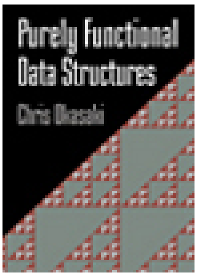

Purely Functional Data Structures

Chris Okasaki

Book DOI: http://dx.doi.org/10.1017/CBO9780511530104

Online ISBN: 9780511530104

Hardback ISBN: 9780521631242

Paperback ISBN: 9780521663502

Chapter

Bibliography pp. 207-216

Chapter DOI: http://dx.doi.org/10.1017/CBO9780511530104.014

Cambridge University Press

Bibliography

[Ada93] Stephen Adams. Efficient sets—a balancing act. *Journal of Functional Programming*, 3(4):553–561, October 1993. (p. 29)

[AFM$^+$95] Zena M. Ariola, Matthias Felleisen, John Maraist, Martin Odersky, and Philip Wadler. A call-by-need lambda calculus. In *ACM Symposium on Principles of Programming Languages*, pages 233–246, January 1995. (p. 37)

[And91] Arne Andersson. A note on searching in a binary search tree. *Software—Practice and Experience*, 21(10):1125–1128, October 1991. (p. 14)

[AVL62] G. M. Adel'son-Vel'skiĭ and E. M. Landis. An algorithm for the organization of information. *Soviet Mathematics–Doklady*, 3(5):1259–1263, September 1962. English translation of Russian orginal appearing in *Doklady Akademia Nauk SSSR*, 146:263-266. (p. 99)

[Bac78] John Backus. Can programming be liberated from the von Neumann style? A functional style and its algebra of programs. *Communications of the ACM*, 21(8):613–641, August 1978. (p. 1)

[BAG92] Amir M. Ben-Amram and Zvi Galil. On pointers versus addresses. *Journal of the ACM*, 39(3):617–648, July 1992. (p. 2)

[BC93] F. Warren Burton and Robert D. Cameron. Pattern matching with abstract data types. *Journal of Functional Programming*, 3(2):171–190, April 1993. (p. 180)

[Bel57] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957. (p. 37)

[BH89] Bror Bjerner and Sören Holmström. A compositional approach to time analysis of first order lazy functional programs. In *Conference on Functional Programming Languages and Computer Architecture*, pages 157–165, September 1989. (p. 82)

[BO96] Gerth Stølting Brodal and Chris Okasaki. Optimal purely functional priority queues. *Journal of Functional Programming*, 6(6):839–857, November 1996. (pp. 140, 170)

[Bro78] Mark R. Brown. Implementation and analysis of binomial queue algorithms. *SIAM Journal on Computing*, 7(3):298–319, August 1978. (pp. 20, 29)

[Bro95] Gerth Stølting Brodal. Fast meldable priority queues. In *Workshop on Algorithms and Data Structures*, volume 955 of *LNCS*, pages 282–290. Springer-Verlag, August 1995. (pp. 170, 184)

[Bro96] Gerth Stølting Brodal. Worst-case priority queues. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 52–58, January 1996. (p. 170)

[BST95] Adam L. Buchsbaum, Rajamani Sundar, and Robert E. Tarjan. Data-structural bootstrapping, linear path compression, and catenable heap-ordered double-ended queues. *SIAM Journal on Computing*, 24(6):1190–1206, December 1995. (p. 169)

[BT95] Adam L. Buchsbaum and Robert E. Tarjan. Confluently persistent deques via data structural bootstrapping. *Journal of Algorithms*, 18(3):513–547, May 1995. (pp. 113, 169, 184)

[Buc93] Adam L. Buchsbaum. *Data-structural bootstrapping and catenable deques*. PhD thesis, Department of Computer Science, Princeton University, June 1993. (pp. 5, 141, 169)

[Bur82] F. Warren Burton. An efficient functional implementation of FIFO queues. *Information Processing Letters*, 14(5):205–206, July 1982. (p. 55)

[But83] T. W. Butler. Computer response time and user performance. In *Conference on Human Factors in Computing Systems*, pages 58–62, December 1983. (p. 83)

[BW88] Richard S. Bird and Philip Wadler. *Introduction to Functional Programming*. Prentice Hall International, 1988. (p. 29)

[CG93] Tyng-Ruey Chuang and Benjamin Goldberg. Real-time deques, multihead Turing machines, and purely functional programming. In *Conference on Functional Programming Languages and Computer Architecture*, pages 289–298, June 1993. (pp. 109, 113)

[CLR90] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. MIT Press, 1990. (p. 27)

[CM95] Richard H. Connelly and F. Lockwood Morris. A generalization of the trie data structure. *Mathematical Structures in Computer Science*, 5(3):381–418, September 1995. (p. 166)

[CMP88] Svante Carlsson, J. Ian Munro, and Patricio V. Poblete. An implicit binomial queue with constant insertion time. In *Scandinavian Workshop on Algorithm Theory*, volume 318 of *LNCS*, pages 1–13. Springer-Verlag, July 1988. (pp. 97, 140)

[Cra72] Clark Allan Crane. *Linear lists and priority queues as balanced binary trees*. PhD thesis, Computer Science Department, Stanford University, February 1972. Available as STAN-CS-72-259. (pp. 18, 29)

[CS96] Seonghun Cho and Sartaj Sahni. Weight biased leftist trees and modified skip lists. In *International Computing and Combinatorics Conference*, pages 361–370, June 1996. (p. 19)

[DGST88] James R. Driscoll, Harold N. Gabow, Ruth Shrairman, and Robert E. Tarjan. Relaxed heaps: An alternative to Fibonacci heaps with applications to parallel computation. *Communications of the ACM*, 31(11):1343–1354, November 1988. (pp. 97, 169)

[Die82] Paul F. Dietz. Maintaining order in a linked list. In *ACM Symposium on Theory of Computing*, pages 122–127, May 1982. (p. 169)

[Die89] Paul F. Dietz. Fully persistent arrays. In *Workshop on Algorithms and Data Structures*, volume 382 of *LNCS*, pages 67–74. Springer-Verlag, August 1989. (pp. 16, 81)

[DR91] Paul F. Dietz and Rajeev Raman. Persistence, amortization and randomization. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 78–88, January 1991. (p. 97)

[DR93] Paul F. Dietz and Rajeev Raman. Persistence, randomization and parallelization: On some combinatorial games and their applications. In *Workshop on Algorithms and Data Structures*, volume 709 of *LNCS*, pages 289–301. Springer-Verlag, August 1993. (p. 97)

[DS87] Paul F. Dietz and Daniel D. Sleator. Two algorithms for maintaining order in a list. In *ACM Symposium on Theory of Computing*, pages 365–372, May 1987. (p. 113)

[DSST89] James R. Driscoll, Neil Sarnak, Daniel D. K. Sleator, and Robert E. Tarjan. Making data structures persistent. *Journal of Computer and System Sciences*, 38(1):86–124, February 1989. (pp. 2, 16, 37, 58, 81)

[DST94] James R. Driscoll, Daniel D. K. Sleator, and Robert E. Tarjan. Fully persistent lists with catenation. *Journal of the ACM*, 41(5):943–959, September 1994. (pp. 81, 169)

[FB97] Manuel Fähndrich and John Boyland. Statically checkable pattern abstractions. In *ACM SIGPLAN International Conference on Functional Programming*, pages 75–84, June 1997. (p. 26)

[FMR72] Patrick C. Fischer, Albert R. Meyer, and Arnold L. Rosenberg. Real-time simulation of multihead tape units. *Journal of the ACM*, 19(4):590–607, October 1972. (p. 113)

[FSST86] Michael L. Fredman, Robert Sedgewick, Daniel D. K. Sleator, and Robert E. Tarjan. The pairing heap: A new form of self-adjusting heap. *Algorithmica*, 1(1):111–129, 1986. (pp. 52, 53, 169)

[FT87] 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, July 1987. (pp. 37, 169)

[FW76] Daniel P. Friedman and David S. Wise. CONS should not evaluate its arguments. In *Automata, Languages and Programming*, pages 257–281, July 1976. (p. 37)

[GMPR77] Leo J. Guibas, Edward M. McCreight, Michael F. Plass, and Janet R. Roberts. A new representation for linear lists. In *ACM Symposium on Theory of Computing*, pages 49–60, May 1977. (pp. 140, 184)

[Gri81] David Gries. *The Science of Programming*. Texts and Monographs in Computer Science. Springer-Verlag, New York, 1981. (p. 55)

[GS78] Leo J. Guibas and Robert Sedgewick. A dichromatic framework for balanced trees. In *IEEE Symposium on Foundations of Computer Science*, pages 8–21, October 1978. (pp. 24, 29, 99)

[GT86] Hania Gajewska and Robert E. Tarjan. Deques with heap order. *Information Processing Letters*, 22(4):197–200, April 1986. (p. 113)

[Hen93] Fritz Henglein. Type inference with polymorphic recursion. *ACM Transactions on Programming Languages and Systems*, 15(2):253–289, April 1993. (p. 170)

[HJ94] Paul Hudak and Mark P. Jones. Haskell vs. Ada vs. C++ vs. ... An experiment in software prototyping productivity, 1994. (p. 1)

[HM76] Peter Henderson and James H. Morris, Jr. A lazy evaluator. In *ACM Symposium on Principles of Programming Languages*, pages 95–103, January 1976. (p. 37)

[HM81] Robert Hood and Robert Melville. Real-time queue operations in pure Lisp. *Information Processing Letters*, 13(2):50–53, November 1981. (pp. 55, 86, 97, 102, 113)

[Hoo82] Robert Hood. *The Efficient Implementation of Very-High-Level Programming Language Constructs.* PhD thesis, Department of Computer Science, Cornell University, August 1982. (Cornell TR 82-503). (p. 113)

[Hoo92] Rob R. Hoogerwoord. A symmetric set of efficient list operations. *Journal of Functional Programming*, 2(4):505–513, October 1992. (pp. 44, 109, 113)

[HU73] John E. Hopcroft and Jeffrey D. Ullman. Set merging algorithms. *SIAM Journal on Computing*, 2(4):294–303, December 1973. (p. 37)

[Hug85] John Hughes. Lazy memo functions. In *Conference on Functional Programming Languages and Computer Architecture*, volume 201 of *LNCS*, pages 129–146. Springer-Verlag, September 1985. (p. 37)

[Hug86] John Hughes. A novel representation of lists and its application to the function "reverse". *Information Processing Letters*, 22(3):141–144, March 1986. (p. 169)

[Hug89] John Hughes. Why functional programming matters. *The Computer Journal*, 32(2):98–107, April 1989. (pp. 1, 113)

[Jon86] Douglas W. Jones. An empirical comparison of priority-queue and event-set implementations. *Communications of the ACM*, 29(4):300–311, April 1986. (p. 56)

[Jos89] Mark B. Josephs. The semantics of lazy functional languages. *Theoretical Computer Science*, 68(1):105–111, October 1989. (p. 37)

[KD96] Anne Kaldewaij and Victor J. Dielissen. Leaf trees. *Science of Computer Programming*, 26(1–3):149–165, May 1996. (p. 118)

[Kin94] David J. King. Functional binomial queues. In *Glasgow Workshop on Functional Programming*, pages 141–150, September 1994. (pp. 29, 82)

[KL93] Chan Meng Khoong and Hon Wai Leong. Double-ended binomial queues. In *International Symposium on Algorithms and Computation*, volume 762 of *LNCS*, pages 128–137. Springer-Verlag, December 1993. (p. 169)

[Knu73a] Donald E. Knuth. *Searching and Sorting*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, 1973. (pp. 18, 29)

[Knu73b} Donald E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming*. Addison-Wesley, 1973. (p. 116)

[KT95] Haim Kaplan and Robert E. Tarjan. Persistent lists with catenation via recursive slow-down. In *ACM Symposium on Theory of*

Computing, pages 93–102, May 1995. (pp. 5, 130, 169, 170, 171, 184, 212)

[KT96a] Haim Kaplan and Robert E. Tarjan. Purely functional lists with catenation via recursive slow-down. Draft revision of [KT95], August 1996. (pp. 171, 184)

[KT96b] Haim Kaplan and Robert E. Tarjan. Purely functional representations of catenable sorted lists. In *ACM Symposium on Theory of Computing*, pages 202–211, May 1996. (pp. 140, 171, 184)

[KTU93] Assaf J. Kfoury, Jerzy Tiuryn, and Pawel Urzyczyn. Type reconstruction in the presence of polymorphic recursion. *ACM Transactions on Programming Languages and Systems*, 15(2):290–311, April 1993. (p. 170)

[Lan65] P. J. Landin. A correspondence between ALGOL 60 and Church's lambda-notation: Part I. *Communications of the ACM*, 8(2):89–101, February 1965. (pp. 37, 113)

[Lau93] John Launchbury. A natural semantics for lazy evaluation. In *ACM Symposium on Principles of Programming Languages*, pages 144–154, January 1993. (p. 37)

[Lia92] Andrew M. Liao. Three priority queue applications revisited. *Algorithmica*, 7(4):415–427, 1992. (p. 56)

[LS81] Benton L. Leong and Joel I. Seiferas. New real-time simulations of multihead tape units. *Journal of the ACM*, 28(1):166–180, January 1981. (p. 113)

[MEP96] Alistair Moffat, Gary Eddy, and Ola Petersson. Splaysort: Fast, versatile, practical. *Software—Practice and Experience*, 26(7):781–797, July 1996. (p. 52)

[Mic68] Donald Michie. "Memo" functions and machine learning. *Nature*, 218:19–22, April 1968. (pp. 3, 37)

[MS91] Bernard M. E. Moret and Henry D. Shapiro. An empirical analysis of algorithms for constructing a minimum spanning tree. In *Workshop on Algorithms and Data Structures*, volume 519 of *LNCS*, pages 400–411. Springer-Verlag, August 1991. (p. 56)

[MT94] David B. MacQueen and Mads Tofte. A semantics for higher-order functors. In *European Symposium on Programming*, pages 409–423, April 1994. (p. 160)

[MTHM97] Robin Milner, Mads Tofte, Robert Harper, and David MacQueen. *The Definition of Standard ML (Revised)*. The MIT Press, Cambridge, Massachusetts, 1997. (p. 31)

[Myc84] Alan Mycroft. Polymorphic type schemes and recursive definitions. In *International Symposium on Programming*, volume 167 of *LNCS*, pages 217–228. Springer-Verlag, April 1984. (pp. 144, 170)

[Mye82] Eugene W. Myers. AVL dags. Technical Report TR82-9, Department of Computer Science, University of Arizona, 1982. (pp. 15, 29)

[Mye83] Eugene W. Myers. An applicative random-access stack. *Information Processing Letters*, 17(5):241–248, December 1983. (pp. 131, 140)

[Mye84] Eugene W. Myers. Efficient applicative data types. In *ACM Symposium on Principles of Programming Languages*, pages 66–75, January 1984. (pp. 15, 29, 169)

[NPP95] Manuel Núñez, Pedro Palao, and Ricardo Peña. A second year course on data structures based on functional programming. In *Functional Programming Languages in Education*, volume 1022 of *LNCS*, pages 65–84. Springer-Verlag, December 1995. (p. 29)

[Oka95a] Chris Okasaki. Amortization, lazy evaluation, and persistence: Lists with catenation via lazy linking. In *IEEE Symposium on Foundations of Computer Science*, pages 646–654, October 1995. (pp. 81, 169)

[Oka95b] Chris Okasaki. Purely functional random-access lists. In *Conference on Functional Programming Languages and Computer Architecture*, pages 86–95, June 1995. (pp. 131, 133, 140)

[Oka95c] Chris Okasaki. Simple and efficient purely functional queues and deques. *Journal of Functional Programming*, 5(4):583–592, October 1995. (pp. 81, 97, 113)

[Oka96a] Chris Okasaki. *Purely Functional Data Structures*. PhD thesis, School of Computer Science, Carnegie Mellon University, September 1996. (p. 34)

[Oka96b] Chris Okasaki. The role of lazy evaluation in amortized data structures. In *ACM SIGPLAN International Conference on Functional Programming*, pages 62–72, May 1996. (pp. 81, 82, 97)

[Oka97] Chris Okasaki. Catenable double-ended queues. In *ACM SIGPLAN International Conference on Functional Programming*, pages 66–74, June 1997. (p. 180)

[OLT94] Chris Okasaki, Peter Lee, and David Tarditi. Call-by-need and continuation-passing style. *Lisp and Symbolic Computation*, 7(1):57–81, January 1994. (p. 37)

[Ove83] Mark H. Overmars. *The Design of Dynamic Data Structures*, volume 156 of *LNCS*. Springer-Verlag, 1983. (pp. 5, 98, 99, 101, 113)

[Pau96] Laurence C. Paulson. *ML for the Working Programmer*. Cambridge University Press, 2nd edition, 1996. (p. x)

[Pet87] Gary L. Peterson. A balanced tree scheme for meldable heaps with updates. Technical Report GIT-ICS-87-23, School of Information and Computer Science, Georgia Institute of Technology, 1987. (p. 169)

[Pip96] Nicholas Pippenger. Pure versus impure Lisp. In *ACM Symposium on Principles of Programming Languages*, pages 104–109, January 1996. (p. 2)

[PPN96] Pedro Palao Gostanza, Ricardo Peña, and Manuel Núñez. A new look at pattern matching in abstract data types. In *ACM SIGPLAN International Conference on Functional Programming*, pages 110–121, May 1996. (p. 180)

[Ram92] Rajeev Raman. *Eliminating Amortization: On Data Structures with Guaranteed Response Times*. PhD thesis, Department of Computer Sciences, University of Rochester, October 1992. (pp. 81, 83, 97)

[Rea92] Chris M. P. Reade. Balanced trees with removals: an exercise in rewriting and proof. *Science of Computer Programming*, 18(2):181–204, April 1992. (p. 29)

[San90] David Sands. Complexity analysis for a lazy higher-order language. In *European Symposium on Programming*, volume 432 of *LNCS*, pages 361–376. Springer-Verlag, May 1990. (p. 82)

[San95] David Sands. A naïve time analysis and its theory of cost equivalence. *Journal of Logic and Computation*, 5(4):495–541, August 1995. (p. 82)

[Sar86] Neil Sarnak. *Persistent Data Structures*. PhD thesis, Department of Computer Sciences, New York University, 1986. (p. 113)

[Sch92] Berry Schoenmakers. *Data Structures and Amortized Complexity in a Functional Setting*. PhD thesis, Eindhoven University of Technology, September 1992. (pp. 41, 55)

[Sch93] Berry Schoenmakers. A systematic analysis of splaying. *Information Processing Letters*, 45(1):41–50, January 1993. (p. 82)

[Sch97] Martin Schwenke. High-level refinement of random access data structures. In *Formal Methods Pacific*, pages 317–318, July 1997. (p. 166)

[SS90] Jörg-Rüdiger Sack and Thomas Strothotte. A characterization of heaps and its applications. *Information and Computation*, 86(1):69–86, May 1990. (p. 118)

[ST85] Daniel D. K. Sleator and Robert E. Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3):652–686, July 1985. (pp. 46, 55, 59)

[ST86a] Neil Sarnak and Robert E. Tarjan. Planar point location using persistent search trees. *Communications of the ACM*, 29(7):669–679, July 1986. (p. 15)

[ST86b] Daniel D. K. Sleator and Robert E. Tarjan. Self-adjusting heaps. *SIAM Journal on Computing*, 15(1):52–69, February 1986. (pp. 37, 55, 59, 169)

[Sta88] John A. Stankovic. Misconceptions about real-time computing: A serious problem for next-generation systems. *Computer*, 21(10):10–19, October 1988. (p. 83)

[Sto70] Hans-Jörg Stoß. K-band simulation von k-Kopf-Turing-maschinen. *Computing*, 6(3):309–317, 1970. (p. 113)

[SV87] John T. Stasko and Jeffrey S. Vitter. Pairing heaps: experiments and analysis. *Communications of the ACM*, 30(3):234–249, March 1987. (p. 56)

[Tar83] Robert E. Tarjan. *Data Structures and Network Algorithms*, volume 44 of *CBMS Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics, Philadelphia, 1983. (p. 81)

[Tar85] Robert E. Tarjan. Amortized computational complexity. *SIAM Journal on Algebraic and Discrete Methods*, 6(2):306–318, April 1985. (pp. 40, 41, 55)

[TvL84] Robert E. Tarjan and Jan van Leeuwen. Worst-case analysis of set union algorithms. *Journal of the ACM*, 31(2):245–281, April 1984. (p. 37)

[Ull94] Jeffrey D. Ullman. *Elements of ML Programming*. Prentice Hall, Englewood Cliffs, New Jersey, 1994. (p. x)

[Vui74] Jean Vuillemin. Correct and optimal implementations of recursion in a simple programming language. *Journal of Computer and System Sciences*, 9(3):332–354, December 1974. (p. 37)

[Vui78] Jean Vuillemin. A data structure for manipulating priority queues. *Communications of the ACM*, 21(4):309–315, April 1978. (pp. 20, 29, 118)

[Wad71] Christopher P. Wadsworth. *Semantics and Pragmatics of the Lamda-Calculus*. PhD thesis, University of Oxford, September 1971. (p. 37)

[Wad87] Philip Wadler. Views: A way for pattern matching to cohabit with data abstraction. In *ACM Symposium on Principles of Programming Languages*, pages 307–313, January 1987. (p. 180)

[Wad88] Philip Wadler. Strictness analysis aids time analysis. In *ACM Symposium on Principles of Programming Languages*, pages 119–132, January 1988. (p. 82)

[WV86] Christopher Van Wyk and Jeffrey Scott Vitter. The complexity of hashing with lazy deletion. *Algorithmica*, 1(1):17–29, 1986. (p. 37)