Bibliography

[AEL88] Andrew W. Appel, John R. Ellis, and Kai Li. Real-time concurrent garbage collection on stock multiprocessors. In *Proceedings of the 1988 SIGPLAN Conference on Programming Language Design and Implementation*, pages 11–20, Atlanta, Georgia, June 1988. ACM Press. Cited on page 137.

[AP87] S. Abraham and J. Patel. Parallel garbage collection on a virtual memory system. In E. Chiricozzi and A. D'Amato, editors, *International Conference on Parallel Processing and Applications*, pages 243–246, L'Aquila, Italy, September 1987. Elsevier North-Holland. Cited on page 146.

[Bak78] Henry G. Baker, Jr. List processing in real time on a serial computer. *Communications of the ACM*, 21(4):280–294, April 1978. Cited on page 136.

[Bak91] Henry G. Baker, Jr. The Treadmill: Real-time garbage collection without motion sickness. In *OOPSLA '91 Workshop on Garbage Collection in Object-Oriented Systems* [OOP91]. Position paper. Also appears as *SIGPLAN Notices 27*(3):66–70, March 1992. Cited on pages 6, 8, 10, 147, and 150.

[BBB+88] Randal E. Bryant, Derek Beatty, Karl Brace, Kyeongsoon Cho, and Thomas Sheffler. Cosmos: A compiled simulator for MOS circuits. In *25 Years of Electronic Design Automation*, pages 496–503, New York, New York, 1988. ACM Press. Cited on page 33.

[BC92] Yves Bekkers and Jacques Cohen, editors. *International Workshop on Memory Management*, number 637 in Lecture Notes in Computer Science, St. Malo, France, September 1992. Springer-Verlag. Cited on pages 319 and 321.

[BDS91] Hans-Juergen Boehm, Alan J. Demers, and Scott Shenker. Mostly parallel garbage collection. In *Proceedings of the 1991 SIGPLAN Conference on Programming Language Design and Implementation* [PLD91], pages 157–164. Cited on pages 137 and 147.

[Bea97] Derek L. Beatty, 1997. personal communication. Cited on page 33.

[Bec82] Leland L. Beck. A dynamic storage allocation technique based on memory residence time. *Communications of the ACM*, 25(10):714–724, October 1982. Cited on page 15.

[Ber88] Steven H. Bergstein. Best-case caching in a symbolic multiprocessor. Bachelor's thesis, Massachusetts Institute of Technology EECS Department, Cambridge, Massachusetts, February 1988. Cited on page 4.

[Bro84] Rodney A. Brooks. Trading data space for reduced time and code space in real-time collection on stock hardware. In *Conference Record of the 1984 ACM Symposium on LISP and Functional Programming* [LFP84], pages 108–113. Cited on pages 137 and 138.

[Che70] C. J. Cheney. A nonrecursive list compacting algorithm. *Communications of the ACM*, 13(11):677–678, November 1970. Cited on page 6.

[CK93] Robert Cmelik and David Keppel. Shade: A fast instruction-set simulator for execution profiling. Technical Report UWCSE 93-06-06, Dept. of Computer Science and Engineering, University of Washington, Seattle, Washington, 1993. Cited on pages 39 and 106.

[Col61] G. O. Collins. Experience in automatic storage allocation. *Communications of the ACM*, 4(10):436–440, October 1961. Cited on pages 12 and 86.

[DDZ93] David Detlefs, Al Dosser, and Benjamin Zorn. Memory allocation costs in large C and C++ programs. Technical Report CU-CS-665-93, University of Colorado at Boulder, Dept. of Computer Science, Boulder, Colorado, August 1993. Cited on pages 4 and 35.

[Den70] Peter J. Denning. Virtual memory. *Computing Surveys*, 3(2):153–189, September 1970. Cited on page 101.

[Det90a] David L. Detlefs. Concurrent garbage collection for C++. Technical Report CMU-CS-90-119, Carnegie-Mellon University, May 1990. Cited on page 137.

[DeT90b] John DeTreville. Experience with concurrent garbage collectors for Modula-2+. Technical Report 64, Digital Equipment Corporation Systems Research Center, Palo Alto, California, August 1990. Cited on page 8.

[DLM$^+$78] Edsger W. Dijkstra, Leslie Lamport, A. J. Martin, C. S. Scholten, and E. F. M. Steffens. On-the-fly garbage collection: An exercise in cooperation. *Communications of the ACM*, 21(11):966–975, November 1978. Cited on pages 7, 8, 141, 144, and 147.

[Ede92] Daniel Ross Edelson. Smart pointers: They're smart, but they're not pointers. In *USENIX C++ Conference*, pages 1–19, Portland, Oregon, August 1992. USENIX Association. Technical Report UCSC-CRL-92-27, University of California at Santa Cruz, Baskin Center for Computer Engineering and Information Sciences, June 1992. Cited on page 161.

[EV91] Steven Engelstad and Jim Vandendorp. Automatic storage management for systems with real time constraints. In *OOPSLA '91 Workshop on Garbage Collection in Object-Oriented Systems* [OOP91]. Position paper. Cited on page 136.

[Fre84] John E. Freund. *Modern Elementary Statistics.* Prentice Hall, Englewood Cliffs, New Jersey, 1984. Cited on page 47.

[FW86] Philip J. Fleming and John J. Wallace. How not to lie with statistics: the correct way to summarize benchmark results. *Communications of the ACM*, 29(3), March 1986. Cited on page 42.

[FY69] Robert R. Fenichel and Jerome C. Yochelson. A LISP garbage-collector for virtual-memory computer systems. *Communications of the ACM*, 12(11):611–612, November 1969. Cited on page 6.

[GZH93] Dirk Grunwald, Benjamin Zorn, and Robert Henderson. Improving the cache locality of memory allocation. In *Proceedings of the 1993 SIGPLAN Conference on Programming Language Design and Implementation*, pages 177–186, Albuquerque, New Mexico, June 1993. ACM Press. Cited on pages 4, 5, 10, 96, and 102.

[Han90] David R. Hanson. Fast allocation and deallocation of memory based on object lifetimes. *Software Practice and Experience*, 20(1), January 1990. Cited on page 37.

[Hay91] Barry Hayes. Using key object opportunism to collect old objects. In Andreas Paepcke, editor, *Conference on Object Oriented Programming Systems, Languages and Applications (OOPSLA '91)*, pages 33–46, Phoenix, Arizona, October 1991. ACM Press. Cited on page 8.

[Hil87] Mark D. Hill. *Aspects of Cache Memory and Instruction Buffer Performance.* PhD thesis, University of California at Berkeley, Berkeley, California, November 1987. Cited on page 106.

[JLS92] Phillip J. Koopman Jr., Peter Lee, and Daniel P. Siewiorek. Cache performance of combinator graph reduction. *ACM Transactions on Programming Languages and Systems*, 14(2):265–297, April 1992. Cited on page 4.

[Joh92] Ralph E. Johnson. Reducing the latency of a real-time garbage collector. *ACM Letters on Programming Languages and Systems*, 1(1):46–58, March 1992. Cited on page 137.

[Kak97] Sheetal V. Kakkad. *Address Translation and Storage Management for Persistent Object Stores.* PhD thesis, The University of Texas at Austin, Austin, Texas, December 1997. Cited on page 161.

[Kin] Chris Kingsley. Implementation of malloc from bsd 4.2 unix. `http://www.plaza.hitachi-sk.co.jp/ftp/FreeBSD/FreeBSD-stable/src/lib/libc/stdlib/malloc.c`. Cited on page 18.

[Kno65] Kenneth C. Knowlton. A fast storage allocator. *Communications of the ACM*, 8(10):623–625, October 1965. Cited on pages 24 and 25.

[Knu73] Donald E. Knuth. *The Art of Computer Programming*, volume 1: Fundamental Algorithms. Addison-Wesley, Reading, Massachusetts, 1973. First edition published in 1968. Cited on pages 15, 21, 22, 23, 45, and 93.

[Kuh70] Thomas S. Kuhn. *The Structure of Scientific Revolutions (Second Edition, Enlarged)*. University of Chicago Press, Chicago, Illinois, 1970. Cited on page 14.

[KV85] David G. Korn and Kiem-Phong Vo. In search of a better malloc. In *Proc. USENIX Summer 1985*, pages 489–506, Portland, Oregon, June 1985. USENIX Association. Cited on pages 27 and 94.

[Lea] Doug Lea. Implementations of malloc. See also the short paper on the implementation of this allocator. Available at `www://g.oswego.edu`. Cited on page 95.

[LFP84] *Conference Record of the 1984 ACM Symposium on LISP and Functional Programming*, Austin, Texas, August 1984. ACM Press. Cited on pages 317 and 319.

[LH83] Henry Lieberman and Carl Hewitt. A real-time garbage collector based on the lifetimes of objects. *Communications of the ACM*, 26(6):419–429, June 1983. Cited on pages 8, 9, and 162.

[McC60] John McCarthy. Recursive functions of symbolic expressions and their computation by machine. *Communications of the ACM*, 3(4):184–195, April 1960. Cited on page 6.

[MH95] B. Magnusson and R. Henriksson. Garbage collection for control systems. In Henry Baker, editor, *International Workshop on Memory Management*, Lecture Notes in Computer Science, Scotland, September 1995. Springer-Verlag. Cited on page 138.

[Moo84] David Moon. Garbage collection in a large Lisp system. In *Conference Record of the 1984 ACM Symposium on LISP and Functional Programming* [LFP84], pages 235–246. Cited on pages 9 and 162.

[Nil88] Kelvin Nilsen. Garbage collection of strings and linked data structures in real time. *Software Practice and Experience*, 18(7):613–640, July 1988. Cited on page 136.

[NOPH92] Scott Nettles, James O'Toole, David Pierce, and Nicholas Haines. Replication-based incremental copying collection. In Bekkers and Cohen [BC92], pages 357–364. Cited on pages 137 and 140.

[NS90] Kelvin Nilsen and William J. Schmidt. A high-level overview of hardware assisted real-time garbage collection. Technical Report TR 90-18a, Dept. of Computer Science, Iowa State University, Ames, Iowa, 1990. Cited on page 137.

[Nut87] Peter R. Nuth. Communication patterns in a symbolic multiprocessor. Technical Report MIT/LCS/TR-395, Massachusetts Institute of Technology Laboratory for Computer Science, Cambridge, Massachusetts, June 1987. Cited on page 4.

[OOP91] *OOPSLA '91 Workshop on Garbage Collection in Object-Oriented Systems*, October 1991. available at `ftp://ftp.cs.utexas.edu/pub/garbage/GC91`. Cited on pages 316, 317, and 322.

[PH86] Ivor P. Page and Jeff Hagins. Improving the performance of buddy systems. *IEEE Transactions on Computers*, C-35(5):441–447, May 1986. Cited on page 25.

[PH96] David A. Patterson and John L. Hennessy. *Computer Architecture: A Quantitative Approach.* Morgan Kaufmann, 1996. 2nd Edition. Cited on pages 4, 42, 98, 99, and 100.

[PLD91] *Proceedings of the 1991 SIGPLAN Conference on Programming Language Design and Implementation*, Toronto, Ontario, June 1991. ACM Press. Published as *SIGPLAN Notices 26*(6), June 1992. Cited on page 316.

[PN77] J. L. Peterson and T. A. Norman. Buddy systems. *Communications of the ACM*, 20(6):421–431, June 1977. Cited on page 24.

[PS89] C.-J. Peng and Gurindar S. Sohi. Cache memory design considerations to support languages with dynamic heap allocation. Technical Report 860, Computer Sciences Dept. University of Wisconsin, Madison, Wisconsin, July 1989. Cited on page 4.

[PSC71] P. W. Purdom, S. M. Stigler, and Tat-Ong Cheam. Statistical investigation of three storage allocation algorithms. *BIT*, 11:187–195, 1971. Cited on page 29.

[RK68] Brian Randell and C. J. Kuehner. Dynamic storage allocation systems. *Communications of the ACM*, 12(7):297–306, May 1968. Cited on page 2.

[Sha88] Robert A. Shaw. *Empirical Analysis of a Lisp System.* PhD thesis, Stanford University, Palo Alto, California, February 1988. Technical Report CSL-TR-88-351, Stanford University Computer Systems Laboratory. Cited on page 8.

[SKW92] Vivek Singhal, Sheetal V. Kakkad, and Paul R. Wilson. Texas: an efficient, portable persistent store. In Antonio Albano and Ron Morrison, editors, *Fifth International Workshop on Persistent Object Systems*, pages 11–33, San Miniato, Italy, September 1992. Springer-Verlag. Cited on page 28.

[Smi82] Alan J. Smith. Cache memories. *Computing Surveys*, 14(3):473–530, September 1982. Cited on page 38.

[Ss92] Bjarne Stroustrup. *The C++ Programming Language.* Addison-Wesley, Reading, Massachusetts, 1992. Cited on pages 135 and 161.

[ST85] Daniel Dominic Sleator and Robert Endre Tarjan. Self-adjusting binary search trees. *Journal of the ACM*, 32(3), 1985. Cited on page 102.

[Sta80] Thomas Standish. *Data Structure Techniques.* Addison-Wesley, Reading, Massachusetts, 1980. Cited on pages 18, 22, and 95.

[Ste75] Guy L. Steele Jr. Multiprocessing compactifying garbage collection. *Communications of the ACM*, 18(9):495–508, September 1975. Cited on page 147.

[Ste83] C. J. Stephenson. Fast fits: New methods for dynamic storage allocation. In *Proceedings of the Ninth Symposium on Operating Systems Principles*, pages 30–32, Bretton Woods, New Hampshire, October 1983. ACM Press. Published as *Operating Systems Review 17*(5), October 1983. Cited on pages 18, 22, 27, and 95.

[Tad78] M. Tadman. Fast-fit: A new hierarchical dynamic storage allocation technique. Master's thesis, UC Irvine, Computer Science Dept., 1978. Cited on page 18.

[Tec97] Solid State Technology. W.s.t.s.: Chip sales back on track, good 4th quarter will usher in 1997 growth, 1997. available at `http://www.solid-state.com/sst/Newsitems/9612/news01.html`. Cited on page 11.

[UM97] Richard A. Uhlig and Trevor N. Mudge. Trace-driven memory simulation: A survey. *Computing Surveys*, 29(2):128–170, June 1997. Cited on page 38.

[Ung84] David M. Ungar. Generation scavenging: A non-disruptive high-performance storage reclamation algorithm. In *ACM SIGSOFT/SIGPLAN Software Engineering Symposium on Practical Software Development Environments*, pages 157–167. ACM Press, April 1984. Published as *ACM SIGPLAN Notices 19*(5), May, 1987. Cited on pages 8, 9, and 162.

[Vo95] Kiem-Phong Vo. Vmalloc: A general and efficient memory allocator. *Software Practice and Experience*, 1995. To appear. Cited on page 4.

[Vui80] Jean Vuillemin. A unifying look at data structures. *Communications of the ACM*, 29(4):229–239, April 1980. Cited on page 18.

[Wan89] Thomas Wang. MM garbage collector for C++. Master's thesis, California Polytechnic State University, San Luis Obispo, California, October 1989. Cited on pages 6, 10, and 147.

[Wil] Paul R. Wilson. Garbage collection. Expanded version of [Wil92]. In revision for ACM Computing Surveys. Draft available at `http://www.cs.utexas.edu/users/oops`. Cited on page 136.

[Wil88] Paul R. Wilson. Opportunistic garbage collection. *SIGPLAN Notices*, 23(12):98–102, December 1988. Cited on page 164.

[Wil92] Paul R. Wilson. Uniprocessor garbage collection techniques. In Bekkers and Cohen [BC92], pages 1–42. Cited on pages 9, 146, 162, and 321.

[Wis78] David S. Wise. The double buddy-system. Technical Report 79, Computer Science Department, Indiana University, Bloomington, Indiana, December 1978. Cited on page 25.

[Wit91] P. T. Withington. How real is "real time" garbage collection? In *OOPSLA '91 Workshop on Garbage Collection in Object-Oriented Systems* [OOP91]. Position paper. Cited on page 136.

[WJ93] Paul R. Wilson and Mark S. Johnstone. Truly real-time non-copying garbage collection. In *OOPSLA '93 Workshop on Memory Management and Garbage Collection*, December 1993. Expanded version of workshop position paper submitted for publication. Cited on page 162.

[WJNB95] Paul R. Wilson, Mark S. Johnstone, Michael Neely, and David Boles. Dynamic storage allocation: A survey and critical review. In *1995 International Workshop on Memory Management*, Kinross, Scotland, UK, 1995. Springer Verlag LNCS. Cited on pages 19, 86, and 95.

[WLM90] Paul R. Wilson, Michael S. Lam, and Thomas G. Moher. Caching consideration for generational garbage collection: A case for large and set-associative caches. Technical Report UIC-EECS-90-5, University of Illinois at Chicago EECS Dept., Chicago, Illinois, December 1990. Cited on page 4.

[WLM92] Paul R. Wilson, Michael S. Lam, and Thomas G. Moher. Caching considerations for generational garbage collection. In *Conference Record of the 1992 ACM Symposium on LISP and Functional Programming*, pages 32–42, San Francisco, California, June 1992. ACM Press. Cited on page 4.

[WM89] Paul R. Wilson and Thomas G. Moher. Design of the Opportunistic Garbage Collector. In *Conference on Object Oriented Programming Systems, Languages and Applications (OOPSLA '89) Proceedings*, pages 23–35, New Orleans, Louisiana, 1989. ACM Press. Cited on page 9.

[Yua90] Taichi Yuasa. Real-time garbage collection on general-purpose machines. *Journal of Systems and Software*, 11:181–198, 1990. Cited on pages 144 and 146.

[ZG92] Benjamin Zorn and Dirk Grunwald. Empirical measurements of six allocation-intensive C programs. Technical Report CU-CS-604-92, University of Colorado at Boulder, Dept. of Computer Science, July 1992. Cited on pages 4 and 35.

[ZG94] Benjamin Zorn and Dirk Grunwald. Evaluating models of memory allocation. *ACM Transactions on Modeling and Computer Simulation*, 1(4):107–131, 1994. Cited on pages 19 and 28.

[Zor90] Benjamin Zorn. Comparing mark-and-sweep and stop-and-copy garbage collection. In *Conference Record of the 1990 ACM Symposium on LISP and Functional Programming*, pages 87–98, Nice, France, June 1990. ACM Press. Cited on page 8.

[Zor91] Benjamin Zorn. The effect of garbage collection on cache performance. Technical Report CU-CS-528-91, University of Colorado at Boulder, Dept. of Computer Science, Boulder, Colorado, May 1991. Cited on page 4.

[Zor93] Benjamin Zorn. The measured cost of conservative garbage collection. *Software—Practice and Experience*, 23(7):733–756, July 1993. Cited on pages 37 and 38.

Vita

Mark Stuart Johnstone was born in San Diego, California on April 13, 1965, the son of Hubert Crosby Johnstone and Margery Schurr Johnstone. After completing his work at Clairemont High School, San Diego, California, in 1983, he attended the University of California at Davis. He received the degree of Bachelor of Science with Honors from the University of California at Davis in June 1987. During the following three years, he was employed as a Scientific Programmer for the Lockheed Missiles and Space Company in the Space Systems Division, Sunnyvale, California. In August 1990 he entered the Graduate School of the University of Texas. He received the degree of Master of Science in Computer Science in May 1992. He has been employed by the Motorola Corporation since June 1996.

Permanent Address: 3721 Mt. Sandy Dr.
San Diego, California 92117

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin.