927
- inversion
 - in an array, 47 *pr.*
 - in linked lists, 798
 - in a sequence, 134 *ex.*, 486 *ex.*

- inversion count, 798
- inverter, 1065
- invertible matrix, 1220
- invocation tree, 756
- isolated vertex, 1165
- isomorphic graphs, 1166
- iterated function, 68, 74 pr.
- iterated logarithm function (\lg^*), 68
- ITERATIVE-TREE-SEARCH, 316
- iter function, 536

- Java Fork-Join Framework, 750
- Jensen's inequality, 1194
- JOHNSON, 666
- Johnson's algorithm, 662–667
- joining
 - of red-black trees, 356 pr.
 - of 2-3-4 trees, 518 pr.
- joint probability density function, 1191
- Josephus permutation, 496 pr.

- Karmarkar's algorithm, 876
- Karp's minimum mean-weight cycle algorithm, 642 pr.
- k -ary tree, 1174
- k -clustering, 1008
- k -CNF, 1043
- k -coloring, 1100 pr., 1176 pr.
- k -combination, 1180
- k -conjunctive normal form, 1043
- Keeling curve, 845 fig.
- key, 17, 157, 173, 249, 283–284
 - in a cryptosystem, 936, 939
 - dummy, 400
 - median, of a B-tree node, 506
- keywords, in pseudocode, 21–22, 64
 - parallel, 750, 752–754, 762–763
- Kleene star (*), 1052
- k -means problem, 1008
- KMP algorithm, 975–985
- KMP-MATCHER, 978
- knapsack problem
 - decision version, 1096
 - fractional, 429
 - 0-1, 428, 430 ex., 1134 pr.
- k -neighbor tree, 358
- knot, of a spline, 847 pr.
- Knuth-Morris-Pratt algorithm, 975–985

- k -permutation, 136, 1180
- Kraft inequality, 1176 ex.
- Kruskal's algorithm, 592–594
 - with integer edge weights, 598 ex.
- k -sorted, 221 pr.
- k -string, 1179
- k -subset, 1156
- k -substring, 1179
- k th power, 910 ex.

- label
 - in machine learning, 1003, 1035
 - of a vertex, 724, 742 pr.
- Lagrange's formula, 881
- Lagrange's theorem, 921
- Lamé's theorem, 913
- language, 1052
 - completeness of, 1072 ex.
 - proving NP-completeness of, 1072–1073
 - verification of, 1058
- lasers, sharks with, 850
- last-in, first-out (LIFO), 254, 803–804
 - implemented with a priority queue, 178 ex.
 - see also* stack
- latency, 499
- LCA, 544 pr.
- lcm (least common multiple), 916 ex.
- LCP, *see* longest common prefix array
- LCS, 393–399
- LCS-LENGTH, 397
- leading submatrix, 839
- leaf, 1172
- least common multiple, 916 ex.
- least frequently used (LFU), 803, 814 ex.
- least recently used (LRU), 445 ex., 803–805
- least-squares approximation, 841–845, 1035–1037
- leaving a vertex, 1164
- LEFT, 162
- left child, 1173
- left-child, right-sibling representation, 265, 268 ex.
- left-leaning red-black binary search tree, 358
- LEFT-ROTATE, 336, 495 ex.
- left rotation, 335
- left shift (\ll), 305
- left subtree, 1173
- Legendre symbol $\left(\frac{a}{p}\right)$, 954 pr.

- length
 - of a cycle, 1165
 - of a path, 1165
 - of a sequence, 1162
 - of a string, 959, 1179
- level
 - of a function, 532
 - of a node in a disjoint-set forest, 535
 - of a tree, 1173
- lexicographically less than, 327 pr.
- lexicographic sorting, 327 pr., 986 n.
- LFU (least frequently used), 803, 814 ex.
- lg (binary logarithm), 66
- lg* (iterated logarithm function), 68
- lg^k (exponentiation of logarithms), 66
- lg lg (composition of logarithms), 66
- LIFO, *see* last-in, first-out; stack
- light edge, 587
- linear constraint, 853–854
- linear dependence, 1220
- linear equality, 853
- linear equations
 - solving modular, 924–928
 - solving systems of, 819–833, 1034–1035
 - solving tridiagonal systems of, 847 pr.
- linear function, 30, 853
- linear independence, 1220
- linear inequality, 853
- linear-inequality feasibility problem, 873 pr.
- linearity of expectation, 1192–1193
 - and indicator random variables, 131
- linearity of summations, 1141
- linear order, 1160
- linear permutation, 1224 pr.
- linear probing, 297, 302–304
- LINEAR-PROBING-HASH-DELETE, 303
- linear programming, 850–876, 1121–1124
 - applications of, 860–866
 - duality in, 866–873
 - ellipsoid algorithm for, 857
 - fundamental theorem of, 872
 - integer, 857, 874 pr.
 - interior-point methods for, 857
 - Karmarkar’s algorithm for, 876
 - and maximum flow, 862
 - and minimum-cost circulation, 875 pr.
 - and minimum-cost flow, 862–864
 - and multicommodity flow, 864–865
 - simplex algorithm for, 857
 - and single-pair shortest path, 861
 - and single-source shortest paths, 626–632
 - standard form for, 854
 - see also* integer linear programming, 0-1 integer programming
- linear-programming relaxation, 1122
- linear regression, 1036
- linear search, 25 ex.
- linear speedup, 758
- line search, 1031
- LINK, 530
- linked list, 258–264
 - compact, 269 pr.
 - deletion from, 261
 - to implement disjoint sets, 523–527
 - insertion into, 260
 - maintained by an online algorithm, 795–802
 - searching, 260, 292 ex.
- linking of trees in a disjoint-set forest, 529
- list, *see* linked list
- LIST-DELETE, 261
- LIST-DELETE’, 262
- LIST-INSERT, 261
- LIST-INSERT’, 263
- LIST-PREPEND, 260
- LIST-SEARCH, 260
- LIST-SEARCH’, 263
- literal, 1076
- little-oh notation (*o*), 60
- little-omega notation (*ω*), 61
- Lloyd’s procedure, 1011–1013, 1039 pr.
- ln (natural logarithm), 66
- load factor
 - of a dynamic table, 461
 - of a hash table, 278
- load instruction, 26, 756
- local minimizer, 1026 fig.
- local variable, 22
- logarithm function (log), 66–67
 - discrete, 933
 - iterated (lg*), 68
- logical parallelism, 753
- logical right shift (*≫*), 285
- logic gate, 1065
- longest common prefix (LCP) array, 986, 992–994
- longest common subsequence, 393–399

- longest common substring, 995 ex.
- longest monotonically increasing subsequence, 399 ex.
- longest palindrome subsequence, 407 pr.
- LONGEST-PATH, 1055 ex.
- LONGEST-PATH-LENGTH, 1055 ex.
- longest repeated substring, 987
- longest simple cycle, 1098 ex.
- longest simple path, 1042
 - in an unweighted graph, 385
 - in a weighted directed acyclic graph, 407 pr.
- lookahead algorithm, 815 ex.
- LOOKUP-CHAIN, 391
- loop, in pseudocode, 22
 - parallel, 762–765
- loop invariant, 19–20
 - for breadth-first search, 555
 - for building a heap, 167
 - for counting sort, 211 ex.
 - for determining the rank of an element in an order-statistic tree, 483
 - and **for** loops, 21 n.
 - for the generic minimum-spanning-tree method, 586
 - for heapsort, 172 ex.
 - for Horner’s rule, 46 pr.
 - for increasing a key in a heap, 177 ex.
 - for insertion sort, 19–20
 - for partitioning, 184
 - for Prim’s algorithm, 597
 - for the Rabin-Karp algorithm, 965
 - for randomly permuting an array, 137
 - for red-black tree insertion, 340
 - for string-matching automata, 970, 973
- loss function, 1036
- low endpoint of an interval, 489
- lower bounds
 - asymptotic, 55
 - on binomial coefficients, 1181, 1184 ex.
 - for comparing water jugs, 220 pr.
 - for competitive ratios for online caching, 804–806
 - for constructing binary search trees, 315 ex.
 - for disjoint-set data structures, 545
 - for finding the minimum, 228
 - for insertion sort, 52–53
 - for k -sorting, 221 pr.
 - for median finding, 247
 - for merging, 222 pr.
 - and potential functions, 475
 - for simultaneous minimum and maximum, 229 ex.
 - for sorting, 205–208, 219 pr., 225
 - for streaks, 147–148, 153 ex.
 - on summations, 1148, 1150
 - for task-parallel computations, 757
 - for traveling-salesperson tour, 1110–1113
 - for vertex cover, 1108, 1121–1123, 1132 pr.
- lower median, 227
- lower-triangular matrix, 1216
- lowest common ancestor, 543 pr.
- low side of a partition, 183
- LRU (least recently used), 445 ex., 803–805
- LU decomposition, 824–827
 - parallel algorithm for, 784 pr.
- LU-DECOMPOSITION, 827
- LUP decomposition, 821
 - computation of, 828–832
 - in matrix inversion, 833–834
 - and matrix multiplication, 838 ex.
 - parallel algorithm for, 784 pr.
 - use of, 821–824
- LUP-DECOMPOSITION, 830
- LUP-SOLVE, 824
- machine learning, 14, 1003–1041
- main memory, 498
- maintenance of a loop invariant, 20
- MAKE-RANKS, 988
- MAKE-SET, 521
 - disjoint-set-forest implementation of, 530
 - linked-list implementation of, 523
- makespan, 1133 pr.
- MAKE-TREE, 542 pr.
- Manhattan distance, 244 pr.
- MARKING, 807, 815 ex.
- Markov’s inequality, 1196 ex.
- master method for solving a recurrence, 101–107
- master recurrence, 101
- master theorem, 102
 - continuous, 112
 - proof of, 107–115
- matched vertex, 693, 705

- matching, 704–743
 - bipartite, 693–697, 704–743
 - fractional, 741 pr.
 - by Hopcroft-Karp algorithm, 709–715
 - maximal, 705, 1108, 1132 pr.
 - maximum, 704–716, 1132 pr.
 - and maximum flow, 693–697
 - perfect, 715 ex., 740 pr.
 - stable, 716
 - of strings, 957–1002
 - unstable, 717
- matrix, 1214–1226
 - addition of, 1217
 - adjacency, 551–552
 - conjugate transpose of, 838 ex.
 - dense, 81
 - determinant of, 1221
 - diagonal, 1215
 - Hermitian, 838 ex.
 - Hessian, 1035
 - identity, 1215
 - incidence, 553 ex.
 - inverse of, 784 pr., 833–837, 1220
 - lower-triangular, 1216
 - minor of, 1221
 - multiplication of, *see* matrix multiplication
 - negative of, 1217
 - permutation, 1217
 - positive-definite, 1222
 - positive-semidefinite, 1222
 - predecessor, 647, 655 ex., 657–659, 661 ex.
 - product of, with a vector, 762–765, 767, 1218
 - pseudoinverse of, 843
 - representation of, 253–254
 - scalar multiple of, 1217
 - sparse, 81
 - subtraction of, 1218
 - symmetric, 1217
 - symmetric positive-definite, 838–841
 - transpose of, 1214
 - tridiagonal, 1216
 - unit lower-triangular, 1216
 - unit upper-triangular, 1216
 - upper-triangular, 1216
 - Vandermonde, 881, 1223 pr.
- matrix-chain multiplication, 373–382
- MATRIX-CHAIN-MULTIPLY, 381 ex.
- MATRIX-CHAIN-ORDER, 378
- matrix multiplication, 80–90, 1218
 - for all-pairs shortest paths, 648–655, 668–669
 - divide-and-conquer method for, 81–90, 770–775, 783 pr.
 - and LUP decomposition, 838 ex.
 - and matrix inversion, 834–837
 - Pan’s method for, 89 ex.
 - parallel algorithm for, 770–775, 783 pr.
 - Strassen’s algorithm for, 85–90, 124–125, 773–774
 - and transitive closure, 838 ex.
- MATRIX-MULTIPLY, 81
- MATRIX-MULTIPLY-RECURSIVE, 83
- matrix-vector multiplication, 762–765, 767, 1218
- MAX-CNF satisfiability, 1124 ex.
- MAX-CUT problem, 1124 ex.
- MAX-FLOW-BY-SCALING, 700 pr.
- max-flow min-cut theorem, 684
- max-heap, 162
 - building, 167–170
 - d -ary, 179 pr.
 - deletion from, 178 ex.
 - extracting the maximum key from, 174
 - in heapsort, 170–172
 - increasing a key in, 174–175
 - insertion into, 175
 - maximum key of, 174
 - as a max-priority queue, 172–178
 - mergeable, 268 n.
- MAX-HEAP-DECREASE-KEY, 176 ex.
- MAX-HEAP-DELETE, 178 ex.
- MAX-HEAP-EXTRACT-MAX, 175
- MAX-HEAPIFY, 165
- MAX-HEAP-INCREASE-KEY, 176
- MAX-HEAP-INSERT, 176
 - building a heap with, 178 pr.
- MAX-HEAP-MAXIMUM, 175
- max-heap property, 162
 - maintenance of, 164–167
- maximal element, 1160
- maximal matching, 705, 1108, 1132 pr.
 - greedy method for, 726
- maximization linear program, 853

- maximum, 227
 - in binary search trees, 317–318
 - of a binomial distribution, 1202 *ex.*
 - finding, 228–229
 - in heaps, 174
 - in red-black trees, 334
- MAXIMUM, 173–174, 250
- maximum bipartite matching, 693–697, 705–716
- maximum flow, 670–703
 - Edmonds-Karp algorithm for, 689–691
 - Ford-Fulkerson method for, 676–693
 - as a linear program, 862
 - and maximum bipartite matching, 693–697
 - push-relabel algorithms for, 702
 - scaling algorithm for, 699 *pr.*
 - updating, 699 *pr.*
- maximum matching, 693, 704, 1132 *pr.*
see also maximum bipartite matching
- maximum spanning tree, 1134 *pr.*
- max-priority queue, 173
- MAX-3-CNF satisfiability, 1120–1121
- MAYBE-MST-A, 602 *pr.*
- MAYBE-MST-B, 602 *pr.*
- MAYBE-MST-C, 602 *pr.*
- mean
 - of a cluster, 1009
 - see also* expected value
- mean weight of a cycle, 642 *pr.*
- median, 227–247
 - weighted, 244 *pr.*
- median key, of a B-tree node, 506
- median-of-3 method, 203 *pr.*
- member of a set (\in), 1153
- memoization, 368, 390–392
- MEMOIZED-CUT-ROD, 369
- MEMOIZED-CUT-ROD-AUX, 369
- MEMOIZED-MATRIX-CHAIN, 391
- memory hierarchy, 27, 301
 - hash functions for, 304–307
- MERGE, 36
- mergeable heap, 268 *pr.*
- MERGE-LISTS, 1125
- merge sort, 34–44, 57
 - compared with insertion sort, 12–13, 15 *ex.*
 - lower bound for, 207
 - parallel algorithm for, 775–782
 - use of insertion sort in, 45 *pr.*
- MERGE-SORT, 39
- merging
 - of k sorted lists, 178 *ex.*
 - lower bounds for, 222 *pr.*
 - parallel algorithm for, 776–780
 - of two sorted subarrays, 35–38
- MILLER-RABIN, 946
- Miller-Rabin primality test, 945–953
- MIN-GAP, 495 *ex.*
- min-heap, 163
 - analyzed by potential method, 459 *ex.*
 - building, 167–170
 - in constructing Huffman codes, 436
 - d -ary, 668 *pr.*
 - in Dijkstra's algorithm, 623
 - in Johnson's algorithm, 666
 - mergeable, 268 *n.*
 - as a min-priority queue, 176 *ex.*
 - in Prim's algorithm, 597
- MIN-HEAPIFY, 166 *ex.*
- MIN-HEAP-INSERT, 176 *ex.*
- min-heap property, 163
 - maintenance of, 166 *ex.*
 - vs. binary-search-tree property, 315 *ex.*
- minimization linear program, 853
- minimizer of a function, 1022, 1024 *fig.*, 1026 *fig.*
- minimum, 227
 - in binary search trees, 317–318
 - finding, 228–229
 - offline, 541 *pr.*
 - in red-black trees, 334
- MINIMUM, 173, 228, 250
- minimum-cost circulation, 875 *pr.*
- minimum-cost flow, 862–864
- minimum-cost multicommodity flow, 866 *ex.*
- minimum-cost spanning tree, *see* minimum spanning tree
- minimum cut, 682
 - global, 701 *pr.*
- minimum degree, of a B-tree, 502
- minimum mean-weight cycle, 642 *pr.*
- minimum path cover, 698 *pr.*
- minimum spanning tree, 585–603
 - in approximation algorithm for traveling-salesperson problem, 1110
 - Borůvka's algorithm for, 603
 - on dynamic graphs, 599 *ex.*

- minimum spanning tree, *continued*
 - generic method for, 586–591
 - Kruskal’s algorithm for, 592–594
 - Prim’s algorithm for, 594–597
 - second-best, 599 *pr.*
- minimum-weight spanning tree, *see* minimum spanning tree
- minimum-weight vertex cover, 1121–1124
- minor of a matrix, 1221
- min-priority queue, 173
 - in constructing Huffman codes, 434
 - in Dijkstra’s algorithm, 623–624
 - in Prim’s algorithm, 596–597
- missing child, 1173
- mod, 64, 905
- modeling, 851
- modifying operation, 250
- modular arithmetic, 64, 901 *pr.*, 916–923
- modular equivalence ($\equiv \pmod{n}$), 64, 905
- modular exponentiation, 934–935
- MODULAR-EXPONENTIATION, 935
- modular linear equations, 924–928
- MODULAR-LINEAR-EQUATION-SOLVER, 926
- modulo, 64, 905
- Monge array, 123 *pr.*
- monotone sequence, 181
- monotonically decreasing, 63
- monotonically increasing, 63
- Monty Hall problem, 1210 *pr.*
- MOVE-TO-FRONT, 796–797
- MST-KRUSKAL, 594
- MST-PRIM, 596
- MST-REDUCE, 601 *pr.*
- much-greater-than (\gg), 533
- much-less-than (\ll), 761
- multicommodity flow, 864–865
- multicore computer, 748
- multidimensional fast Fourier transform, 899 *pr.*
- multigraph, 1167
- multiple, 904
 - of an element modulo n , 924–928
 - least common, 916 *ex.*
 - scalar, 1217
- multiple sources and sinks, 674
- multiplication
 - of complex numbers, 90 *ex.*
 - divide-and-conquer method for, 899 *pr.*
 - of matrices, *see* matrix multiplication
 - of a matrix chain, 373–382
 - matrix-vector, 762–765, 767, 1218
 - modulo n (\cdot_n), 917
 - of polynomials, 878
- multiplication method, 284–286
- multiplicative group modulo n , 919
- multiplicative inverse, modulo n , 927
- multiplicative weights, 1015–1022
- multiply instruction, 26
- multiply-shift method, 285–286
- MULTIPOP, 450
- multiset, 1153 *n.*
 - dynamic, 460 *ex.*
- mutually exclusive events, 1185
- mutually independent events, 1188
- mutually noninterfering strands, 767
- \mathbb{N} (set of natural numbers), 1153
- naive algorithm for string matching, 960–962
- NAIVE-STRING-MATCHER, 960
- National Resident Matching Program, 704, 722 *ex.*
- natural cubic spline, 847 *pr.*
- natural logarithm (\ln), 66
- natural numbers (\mathbb{N}), 1153
 - keys interpreted as, 283–284
- nearest-center rule, 1008
- negative of a matrix, 1217
- negative-weight cycle
 - and difference constraints, 629
 - and relaxation, 639 *ex.*
 - and shortest paths, 606–607, 614–615, 655 *ex.*, 662 *ex.*
- negative-weight edges, 606–607
- neighbor, 1167
- neighborhood, 715 *ex.*
- nesting boxes, 640 *pr.*
- net flow across a cut, 682
- network
 - flow, *see* flow network
 - residual, 677–681
 - for sorting, 789

- new request, 810
- Newton's method, 1038 pr.
- NIL, 23
- node, 1172
 - see also* vertex
- nondeterministic algorithm, 765
- nondeterministic polynomial time, 1058 n.
 - see also* NP
- nonempty suffix, 997 pr.
- nonhamiltonian graph, 1056
- noninstance, 1051 n.
- noninvertible matrix, 1220
- nonnegativity constraint, 854
- nonoblivious adversary, 807
- nonoverlappable string pattern, 974 ex.
- nonsample position, 997 pr.
- nonsample suffix, 997 pr.
- nonsingular matrix, 1220
- nontrivial power, 910 ex.
- nontrivial square root of 1, modulo n , 934
- no-path property, 611, 634
- normal equation, 843
- norm of a vector ($\| \ \|$), 1219
- NOT function (\neg), 1065
- not a set member (\notin), 1153
- not equivalent ($\neq \pmod{n}$), 64
- NOT gate, 1065
- NP (complexity class), 1043, 1058, 1060 ex.
- NPC (complexity class), 1044, 1063
- NP-complete, 1044, 1063
- NP-completeness, 9–10, 1042–1103
 - of the circuit-satisfiability problem, 1064–1071
 - of the clique problem, 1081–1084
 - of the formula-satisfiability problem, 1073–1076
 - of the graph-coloring problem, 1100 pr.
 - of the half 3-CNF satisfiability problem, 1099 ex.
 - of the hamiltonian-cycle problem, 1085–1090
 - of the hamiltonian-path problem, 1098 ex.
 - of the independent-set problem, 1099 pr.
 - of integer linear programming, 1098 ex.
 - of the longest-simple-cycle problem, 1098 ex.
 - proving, of a language, 1072–1073
 - reduction strategies for, 1095–1098
 - of scheduling with profits and deadlines, 1102 pr.
 - of the set-covering problem, 1119 ex.
 - of the set-partition problem, 1098 ex.
 - of the subgraph-isomorphism problem, 1098 ex.
 - of the subset-sum problem, 1092–1095
 - of the 3-CNF-satisfiability problem, 1076–1079
 - of the traveling-salesperson problem, 1090–1092
 - of the vertex-cover problem, 1084–1085
 - of 0-1 integer programming, 1098 ex.
- NP-hard, 1063
- n -set, 1156
- n -tuple, 1157
- null event, 1185
- null tree, 1173
- null vector, 1221
- number-field sieve, 956
- numerical stability, 819, 821
- n -vector, 1215
- o -notation, 60
- O -notation, 50, 54–55
- \widetilde{O} -notation, 73 pr.
- \widetilde{O} -notation, 73 pr.
- object, 23
- objective function, 626, 852, 854
- objective value, 854
- oblivious adversary, 807
- oblivious compare-exchange algorithm, 222 pr.
- occurrence of a pattern, 957
- offline algorithm, 791
- OFFLINE-MINIMUM, 542 pr.
- offline problem
 - caching, 440–446
 - lowest common ancestors, 543 pr.
 - minimum, 541 pr.
- old request, 810
- Omega-notation, 51, 54 fig., 55–56
- 1-approximation algorithm, 1105
- one-pass method, 544
- one-to-one correspondence, 1163
- one-to-one function, 1162
- online algorithm, 791–818
 - for caching, 802–815
 - for the cow-path problem, 815 pr.

- online algorithm, *continued*
 - for hiring, 150–152
 - for maintaining a linked list, 795–802
 - for task scheduling, 816 pr.
 - for waiting for an elevator, 792–794
- online learning, 1003
- ONLINE-MAXIMUM, 150
- online task-parallel scheduler, 759
- onto function, 1162
- open-address hash table, 293–301, 308 pr.
 - with double hashing, 295–297, 301 ex.
 - with linear probing, 297, 302–304
- open interval $((a, b))$, 1157
- OpenMP, 750
- optimal assignment, 723–739
- optimal binary search tree, 400–407
- OPTIMAL-BST, 405
- optimal objective value, 854
- optimal solution, 854
- optimal substructure, 382–387
 - of activity selection, 419
 - of binary search trees, 402–403
 - of the fractional knapsack problem, 429
 - in greedy method, 428
 - of Huffman codes, 438
 - of longest common subsequences, 394–395
 - of matrix-chain multiplication, 376
 - of offline caching, 441–442
 - of rod cutting, 365
 - of shortest paths, 605–606, 649, 655–656
 - of unweighted shortest paths, 385
 - of the 0-1 knapsack problem, 429
- optimal vertex cover, 1106
- optimization problem, 362, 1045, 1049
 - approximation algorithms for, 1104–1136
 - and decision problems, 1045
- OR function (\vee), 659, 1065
- order
 - of a group, 922
 - of growth, 32
 - linear, 1160
 - partial, 1160
 - total, 1160
- ordered pair, 1156
- ordered tree, 1173
- order statistics, 160, 227–247
 - dynamic, 480–486
- order-statistic tree, 480–486
- ord function, 987
- OR gate, 1065
- or, in pseudocode, 24
- orthonormal, 849
- OS-KEY-RANK, 485 ex.
- OS-RANK, 483
- OS-SELECT, 482
- outcome, 1185
- out-degree, 1165
- outer product, 1219
- output
 - of an algorithm, 5
 - of a combinational circuit, 1066
 - of a logic gate, 1065
- overdetermined system of linear equations, 821
- overflow
 - of a queue, 257
 - of a stack, 255
- overflowing vertex, 703
- overlapping intervals, 489
 - finding all, 495 ex.
 - point of maximum overlap, 496 pr.
- overlapping rectangles, 495 ex.
- overlapping subproblems, 387–390
- overlapping-suffix lemma, 959
- P (complexity class), 1043, 1050, 1054, 1055 ex.
- page, in virtual memory, 440
- pair
 - blocking, 716
 - ordered, 1156
- pairwise disjoint sets, 1156
- pairwise independence, 1188
- pairwise relatively prime, 908
- palindrome, 407 pr., 995 ex.
- Pan's method for matrix multiplication, 89 ex.
- parallel algorithm, 748–790
 - for computing Fibonacci numbers, 750–753
 - grain size in, 783 pr.
 - for LU decomposition, 784 pr.
 - for LUP decomposition, 784 pr.
 - for matrix inversion, 784 pr.
 - for matrix multiplication, 770–775, 783 pr.
 - for matrix-vector product, 762–765, 767
 - for merge sort, 775–782
 - for merging, 776–780
 - for prefix computation, 784 pr.

- parallel algorithm, *continued*
 - for quicksort, 789 pr.
 - randomized, 789 pr.
 - for reduction, 784 pr.
 - for a simple stencil calculation, 787 pr.
 - for solving systems of linear equations, 784 pr.
 - Strassen's algorithm, 773–774
 - for well-formed parentheses, 786 pr.
- parallel computer, 10, 748, 756
- parallel for**, in pseudocode, 762–763
- parallelism
 - logical, 753
 - of a randomized parallel algorithm, 789 pr.
 - spawning, 753
 - syncing, 754
 - of a task-parallel computation, 758
- parallel keywords, 750, 752, 762
- parallel loop, 762–765, 783 pr.
- parallel-machine-scheduling problem, 1133 pr.
- parallel prefix, 784 pr.
- parallel random-access machine, 789
- parallel slackness, 758
 - rule of thumb, 761
- parallel, strands logically in, 756
- parallel trace, 754–756
 - series-parallel composition of, 762 fig.
- parameter, 23
 - costs of passing, 120 pr.
- parent
 - in a breadth-first tree (π), 555
 - in a parallel computation, 753
 - in a rooted tree, 1172
- PARENT, 162
- parenthesis theorem, 567
- parenthesization of a matrix-chain product, 374
- Pareto optimality, 722 ex.
- parse tree, 1077
- partial derivative (∂), 1023
- partial order, 1160
- PARTITION, 184
- PARTITION', 200 pr.
- PARTITION-AROUND, 237
- partition function, 363 n.
- partitioning, 183–186
 - around median of 3 elements, 198 ex.
 - helpful, 232
 - Hoare's method for, 199 pr.
 - randomized, 192, 198 ex., 200 pr., 203 pr.
- partition of a set, 1156, 1159
- Pascal's triangle, 1183 ex.
- path, 1165
 - alternating, 705
 - augmenting, 681–682, 705
 - critical, 619
 - find, 528
 - hamiltonian, 1060 ex., 1098 ex.
 - longest, 385, 1042
 - shortest, *see* shortest paths
 - simple, 1165
 - weight of, 407 pr., 604
- PATH, 1045, 1053
- path compression, 528
- path cover, 698 pr.
- path length, of a tree, 328 pr., 1175 ex.
- path-relaxation property, 611, 635
- pattern, 957
 - nonoverlappable, 974 ex.
- pattern matching, *see* string matching
- perfect hashing, 310
- perfect linear speedup, 758
- perfect matching, 715 ex., 740 pr.
- permutation, 1163, 1179–1180
 - bit-reversal, 897
 - identity, 138 ex.
 - in place, 136
 - Josephus, 496 pr.
 - k -permutation, 136, 1180
 - linear, 1224 pr.
 - random, 136–138
 - uniform random, 128, 136
- permutation matrix, 1217
- PERMUTE-BY-CYCLE, 139 ex.
- PERMUTE-WITH-ALL, 139 ex.
- PERMUTE-WITHOUT-IDENTITY, 138 ex.
- persistent data structure, 355 pr., 478
- PERSISTENT-TREE-INSERT, 355 pr.
- PERT chart, 617, 619 ex.
- P-FIB, 753
- phi function ($\phi(n)$), 920
- pivot
 - in LU decomposition, 826
 - in quicksort, 183
 - in selection, 230

- $P[:k]$ (prefix of a pattern), 959
- planar graph, 584 pr.
- platter in a disk drive, 498
- P-MATRIX-MULTIPLY, 771
- P-MATRIX-MULTIPLY-RECURSIVE, 772
- P-MAT-VEC, 763
- P-MAT-VEC-RECURSIVE, 763
- P-MAT-VEC-WRONG, 768
- P-MERGE, 779
- P-MERGE-AUX, 779
- P-MERGE-SORT, 775
- P-NAIVE-MERGE-SORT, 775
- pointer, 23
 - trailing, 321
- point, in clustering, 1006
- point-value representation, 880
- polylogarithmically bounded, 67
- polynomial, 65, 877–885
 - addition of, 877
 - asymptotic behavior of, 71 pr.
 - coefficient representation of, 879
 - derivatives of, 900 pr.
 - evaluation of, 46 pr., 879, 884 ex., 900 pr.
 - interpolation by, 880, 885 ex.
 - multiplication of, 878, 882–884, 899 pr.
 - point-value representation of, 880
- polynomial-growth condition, 116–117
- polynomially bounded, 65
- polynomially related, 1051
- polynomial-time acceptance, 1053
- polynomial-time algorithm, 904, 1042
- polynomial-time approximation scheme, 1105
 - for maximum clique, 1131 pr.
 - for subset sum, 1124–1130
- polynomial-time computability, 1051
- polynomial-time decision, 1053
- polynomial-time reducibility (\leq_p), 1062, 1071 ex.
- polynomial-time solvability, 1050
- polynomial-time verification, 1056–1061
- POP, 255, 449
- pop from a runtime stack, 202 pr.
- positional tree, 1174
- positive-definite matrix, 1222
- positive-semidefinite matrix, 1222
- post-office location problem, 244 pr.
- postorder tree walk, 314
- potential function, 456
 - for lower bounds, 475
- potential method, 456–460
 - for binary counters, 458–459
 - for disjoint-set data structures, 534–540, 541 ex.
 - for dynamic tables, 463–470
 - for maintaining a linked list, 799–801
 - for min-heaps, 459 ex.
 - for restructuring red-black trees, 473 pr.
 - for stack operations, 457–458
- potential of a data structure, 456
- power
 - of an element, modulo n , 932–936
 - k th, 910 ex.
 - nontrivial, 910 ex.
- power series, 121 pr.
- power set, 1156
- $\Pr\{\}$ (probability distribution), 1185
- PRAM, 789
- predecessor
 - in binary search trees, 318–319
 - in breadth-first trees (π), 555
 - in linked lists, 259
 - in red-black trees, 334
 - in shortest-paths trees (π), 608
- PREDECESSOR, 250
- predecessor matrix, 647, 655 ex., 657–659, 661 ex.
- predecessor subgraph
 - in all-pairs shortest paths, 647
 - in breadth-first search, 561
 - in depth-first search, 564
 - in single-source shortest paths, 608
- predecessor-subgraph property, 611, 637–638
- prediction, 1004
- prediction phase, 1003
- preemption, 446 pr., 816 pr.
- prefix
 - of a sequence, 395
 - of a string (\sqsubset), 959
- prefix computation, 784 pr.
- prefix-free code, 432
- prefix function, 975–977
- prefix-function iteration lemma, 980
- preflow, 703
- preimage of a matrix, 1224 pr.

- preorder, total, 1160
- preorder tree walk, 314
- Prim's algorithm, 594–597
 - with an adjacency matrix, 598 ex.
 - in approximation algorithm for traveling-salesperson problem, 1110
 - with integer edge weights, 598 ex.
 - similarity to Dijkstra's algorithm, 624
 - for sparse graphs, 599 pr.
- primality testing, 942–953, 956
 - Miller-Rabin test, 945–953
 - pseudoprimality testing, 944–945
- primal linear program, 866
 - augmented, 870
- primary clustering, 303
- prime distribution function, 943
- prime factorization of integers, 909
- prime number, 905
 - density of, 943
- prime number theorem, 943
- primitive root of \mathbb{Z}_n^* , 932
- principal root of unity, 886
- principle of inclusion and exclusion, 1158 ex.
- PRINT-ALL-PAIRS-SHORTEST-PATH, 648
- PRINT-CUT-ROD-SOLUTION, 372
- PRINT-LCS, 397
- PRINT-OPTIMAL-PARENS, 381
- PRINT-PATH, 562
- PRINT-SET, 531 ex.
- priority queue, 172–178
 - in constructing Huffman codes, 434
 - in Dijkstra's algorithm, 623–624
 - heap implementation of, 172–178
 - max-priority queue, 173
 - min-priority queue, 173, 176 ex.
 - with monotone extractions, 181
 - in Prim's algorithm, 596–597
 - see also* Fibonacci heap
- probabilistically checkable proof, 1103, 1136
- probabilistic analysis, 127–128, 140–153
 - of approximation algorithm for MAX-3-CNF satisfiability, 1120–1121
 - and average inputs, 32
 - of average node depth in a randomly built binary search tree, 328 pr.
 - of balls and bins, 143–144
 - of birthday paradox, 140–143
 - of bucket sort, 216–218, 218 ex.
 - of collisions, 281 ex.
 - of file comparison, 967 ex.
 - of fuzzy sorting of intervals, 203 pr.
 - of hashing with chaining, 278–281
 - of hiring problem, 132–133, 150–152
 - of insertion into a binary search tree with equal keys, 327 pr.
 - of longest probe bound for hashing, 308 pr.
 - of lower bound for sorting, 219 pr.
 - of Miller-Rabin primality test, 948–953
 - of online hiring problem, 150–152
 - of open-address hashing, 297–300
 - and parallel algorithms, 789 pr.
 - of partitioning, 191 ex., 198 ex., 200 pr., 203 pr.
 - of probabilistic counting, 153 pr.
 - of quicksort, 194–198, 200 pr., 203 pr.
 - of Rabin-Karp algorithm, 965–966
 - and randomized algorithms, 134–136
 - of randomized online caching, 809–814
 - of randomized selection, 232–236, 245 pr.
 - of randomized weighted majority, 1022 ex.
 - of searching a sorted compact list, 269 pr.
 - of slot-size bound for chaining, 308 pr.
 - of sorting points by distance from origin, 218 ex.
 - of streaks, 144–150
 - of universal hashing, 286–290
- probabilistic counting, 153 pr.
- probability, 1184–1191
- probability axioms, 1185
- probability density function, 1191
- probability distribution, 1185
- probability distribution function, 218 ex.
- probe sequence, 293
- probing, 293
 - see also* linear probing, double hashing
- problem
 - abstract, 1048
 - computational, 5–6
 - concrete, 1049
 - decision, 1045, 1049
 - intractable, 1042
 - optimization, 362, 1045, 1049
 - solution to, 6, 1049
 - tractable, 1042
- procedure, 18
 - calling, 23, 26, 29 n.

- product (\prod), 1144
 - Cartesian (\times), 1157
 - inner, 1219
 - of matrices, *see* matrix multiplication
 - outer, 1219
 - of polynomials, 878
 - rule of, 1179
 - scalar flow, 675 *ex.*
- professional wrestler, 563 *ex.*
- program counter, 1068
- programming, *see* dynamic programming,
 - linear programming
- projection, 1032
- proper ancestor, 1172
- proper descendant, 1172
- proper prefix, 959
- proper subgroup, 921
- proper subset (\subset), 1154
- proper suffix, 959
- P-SCAN-1, 785 *pr.*
- P-SCAN-1-AUX, 785 *pr.*
- P-SCAN-2, 786 *pr.*
- P-SCAN-2-AUX, 786 *pr.*
- P-SCAN-3, 787 *pr.*
- P-SCAN-DOWN, 787 *pr.*
- P-SCAN-UP, 787 *pr.*
- pseudocode, 18, 21–24
- pseudoinverse, 843
- pseudoprime, 944–945
- PSEUDOPRIME, 945
- pseudorandom-number generator, 129
- P-TRANPOSE, 770 *ex.*
- public key, 936, 939
- public-key cryptosystem, 936–942
- PUSH, 255, 449
- push onto a runtime stack, 202 *pr.*
- push-relabel algorithms, 702

- quadratic convergence, 1039 *pr.*
- quadratic function, 31
- quadratic residue, 954 *pr.*
- quantile, 242 *ex.*
- query, 250
- queue, 254, 256–257
 - in breadth-first search, 554
 - implemented by stacks, 258 *ex.*, 460 *ex.*
 - linked-list implementation of, 264 *ex.*
 - priority, *see* priority queue
- quicksort, 182–204
 - analysis of, 187–191, 193–198
 - average-case analysis of, 194–198
 - compared with insertion sort, 191 *ex.*
 - compared with radix sort, 214
 - with equal element values, 200 *pr.*
 - good worst-case implementation of, 241 *ex.*
 - with median-of-3 method, 203 *pr.*
 - parallel algorithm for, 789 *pr.*
 - randomized version of, 191–193, 200 *pr.*, 203 *pr.*
 - stack depth of, 202 *pr.*
 - and tail recursion, 202 *pr.*
 - use of insertion sort in, 198 *ex.*
 - worst-case analysis of, 193–194
- QUICKSORT, 183
- QUICKSORT', 200 *pr.*
- quotient, 905

- \mathbb{R} (set of real numbers), 1153
- Rabin-Karp algorithm, 962–967
- RABIN-KARP-MATCHER, 966
- race condition, 765–768
- RACE-EXAMPLE, 766
- radix sort, 211–215
 - compared with quicksort, 214
 - in computing suffix arrays, 992
- RADIX-SORT, 213
- radix tree, 327 *pr.*
- RAM, 26–27
- RANDOM, 129
- random-access machine, 26–27
 - parallel, 789
- random hashing, 286–290
- randomized algorithm, 128–129, 134–140
 - and average inputs, 32
 - comparison sort, 219 *pr.*
 - for fuzzy sorting of intervals, 203 *pr.*
 - for hiring problem, 135–136
 - for insertion into a binary search tree with equal keys, 327 *pr.*
 - for MAX-3-CNF satisfiability, 1120–1121
 - Miller-Rabin primality test, 945–953
 - for online caching, 807–814
 - parallel, 789 *pr.*
 - for partitioning, 192, 198 *ex.*, 200 *pr.*, 203 *pr.*
 - for permuting an array, 136–138
 - and probabilistic analysis, 134–136

- randomized algorithm, *continued*
 - quicksort, 191–193, 200 pr., 203 pr.
 - random hashing, 286–290
 - randomized rounding, 1136
 - for searching a sorted compact list, 269 pr.
 - for selection, 230–236, 245 pr.
 - universal hashing, 286–290
 - for weighted majority, 1022 ex.
- RANDOMIZED-HIRE-ASSISTANT, 135
- RANDOMIZED-MARKING, 808
- RANDOMIZED-PARTITION, 192
- RANDOMIZED-PARTITION', 200 pr.
- RANDOMIZED-QUICKSORT, 192
 - relation to randomly built binary search trees, 328 pr.
- randomized rounding, 1136
- RANDOMIZED-SELECT, 230
- randomly built binary search tree, 328 pr.
- RANDOMLY-PERMUTE, 136, 138 ex.
- random-number generator, 129
- random oracle, 276
- random permutation, 136–138
 - uniform, 128, 136
- RANDOM-SAMPLE, 139 ex.
- RANDOM-SEARCH, 154 pr.
- random variable, 1191–1196
 - indicator, *see* indicator random variable
- range, 1162
 - of a matrix, 1224 pr.
- rank
 - column, 1220
 - in computing suffix arrays, 987
 - full, 1220
 - of a matrix, 1220, 1224 pr.
 - of a node in a disjoint-set forest, 528, 533–534, 540 ex.
 - of a number in an ordered set, 480
 - in order-statistic trees, 482–484, 485–486 ex.
 - row, 1220
- rate of growth, 32
- RB-DELETE, 348
- RB-DELETE-FIXUP, 351
- RB-ENUMERATE, 355 ex.
- RB-INSERT, 338
- RB-INSERT-FIXUP, 339
- RB-JOIN, 356 pr.
- RB-TRANSPLANT, 347
- RC6, 304
- reachability in a graph (\leadsto), 1165
- real numbers (\mathbb{R}), 1153
- reconstructing an optimal solution, in dynamic programming, 390
- record, 17, 157
- rectangle, 495 ex.
- RECTANGULAR-MATRIX-MULTIPLY, 374
- recurrence, 39, 76–80, 90–125
 - Akra-Bazzi, 115–119
 - algorithmic, 77–78
 - inequalities in, 78
 - master, 101
 - solution by Akra-Bazzi method, 117–118
 - solution by master method, 101–107
 - solution by recursion-tree method, 95–101
 - solution by substitution method, 90–95
- recursion, 34
- recursion tree, 42, 95–101
 - in matrix-chain multiplication analysis, 388–390
 - in merge sort analysis, 42–44
 - in proof of continuous master theorem, 108–110
 - in quicksort analysis, 188–190
 - in rod cutting analysis, 366–367
 - and the substitution method, 98
- RECURSIVE-ACTIVITY-SELECTOR, 422
- recursive case, 34
 - of a divide-and-conquer algorithm, 76
 - of a recurrence, 77
- RECURSIVE-MATRIX-CHAIN, 389
- red-black properties, 331–332
- red-black tree, 331–359
 - augmentation of, 487–489
 - compared with B-trees, 497, 503
 - deletion from, 346–355
 - for enumerating keys in a range, 355 ex.
 - height of, 332
 - insertion into, 338–346
 - in interval trees, 490–495
 - joining of, 356 pr.
 - left-leaning, 358
 - maximum key of, 334
 - minimum key of, 334
 - in order-statistic trees, 480–486
 - persistent, 355 pr.
 - predecessor in, 334
 - properties of, 331–335

- red-black tree, *continued*
 - relaxed, 334 *ex.*
 - restructuring, 473 *pr.*
 - rotation in, 335–338
 - searching in, 334
 - successor in, 334
 - see also* interval tree, order-statistic tree
- REDUCE, 784 *pr.*
- reducibility, 1061–1063
- reduction algorithm, 1046, 1062
- reduction function, 1062
- reduction, of an array, 784 *pr.*
- reduction strategies, 1095–1098
- reference, 23
- reflexive relation, 1158
- reflexivity of asymptotic notation, 61
- region, feasible, 854
- register, 301, 756
- regret, 1016
- regular graph, 716 *ex.*, 740 *pr.*
- regularity condition, 103, 112, 114 *ex.*
- regularization, 1012, 1036–1037
- reindexing summations, 1143–1144
- reinforcement learning, 1004
- rejection
 - by an algorithm, 1053
 - by a finite automaton, 968
- relation, 1158–1161
- relatively prime, 908
- RELAX, 610
- relaxation
 - of an edge, 609–611
 - linear programming, 1122
- relaxed red-black tree, 334 *ex.*
- release time, 446 *pr.*, 816 *pr.*
- remainder, 64, 905
- remainder instruction, 26
- repeated squaring
 - for all-pairs shortest paths, 652–653
 - for raising a number to a power, 934
- repeat**, in pseudocode, 22
- repetition factor, of a string, 996 *pr.*
- REPETITION-MATCHER, 996 *pr.*
- representative of a set, 520
- RESET, 456 *ex.*
- residual capacity, 677, 681
- residual edge, 678
- residual network, 677–681
- residue, 64, 905, 954 *pr.*
- respecting a set of edges, 587
- return**, in pseudocode, 24
- return instruction, 26
- reweighting
 - in all-pairs shortest paths, 662–664
 - in single-source shortest paths, 641 *pr.*
- $\rho(n)$ -approximation algorithm, 1104, 1120
- RIGHT, 162
- right child, 1173
- right-conversion, 337 *ex.*
- RIGHT-ROTATE, 336
- right rotation, 335
- right shift (\gg), 285
- right subtree, 1173
- rod cutting, 363–373, 393 *ex.*
- root
 - of a tree, 1171
 - of unity, 885–886
 - of \mathbb{Z}_n^* , 932
- rooted tree, 1171
 - representation of, 265–268
- rotation, 335–338
- rounding, 1122
 - randomized, 1136
- row-major order, 253, 396
- row rank, 1220
- row vector, 1215
- RSA public-key cryptosystem, 936–942
- rule of product, 1179
- rule of sum, 1178
- running time, 29
 - asymptotic, 49
 - average-case, 32, 128
 - best-case, 34 *ex.*
 - expected, 32, 129
 - of a graph algorithm, 548
 - order of growth, 32
 - parallel, 757–758
 - and proper use of asymptotic notation, 56–57
 - rate of growth, 32
 - worst-case, 31
- SA, *see* suffix array
- sabermetrics, 415 *n.*
- safe edge, 587
- SAME-COMPONENT, 522

- sample position, 997 pr.
- sample space, 1185
- sample suffix, 997 pr.
- sampling, 139 ex.
- SAT, 1074
- satellite data, 17, 157, 249
- satisfiability, 1066, 1073–1079, 1120–1121, 1124 ex.
- satisfiable formula, 1043, 1074
- satisfying assignment, 1066, 1074
- scalar, 1217
- scalar flow product, 675 ex.
- scaling
 - in maximum flow, 699 pr.
 - in single-source shortest paths, 641 pr.
- scan, 784 pr.
- SCAN, 785 pr.
- scapegoat tree, 358
- schedule, 1133 pr.
- scheduler for task-parallel computations, 753, 759–761, 769 ex., 789
- scheduling, 446 pr., 816 pr., 1102 pr., 1133 pr.
- Schur complement, 825, 839
- Schur complement lemma, 840
- SCRAMBLE-SEARCH, 154 pr.
- seam carving, 412 pr.
- SEARCH, 250
- searching
 - binary search, 44 ex., 777–778
 - in binary search trees, 316–317
 - in B-trees, 504–505
 - in chained hash tables, 278
 - in direct-address tables, 274
 - for an exact interval, 495 ex.
 - in interval trees, 492–494
 - linear search, 25 ex.
 - in linked lists, 260
 - in open-address hash tables, 294
 - in red-black trees, 334
 - in sorted compact lists, 269 pr.
 - of static sets, 308 pr.
 - in an unsorted array, 154 pr.
- search list, *see* linked list
- search tree, *see* balanced search tree, binary search tree, B-tree, exponential search tree, interval tree, optimal binary search tree, order-statistic tree, red-black tree, splay tree, 2-3 tree, 2-3-4 tree
- secondary storage
 - search tree for, 497–519
 - stacks on, 517 pr.
- second-best minimum spanning tree, 599 pr.
- secret key, 936, 939
- SELECT, 237
 - used in quicksort, 241 ex.
- SELECT3, 247 pr.
- selection, 227
 - of activities, 418–425
 - and comparison sorts, 241
 - in order-statistic trees, 481–482
 - randomized, 230–236, 245 pr.
 - in worst-case linear time, 236–243
- selection sort, 33 ex., 53 ex.
- selector vertex, 1087
- self-loop, 1164
- semiconnected graph, 581 ex.
- semiring, 651 n., 669
- sentinel
 - in linked lists, 261–264
 - in red-black trees, 332
- sequence ($()$), 1162
 - bitonic, 644 pr.
 - inversion in, 134 ex., 486 ex.
 - probe, 293
- sequential consistency, 756
- serial algorithm versus parallel algorithm, 748
- serial projection, 750, 753
- series, 1141–1144
 - strands logically in, 756
- series-parallel composition of parallel traces, 762 fig.
- set ($\{ \}$), 1153–1158
 - cardinality ($| |$), 1156
 - collection of, 1156
 - convex, 675 ex.
 - difference ($-$), 1154
 - independent, 1099 pr.
 - intersection (\cap), 1154
 - member (\in), 1153
 - not a member (\notin), 1153
 - partially ordered, 1160
 - static, 308 pr.
 - union (\cup), 1154
- set-covering problem, 1115–1119
 - weighted, 1132 pr.
- set-partition problem, 1098 ex.

- SHA-256, 291
- shared memory, 748
- sharks with lasers, 850
- Shell's sort, 48
- shift
 - left (\lll), 305
 - right (\ggg), 285
 - in string matching, 957
- shift instruction, 27
- short-circuiting operator, 24
- SHORTEST-PATH, 1045
- shortest paths, 604–669
 - all-pairs, 605, 646–669
 - Bellman-Ford algorithm for, 612–616
 - with bitonic shortest paths, 644 pr.
 - and breadth-first search, 558–561, 605
 - convergence property of, 611, 634–635
 - and cycles, 607–608
 - and difference constraints, 626–632
 - Dijkstra's algorithm for, 620–626
 - in a directed acyclic graph, 616–619
 - distance in (δ) , 558
 - in ϵ -dense graphs, 668 pr.
 - estimate of, 609
 - Floyd-Warshall algorithm for, 655–659, 662 ex.
 - Gabow's scaling algorithm for, 641 pr.
 - Johnson's algorithm for, 662–667
 - as a linear program, 861
 - and longest paths, 1042
 - by matrix multiplication, 648–655, 668–669
 - and negative-weight cycles, 606–607, 614–615, 655 ex., 662 ex.
 - with negative-weight edges, 606–607
 - no-path property of, 611, 634
 - optimal substructure of, 605–606, 649, 655–656
 - path-relaxation property of, 611, 635
 - predecessor in (π) , 608
 - predecessor-subgraph property of, 611, 637–638
 - problem variants, 605
 - and relaxation, 609–611
 - by repeated squaring, 652–653
 - single-destination, 605
 - single-pair, 385, 605
 - single-source, 604–645
 - tree of, 608–609, 635–638
 - triangle inequality of, 611, 633
 - in an unweighted graph, 385, 558
 - upper-bound property of, 611, 633–634
 - in a weighted graph, 604
 - weight in (δ) , 604
- shortest remaining processing time (SRPT), 816 pr.
- sibling, 1172
- signature, 938
- simple cycle, 1165–1166
- simple graph, 1166
- simple path, 1165
 - longest, 385, 1042
- SIMPLER-RANDOMIZED-SELECT, 243 pr.
- simplex, 857
- simplex algorithm, 626, 857, 876
- simulation, 173, 181
- single-destination shortest paths, 605
- single-pair shortest path, 385, 605
 - as a linear program, 861
- single-source shortest paths, 604–645
 - Bellman-Ford algorithm for, 612–616
 - with bitonic shortest paths, 644 pr.
 - and difference constraints, 626–632
 - Dijkstra's algorithm for, 620–626
 - in a directed acyclic graph, 616–619
 - in ϵ -dense graphs, 668 pr.
 - Gabow's scaling algorithm for, 641 pr.
 - and longest paths, 1042
- singleton, 1156
- singly connected graph, 572 ex.
- singly linked list, 259
- singular matrix, 1220
- singular value decomposition, 849
- sink vertex, 553 ex., 671, 674
- size
 - of an algorithm's input, 28, 903–904, 1049–1052
 - of a boolean combinational circuit, 1067
 - of a clique, 1081
 - of a group, 917
 - of a set, 1156
 - of a vertex cover, 1084, 1106
- skip list, 359

- slackness
 - complementary, 873 pr.
 - parallel, 758
- slot
 - of a direct-access table, 273
 - of a hash table, 275
- SLOW-APSP, 652
- smoothed analysis, 876
- solution
 - to an abstract problem, 1049
 - to a computational problem, 6
 - to a concrete problem, 1049
 - feasible, 627, 854
 - infeasible, 854
 - optimal, 854
 - to a system of linear equations, 820
- sorted linked list, 259
- sorting, 5, 17–21, 34–44, 51–53, 56–57, 157–226, 775–782
 - bubblesort, 46 pr.
 - bucket sort, 215–219
 - columnsort, 222 pr.
 - comparison sort, 205
 - counting sort, 208–211
 - fuzzy, 203 pr.
 - heapsort, 161–181
 - in place, 158, 220 pr.
 - insertion sort, 12–13, 17–21, 51–53, 56–57
 - k -sorting, 221 pr.
 - lexicographic, 327 pr., 986 n.
 - in linear time, 208–219, 220 pr.
 - lower bounds for, 205–208, 225
 - merge sort, 12–13, 34–44, 57, 775–782
 - by an oblivious compare-exchange algorithm, 222 pr.
 - parallel merge sort, 775–782
 - parallel quicksort, 789 pr.
 - probabilistic lower bound for, 219 pr.
 - quicksort, 182–204
 - radix sort, 211–215
 - selection sort, 33 ex., 53 ex.
 - Shell's sort, 48
 - stable, 210
 - table of running times, 159
 - topological, 573–576
 - using a binary search tree, 326 ex.
 - with variable-length items, 220 pr.
 - 0-1 sorting lemma, 222 pr.
 - sorting network, 789
 - source vertex, 554, 605, 671, 674
 - span, 757
 - span law, 758
 - spanning tree, 585
 - bottleneck, 601 pr.
 - maximum, 1134 pr.
 - verification of, 603
 - see also* minimum spanning tree
 - sparse graph, 549
 - all-pairs shortest paths for, 662–667
 - and Prim's algorithm, 599 pr.
 - sparse matrix, 81
 - spawn**, in pseudocode, 752–754
 - spawning, 753
 - speedup, 758
 - of a randomized parallel algorithm, 789 pr.
 - spindle in a disk drive, 498
 - spine of a string-matching automaton, 970
 - splay tree, 359, 478
 - splicing
 - in a binary search tree, 324–325
 - in a linked list, 260–261
 - spline, 847 pr.
 - splitting
 - of B-tree nodes, 506–508
 - of 2-3-4 trees, 518 pr.
 - splitting summations, 1148–1149
 - spurious hit, 965
 - square matrix, 1215
 - square of a directed graph, 553 ex.
 - square root, modulo a prime, 954 pr.
 - squaring, repeated
 - for all-pairs shortest paths, 652–653
 - for raising a number to a power, 934
 - SRPT (shortest remaining processing time), 816 pr.
 - stability
 - numerical, 819, 821
 - of sorting algorithms, 210
 - stable-marriage problem, 716–723
 - stable matching, 716
 - stable-roommates problem, 723 ex.
 - stack, 254–255
 - implemented by queues, 258 ex.
 - implemented with a priority queue, 178 ex.
 - linked-list implementation of, 264 ex.

- stack, *continued*
 - operations analyzed by accounting method, 454–455
 - operations analyzed by aggregate analysis, 449–451
 - operations analyzed by potential method, 457–458
 - for procedure execution, 202 pr.
 - on secondary storage, 517 pr.
- STACK-EMPTY, 255
- standard deviation, 1195
- standard encoding ($()$), 1052
- standard form of a linear program, 854
- start state, 967
- start time, 418
- state of a finite automaton, 967
- static graph, 522
- static hashing, 282, 284–286
- static set, 308 pr.
- stencil, 787 pr.
- Stirling's approximation, 67
- stochastic gradient descent, 1040 pr.
- STOGE-SORT, 202 pr.
- store instruction, 26, 756
- strand, 754
 - mutually noninterfering, 767
- Strassen's algorithm, 85–90, 124–125
 - parallel algorithm for, 773–774
- streaks, 144–150, 153 ex.
- streaming algorithms, 818
- strict Fibonacci heap, 478
- strictly decreasing, 63
- strictly increasing, 63
- string, 957, 1179
 - interpreted as a key, 290–291, 292 ex.
- string matching, 957–1002
 - based on repetition factors, 996 pr.
 - by finite automata, 967–975
 - with gap characters, 961 ex., 975 ex.
 - Knuth-Morris-Pratt algorithm for, 975–985
 - naive algorithm for, 960–962
 - Rabin-Karp algorithm for, 962–967
 - by suffix arrays, 985–996
- string-matching automaton, 968–975
- strongly connected component, 1166
 - decomposition into, 576–581
- STRONGLY-CONNECTED-COMPONENTS, 577
- strongly connected graph, 1166
- subarray ($:$), 19, 23
- subgraph, 1166
 - equality, 724
 - predecessor, *see* predecessor subgraph
- subgraph-isomorphism problem, 1098 ex.
- subgroup, 921–923
- subpath, 1165
- subproblem graph, 370–371
- subroutine, 23, 26, 29 n.
- subsequence, 394
- subset (\subseteq), 1154, 1156
- SUBSET-SUM, 1092
- subset-sum problem
 - approximation algorithm for, 1124–1130
 - NP-completeness of, 1092–1095
 - with unary target, 1098 ex.
- substitution method, 90–95
 - in quicksort analysis, 191 ex., 193–194
 - and recursion trees, 98
 - in selection analysis, 240–241
- substring, 962, 1179
 - rank of, 987
- subtracting a low-order term, in the substitution method, 92–93
- subtract instruction, 26
- subtraction of matrices, 1218
- subtree, 1172
 - maintaining size of, in order-statistic trees, 484–485
- success, in a Bernoulli trial, 1196
- successor
 - in binary search trees, 318–319
 - finding i th, of a node in an order-statistic tree, 486 ex.
 - in linked lists, 259
 - in red-black trees, 334
- SUCCESSOR, 250
- such that ($:$), 1154
- suffix (\sqsupset), 959
- suffix array (SA), 985–996, 997 pr.
- suffix function, 968
- suffix-function inequality, 971
- suffix-function recursion lemma, 972
- sum (\sum), 1140
 - Cartesian, 885 ex.
 - of matrices, 1217

- sum (Σ), *continued*
 - of polynomials, 877
 - rule of, 1178
 - telescoping, 1143
- SUM-ARRAY, 25 *ex.*
- SUM-ARRAYS, 783 *pr.*
- SUM-ARRAYS', 783 *pr.*
- summation, 1140–1152
 - approximated by integrals, 1150
 - in asymptotic notation, 58, 1141
 - bounding, 1145–1152
 - formulas and properties of, 1140–1145
 - linearity of, 1141
 - lower bounds on, 1148, 1150
 - splitting, 1148–1149
- summation lemma, 887
- supercomputer, 748
- superpolynomial time, 1042
- supersink, 674
- supersource, 674
- supervised learning, 1004
- surjection, 1162
- SVD, 849
- symbol table, 272
- symmetric difference, 706
- symmetric-key cryptosystem, 941
- symmetric matrix, 1217
- symmetric positive-definite matrix, 838–841
 - inverse of, 784 *pr.*
- symmetric relation, 1159
- symmetry of Θ -notation, 61
- sync**, in pseudocode, 752–754
- system of difference constraints, 626–632
- system of linear equations, 784 *pr.*, 819–833, 847 *pr.*, 1034–1035

- TABLE-DELETE, 467
- TABLE-INSERT, 462
- tail
 - of a binomial distribution, 1203–1210
 - of a linked list, 259
 - of a queue, 256
- tail recursion, 202 *pr.*, 422
- target, 1092
- Tarjan's offline lowest-common-ancestors algorithm, 543 *pr.*
- task parallelism, 749
 - see also* parallel algorithm
- Task Parallel Library, 750
- task-parallel scheduling, 759–761, 769 *ex.*
- task scheduling, 446 *pr.*, 816 *pr.*
- tautology, 1060 *ex.*
- Taylor series, 329 *pr.*
- telescoping series, 1143
- telescoping sum, 1143
- termination of a loop invariant, 20
- testing
 - of primality, 942–953, 956
 - of pseudoprimalty, 944–945
- text, 957
- Theta-notation (Θ), 33, 51, 54 *fig.*, 56
- thread, 748
- Threading Building Blocks, 750
- thread parallelism, 748
- 3-CNF, 1076
- 3-CNF-SAT, 1076
- 3-CNF satisfiability, 1076–1079
 - approximation algorithm for, 1120–1121
 - and 2-CNF satisfiability, 1043
- 3-COLOR, 1100 *pr.*
- 3-conjunctive normal form, 1076
- threshold constant, 77
- tight bounds, 56
- time, *see* running time
- time domain, 877
- time-memory trade-off, 367
- timestamp, 564, 571 *ex.*
- $T[i:]$ (suffix of a text), 986
- $T[:k]$ (prefix of a text), 959
- to**, in pseudocode, 22
- top-down method, for dynamic programming, 368
- top of a stack, 254
- topological sort, 573–576
 - in computing single-source shortest paths in a dag, 616
- TOPOLOGICAL-SORT, 573
- total order, 1160
- total path length, 328 *pr.*
- total preorder, 1160
- total relation, 1160
- tour
 - bitonic, 407 *pr.*
 - Euler, 583 *pr.*, 1043
 - of a graph, 1090

- trace, 754–756
 - series-parallel composition of, 762 fig.
- track in a disk drive, 498
- tractability, 1042
- trailing pointer, 321
- training data, 1003
- training phase, 1003
- transition function, 967, 973–974
- transitive closure, 659–661
 - and boolean matrix multiplication, 838 ex.
 - of dynamic graphs, 667 pr., 669
- TRANSITIVE-CLOSURE, 660
- transitive relation, 1159
- transitivity of asymptotic notation, 61
- TRANSPLANT, 324, 346
- transpose
 - conjugate, 838 ex.
 - of a directed graph, 553 ex.
 - of a matrix, 1214
- transpose symmetry of asymptotic notation, 62
- traveling-salesperson problem
 - approximation algorithm for, 1109–1115
 - bitonic euclidean, 407 pr.
 - bottleneck, 1115 ex.
 - NP-completeness of, 1090–1092
 - with the triangle inequality, 1110–1113
 - without the triangle inequality, 1113–1114
- traversal of a tree, 314, 320 ex., 1112
- treap, 358
- tree, 1169–1176
 - AA-trees, 358
 - AVL, 357 pr., 358
 - binary, *see* binary tree
 - bisection of, 1177 pr.
 - breadth-first, 555, 561
 - B-trees, 497–519
 - decision, 206–207, 219 pr.
 - depth-first, 564
 - diameter of, 563 ex.
 - dynamic, 478
 - free, 1167, 1169–1171
 - full walk of, 1112
 - fusion, 226, 478
 - heap, 161–181
 - height-balanced, 357 pr.
 - height of, 1173
 - interval, 489–495
 - k -neighbor, 358
 - left-leaning red-black binary search trees, 358
 - minimum spanning, *see* minimum spanning tree
 - optimal binary search, 400–407
 - order-statistic, 480–486
 - parse, 1077
 - recursion, 42, 95–101
 - red-black, *see* red-black tree
 - rooted, 265–268, 1171
 - scapegoat, 358
 - search, *see* search tree
 - shortest-paths, 608–609, 635–638
 - spanning, *see* minimum spanning tree, spanning tree
 - splay, 359, 478
 - treap, 358
 - 2-3, 358, 519
 - 2-3-4, 502, 518 pr.
 - van Emde Boas, 478
 - walk of, 314, 320 ex., 1112
 - weight-balanced trees, 358
- TREE-DELETE, 325, 326 ex., 346–347
- tree edge, 561, 564, 569
- TREE-INSERT, 321, 338
- TREE-MAXIMUM, 318
- TREE-MINIMUM, 318
- TREE-PREDECESSOR, 319
- TREE-SEARCH, 316
- TREE-SUCCESSOR, 319
- tree walk, 314, 320 ex., 1112
- TRE-QUICKSORT, 202 pr.
- trial division, 943
- triangle inequality, 1110
 - for shortest paths, 611, 633
- triangular matrix, 1216
- trichotomy, interval, 490
- trichotomy property of real numbers, 62
- tridiagonal linear systems, 847 pr.
- tridiagonal matrix, 1216
- trie (radix tree), 327 pr.
- TRIM, 1127
- trimming a list, 1126
- trivial divisor, 904
- tropical semiring, 651 n.
- truth assignment, 1066, 1073
- truth table, 1065
- TSP, 1091

- tuple, 1157
- twiddle factor, 891
- 2-CNF-SAT, 1080 *ex.*
- 2-CNF satisfiability, 1080 *ex.*
 - and 3-CNF satisfiability, 1043
- two-pass method, 529
- 2-3-4 tree, 502, 518 *pr.*
- 2-3 tree, 358, 519

- unary, 1050
- unbounded competitive ratio, 804
- unbounded linear program, 854
- uncle, 340
- unconditional branch instruction, 26
- unconstrained gradient descent, 1023–1031
- uncountable set, 1156
- underdetermined system of linear equations, 820
- underflow
 - of a queue, 256
 - of a stack, 255
- undirected graph, 1164
 - articulation point of, 582 *pr.*
 - biconnected component of, 582 *pr.*
 - bridge of, 582 *pr.*
 - clique in, 1081
 - coloring of, 1100 *pr.*, 1176 *pr.*
 - computing a minimum spanning tree in, 585–603
 - d -regular, 716 *ex.*, 740 *pr.*
 - grid, 697 *pr.*
 - hamiltonian, 1056
 - independent set of, 1099 *pr.*
 - matching in, 693–697, 704–743
 - nonhamiltonian, 1056
 - vertex cover of, 1084, 1106
 - see also* graph
- undirected version of a directed graph, 1167
- uniform family of hash functions, 287
- uniform hash function, 278
- uniform hashing, 295
- uniform probability distribution, 1186–1187
- uniform random permutation, 128, 136
- union
 - of languages, 1052
 - of linked lists, 264 *ex.*
 - of sets (\cup), 1154
- UNION, 264 *ex.*, 521
 - disjoint-set-forest implementation of, 530
 - linked-list implementation of, 524–526
- union by rank, 528
- unit (1), 905
- unit lower-triangular matrix, 1216
- unit upper-triangular matrix, 1216
- unit vector, 1215
- universal family of hash functions, 286–287
- universal hash function, 278
- universal hashing, 286–290, 309 *pr.*
- universal sink, 553 *ex.*
- universe, 273, 1155
- unmatched vertex, 693, 705
- unsorted linked list, 259
- unstable matching, 717
- unsupervised learning, 1004
- until**, in pseudocode, 22
- unweighted longest simple paths, 385
- unweighted shortest paths, 385
- upper bound, 54
- upper-bound property, 611, 633–634
- upper median, 227
- upper-triangular matrix, 1216

- valid shift, 957
- value
 - of a flow, 672
 - of a function, 1161
 - objective, 854
- Vandermonde matrix, 881, 1223 *pr.*
- van Emde Boas tree, 478
- Var [\cdot], *see* variance
- variable
 - decision, 851
 - in pseudocode, 22
 - random, 1191–1196
 - see also* indicator random variable
- variable-length code, 432
- variable-length input
 - interpreted as a key, 290–291
 - to the wee hash function, 306
- variance, 1194
 - of a binomial distribution, 1200
 - of a geometric distribution, 1198
- vector, 1215, 1219–1221
 - convolution of, 880