

Publications related to Chez Scheme

- [1] Andrew W. Keep and R. Kent Dybvig. Automatic cross-library optimization. In *Scheme 2013: Workshop on Scheme and Functional Programming*, 2013. Describes how Chez Scheme’s expander and source optimizer collaborate to perform constant propagation and procedure inlining across library boundaries
full text: <http://www.cs.indiana.edu/~dyb/pubs/auto-xlib-opt.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/auto-xlib-opt-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/auto-xlib-opt.bib>
- [2] Andrew W. Keep and R. Kent Dybvig. A nanopass framework for commercial compiler development. In *Proceedings of the 18th ACM SIGPLAN International Conference on Functional Programming*, 343–350, 2013. A description of the Nanopass infrastructure used by the Chez Scheme Compiler starting with Version 9
full text: <http://www.cs.indiana.edu/~dyb/pubs/commercial-nanopass.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/commercial-nanopass-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/commercial-nanopass.bib>
- [3] Andrew W. Keep, Alex Hearn, and R. Kent Dybvig. Optimizing closures in $o(0)$ time. In *Scheme 2012: Workshop on Scheme and Functional Programming*, 2012. Describes how Chez Scheme’s compiler reduces the number and sizes of closures created at run time
full text: <http://www.cs.indiana.edu/~dyb/pubs/closureopt.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/closureopt-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/closureopt.bib>
- [4] Andrew W. Keep and R. Kent Dybvig. A sufficiently smart compiler for procedural records. In *Scheme 2012: Workshop on Scheme and Functional Programming*, 2012. Describes how Chez Scheme’s compiler optimizes record operations into fully optimal inline memory operations even, in many cases, when defined using the run-time record-type, constructor, predicate, accessor, and mutator operations
full text: <http://www.cs.indiana.edu/~dyb/pubs/records.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/records-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/records.bib>
- [5] Andrew W. Keep and R. Kent Dybvig. Ftypes: Structured foreign types. In *Scheme 2010: Workshop on Scheme and Functional Programming*, 2010. Introduces the mechanism for defining and accessing foreign objects, e.g., C structs, from Chez Scheme programs
full text: <http://www.cs.indiana.edu/~dyb/pubs/ftypes.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/ftypes-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/ftypes.bib>
- [6] Andrew W. Keep and R. Kent Dybvig. Enabling cross-library optimization and compile-time error checking in the presence of procedural macros. In *Scheme 2010: Workshop on Scheme and Functional Programming*, 2010. Introduces the library-group form and how Chez Scheme optimizes across library and program boundaries within a library group
full text: <http://www.cs.indiana.edu/~dyb/pubs/library-groups.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/library-groups-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/library-groups.bib>
- [7] R. Kent Dybvig. *Chez Scheme Version 8 User’s Guide*. Cadence Research Systems, 2010. User’s guide and reference manual for Chez Scheme Version 8. Complements [9].
full text: <http://www.scheme.com/csug8/>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/csug8.bib>
- [8] Abdulaziz Ghuloum and R. Kent Dybvig. Fixing letrec (reloaded). In *Proceedings of the 2009 Workshop on Scheme and Functional Programming*, 57–65, 2009. Describes how Chez Scheme and Ikarus handle `letrec` and `letrec*` expressions efficiently while taking advantage of the R6RS prohibition on invoking the continuation of a right-hand side more than once.

full text: <http://www.cs.indiana.edu/~dyb/pubs/letrec-reloaded.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/letrec-reloaded-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/letrec-reloaded.bib>

- [9] R. Kent Dybvig. *The Scheme Programming Language*, fourth edition. MIT Press, 2009. Introduction and reference manual for R6RS Scheme with numerous short and extended examples and exercises.
full text: <http://www.scheme.com/tspl4/>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/tspl4.bib>
- [10] Michael D. Adams and R. Kent Dybvig. Efficient nondestructive equality checking for trees and graphs. In *Proceedings of the 13th ACM SIGPLAN International Conference on Functional Programming*, 179–188, 2008. An efficient algorithm for the nonterminating equal? R6RS procedure
full text: <http://www.cs.indiana.edu/~dyb/pubs/equal.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/equal-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/equal.bib>
- [11] Abdulaziz Ghuloum and R. Kent Dybvig. Implicit phasing for R6RS libraries. In *Proceedings of the 12 ACM SIGPLAN International Conference on Functional Programming*, 303–314, 2007. A description of the implicit phasing model for R6RS libraries, adopted by Chez Scheme, Ikarus, and various other Scheme systems
full text: <http://www.cs.indiana.edu/~dyb/pubs/implicit-phasing.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/implicit-phasing-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/implicit-phasing.bib>
- [12] R. Kent Dybvig. Syntactic abstraction: the syntax-case expander. In Andy Oram and Greg Wilson, editors, *Beautiful Code: Leading Programmers Explain How They Think*, chapter 25, 407–428. O’Reilly and Associates, June 2007. Provides a description and examples of the syntax-case expansion algorithm
full text: <http://www.cs.indiana.edu/~dyb/pubs/bc-syntax-case.pdf>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/bc-syntax-case.bib>
- [13] R. Kent Dybvig. The development of Chez Scheme. In *Proceedings of the Eleventh ACM SIGPLAN International Conference on Functional Programming*, 1–12, September 2006. A brief history of Chez Scheme’s development
full text: <http://www.cs.indiana.edu/~dyb/pubs/hocs.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/hocs-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/hocs.bib>
- [14] R. Kent Dybvig. *Chez Scheme Version 7 User’s Guide*. Cadence Research Systems, 2005. User’s guide and reference manual for Chez Scheme Version 7. Complements [16].
full text: <http://www.scheme.com/csug7/>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/csug7.bib>
- [15] Oscar Waddell, Dipanwita Sarkar, and R. Kent Dybvig. Fixing letrec: A faithful yet efficient implementation of Scheme’s recursive binding construct. *Higher-order and symbolic computation*, 18(3/4):299–326, 2005. Describes how Chez Scheme handles `letrec` expressions efficiently and with full enforcement of the revised report’s restriction preventing evaluation of left-hand-side variable references and assignments before the right-hand sides have been evaluated.
full text: <http://www.cs.indiana.edu/~dyb/pubs/fixing-letrec.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/fixing-letrec-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/fixing-letrec.bib>
- [16] R. Kent Dybvig. *The Scheme Programming Language*, third edition. MIT Press, 2003. Introduction and reference manual for ANSI Standard Scheme with numerous short and extended examples and exercises.
full text: <http://www.scheme.com/tspl3/>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/tspl3.bib>

- [17] Oscar Waddell and R. Kent Dybvig. Extending the scope of syntactic abstraction. In *Conference Record of the Twenty Sixth Annual ACM Symposium on Principles of Programming Languages*, 203–213, January 1999. Describes the Chez Scheme module system and its interaction with the syntax-case expander.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/popl99.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/popl99-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/popl99.bib>
- [18] R. Kent Dybvig. *Chez Scheme User's Guide*. Cadence Research Systems, 1998. User's guide and reference manual for Chez Scheme. Complements [16].
 full text: <http://www.scheme.com/csug/>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/csug.bib>
- [19] Robert G. Burger and R. Kent Dybvig. An infrastructure for profile-driven dynamic recompilation. In *Proceedings of the IEEE Computer Society 1998 International Conference on Computer Languages*, 240–251, May 1998. Describes support for dynamic recompilation based on information gathered by an edge-count profiler. Only the profiling support is implemented in Chez Scheme.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/pdrtc.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/pdrtc-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/pdrtc.bib>
- [20] Oscar Waddell and R. Kent Dybvig. Fast and effective procedure inlining. In *Proceedings of the Fourth International Symposium on Static Analysis*, volume 1302 of *Lecture Notes in Computer Science*, 35–52. Springer-Verlag, September 1997. Describes a source optimization pass used in Chez Scheme to perform several optimizations, including procedure inlining.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/inlining.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/inlining-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/inlining.bib>
- [21] R. Kent Dybvig. *The Scheme Programming Language*, second edition. Prentice Hall, 1996. Second edition of [16].
 full text: <http://www.scheme.com/tspl2/>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/tspl2.bib>
- [22] Carl Bruggeman, Oscar Waddell, and R. Kent Dybvig. Representing control in the presence of one-shot continuations. In *Proceedings of the SIGPLAN '96 Conference on Programming Language Design and Implementation*, 99–107, May 1996. Describes the implementation of call/cc.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/call1cc.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/call1cc-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/call1cc.bib>
- [23] Robert G. Burger and R. Kent Dybvig. Printing floating-point numbers quickly and accurately. In *Proceedings of the SIGPLAN '96 Conference on Programming Language Design and Implementation*, 108–116, May 1996. Describes an algorithm for printing floating point numbers accurately yet with the minimum number of digits.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/FP-Printing-PLDI96.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/FP-Printing-PLDI96-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/FP-Printing-PLDI96.bib>
- [24] Eric Hilsdale, J. Michael Ashley, R. Kent Dybvig, and Daniel P. Friedman. Compiler construction using Scheme. In P. H. Hartel and M. J. Plasmeijer, editors, *Functional programming languages in education (FPLE)*, LNCS 10 22, 251–268, Nijmegen, The Netherlands, Dec 1995. Springer-Verlag, Heidelberg. Describes a course on compiler construction and sketches a simple compiler model similar in principle to Chez Scheme's.
 full text: <http://www.cs.indiana.edu/~dyb/pubs/fple95.pdf>.
 abstract: <http://www.cs.indiana.edu/~dyb/pubs/fple95-abstract.html>.
 bibtex: <http://www.cs.indiana.edu/~dyb/pubs/fple95.bib>

- [25] Robert G. Burger, Oscar Waddell, and R. Kent Dybvig. Register allocation using lazy saves, eager restores, and greedy shuffling. In *Proceedings of the SIGPLAN '95 Conference on Programming Language Design and Implementation*, 130–138, June 1995. Describes a register allocation mechanism used by Chez Scheme.
full text: <http://www.cs.indiana.edu/~dyb/pubs/Reg-Alloc-PLDI95.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/Reg-Alloc-PLDI95-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/Reg-Alloc-PLDI95.bib>
- [26] Cadence Research Systems. *Chez Scheme System Manual, Rev. 2.5*, October 1994. Describes various aspects of Chez Scheme not covered in [41]. Obsoleted by the Chez Scheme User’s Guide [18].
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/cssm.bib>
- [27] J. Michael Ashley and R. Kent Dybvig. An efficient implementation of multiple return values in Scheme. In *Proceedings of the 1994 ACM Conference on Lisp and Functional Programming*, 140–149, June 1994. Describes Chez Scheme’s implementation of the Scheme multiple return values interface.
full text: <http://www.cs.indiana.edu/~dyb/pubs/mrvs.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/mrvs-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/mrvs.bib>
- [28] R. Kent Dybvig, David Eby, and Carl Bruggeman. Don’t stop the BiBOP: Flexible and efficient storage management for dynamically-typed languages. Technical Report 400, Indiana Computer Science Department, March 1994. Describes Chez Scheme’s storage management system.
full text: <http://www.cs.indiana.edu/~dyb/pubs/bibop.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/bibop-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/bibop.bib>
- [29] R. Kent Dybvig, Robert Hieb, and Carl Bruggeman. Syntactic abstraction in Scheme. *Lisp and Symbolic Computation*, 5(4):295–326, 1993. Describes the `syntax-case` macro system and its implementation and argues why it is superior to previous attempts at hygienic macro systems.
full text: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-5-4-pp295-326.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-5-4-pp295-326-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-5-4-pp295-326.bib>
- [30] R. Kent Dybvig, Carl Bruggeman, and David Eby. Guardians in a generation-based garbage collector. In *Proceedings of the SIGPLAN '93 Conference on Programming Language Design and Implementation*, 207–216, June 1993. Introduces guardians and describes the implementation of guardians and weak pairs in Chez Scheme.
full text: <http://www.cs.indiana.edu/~dyb/pubs/guardians-pldi93.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/guardians-pldi93-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/guardians-pldi93.bib>
- [31] R. Kent Dybvig. Writing hygienic macros in Scheme with `syntax-case`. Technical Report 356, Indiana Computer Science Department, June 1992. Introduces the `syntax-case` macro system through a series of examples. Some overlap with [29].
full text: <http://www.cs.indiana.edu/~dyb/pubs/tr356.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/tr356-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/tr356.bib>
- [32] R. Kent Dybvig and Robert Hieb. A new approach to procedures with variable arity. *Lisp and Symbolic Computation*, 3(3):229–244, September 1990. Describes an alternative to the “dot” notation for defining procedures that accept variable numbers of arguments; only the multiple-case notion is adopted in Chez Scheme (as `case-lambda`).
full text: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-3-3-pp229-244.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-3-3-pp229-244-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-3-3-pp229-244.bib>

- [33] Cadence Research Systems, Bloomington, Indiana. *Chez Scheme System Manual, Rev. 2.0*, December 1990.
- [34] Robert Hieb, R. Kent Dybvig, and Carl Bruggeman. Representing control in the presence of first-class continuations. In *Proceedings of the SIGPLAN '90 Conference on Programming Language Design and Implementation*, 66–77, June 1990. Describes Chez Scheme’s segmented stack representation of continuations.
full text: <http://www.cs.indiana.edu/~dyb/pubs/stack.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/stack-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/stack.bib>
- [35] R. Kent Dybvig, Robert Hieb, and Tom Butler. Destination-driven code generation. Technical Report 302, Indiana Computer Science Department, February 1990. Describes the basic mechanism used by Chez Scheme’s code generator to generate code from abstract syntax trees.
full text: <http://www.cs.indiana.edu/~dyb/pubs/ddcg.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/ddcg-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/ddcg.bib>
- [36] Cadence Research Systems, Bloomington, Indiana. *Chez Scheme System Manual*, August 1989.
- [37] R. Kent Dybvig and Robert Hieb. Engines from continuations. *Computer Languages*, 14(2):109–123, 1989. Describes the continuation-based engine implementation used by Chez Scheme.
full text: <http://www.cs.indiana.edu/~dyb/pubs/engines.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/engines-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/engines.bib>
- [38] R. Kent Dybvig and Robert Hieb. A variable-arity procedural interface. In *Proceedings of the 1988 ACM Conference on Lisp and Functional Programming*, 106–115, July 1988. Preliminary version of [32].
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/arityprelim.bib>
- [39] R. Kent Dybvig, Daniel P. Friedman, and Christopher T. Haynes. Expansion-passing style: A general macro mechanism. *Lisp and Symbolic Computation*, 1(1):53–75, 1988. Describes the low-level macro mechanism provided by Chez Scheme Versions 2–4.
full text: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-1-1-pp53-75.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-1-1-pp53-75-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/LaSC-1-1-pp53-75.bib>
- [40] R. Kent Dybvig. *Three Implementation Models for Scheme*. PhD thesis, University of North Carolina Technical Report #87-011, Chapel Hill, April 1987. Chapter 4 describes the essence of the Chez Scheme Version 1 run-time architecture.
full text: <http://www.cs.indiana.edu/~dyb/pubs/3imp.pdf>.
abstract: <http://www.cs.indiana.edu/~dyb/pubs/3imp-abstract.html>.
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/3imp.bib>
- [41] R. Kent Dybvig. *The Scheme Programming Language*. Prentice-Hall, 1987. First edition of [16].
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/tspl.bib>
- [42] R. Kent Dybvig, Daniel P. Friedman, and Christopher T. Haynes. Expansion-passing style: Beyond conventional macros. In *Proceedings of the 1986 ACM Conference on Lisp and Functional Programming*, 143–150, 1986. Preliminary version of [39].
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/epsprelim.bib>
- [43] R. Kent Dybvig and Bruce T. Smith. *Chez Scheme Reference Manual, Version 1.0*. Cadence Research Systems, Chapel Hill, North Carolina, May 1985. Precursor to [41].
bibtex: <http://www.cs.indiana.edu/~dyb/pubs/csrn.bib>

Papers published by the Association for Computing Machinery are made available by permission: <http://www.cs.indiana.edu/~dyb/pubs/acmcopyright.html> from ACM, Inc.