

Lisp Implementations

Walter van Roggen
Artificial Intelligence Technology Group
Digital Equipment Corporation
290 Donald Lynch Blvd, Marlborough, MA, 01752, DLB5-2/B10
vanroggen@hudson.dec.com

At last some benchmark numbers are available for several machines and implementations. I've tried to get the latest numbers for each implementation. Unfortunately, some vendors are less than enthusiastic about releasing any numbers at all, so they are either not listed here or their numbers were taken from reliable sources. If you want to see numbers for other implementations, contact the vendors to get them to send their latest for publication here.

The benchmark times are in cpu seconds. The geometric mean listed at the bottom of each column is of the ratios of individual times for each implementation compared to the time for VAX LISP V2.2 on a VAX-11/780. [I had to choose something as the standard, and most vendors compare with the VAX-11/780 at one time or another, despite its age.]

I believe all the numbers were taken on machines with sufficient memory to avoid significant paging. For VAX LISP we've found that the "knee" in the curve of the graph relating paging to memory size is less than 1 Megabyte for the benchmarks as a whole.

- The Symbolics implementation was running release 7.1.
- The TI implementation was running release 3.0.
- The Lucid Common Lisp for the VAX implementation was running V1.0 on VMS V4.2.
- The VAX LISP implementation was running V2.2 on VMS V4.6.
- The Sun implementation was Sun Common Lisp 2.1, running on 4.2bsd release 3.2.
- The Apollo implementation was DOMAIN/Common Lisp.

Benchmark	Symbolics	TI Explorer	DEC MicroVAX-II	
	3650	I	LucidCL	VAX LISP
Tak	0.45	1.15	0.99	1.16
Stak	2.31	5.21	3.68	4.70
Ctak	5.47	2.42	3.78	9.91
Takl	4.81	8.79	3.69	7.12
Takr	0.45	1.18	1.50	2.51
Boyer	8.63	19.56	28.00	44.27
Browse	11.17	45.70	49.44	52.80
Destru	1.25	2.97	5.26	4.08
Trav-Init	6.59	15.70	11.53	14.69
Trav-Run	36.69	94.52	76.78	129.00
Deriv	2.68	5.69	11.59	12.28
DDeriv	2.55	5.98	15.36	21.24
Div2-Iter	1.07	2.49	3.92	4.56
Div2-Recur	2.02	3.53	4.97	6.80
FFT	2.59	19.45	103.52	35.58
Puzzle	6.20	25.28	27.09	23.38
Triang	136.77	330.90	256.79	435.00
Fprint	na	2.57	3.77	5.03
Fread	na	6.66	8.64	6.00
Tprint	na	na	4.62	2.68
Frpoly-r-15	2.56	5.23	na	16.84
Frpoly-r2-15	15.04	10.86	109.05	39.90
Frpoly-r3-15	2.53	6.99	17.02	23.80
Geo mean	2.87	1.26	0.83	0.74

Table 1: Common Lisp benchmarks for Symbolics, TI Explorer, and DEC MicroVAX systems

Benchmark	Sun			Apollo		
	3/160	3/60	3/260	3000	580T	4000
Tak	0.44	0.36	0.24	0.59	0.37	0.29
Stak	2.38	1.82	1.12	3.00	1.57	1.32
Ctak	1.70	1.32	0.90	2.99	2.07	1.18
Takl	2.26	1.78	1.26	2.85	1.69	1.32
Takr	0.70	0.56	0.38	0.92	0.55	0.43
Boyer	12.54	9.62	6.48	18.41	13.02	9.89
Browse	19.00	18.14	9.92	32.20	22.77	18.09
Destru	1.58	1.24	0.88	2.56	1.79	1.46
Trav-Init	4.62	3.60	2.58	5.82	4.07	2.97
Trav-Run	37.94	29.72	23.32	45.72	38.43	29.16
Deriv	3.80	2.80	1.98	7.61	5.13	4.81
DDeriv	5.18	3.86	2.62	10.43	7.06	6.47
Div2-Iter	0.98	0.74	0.58	2.68	2.01	1.94
Div2-Recur	1.44	1.16	0.86	3.39	2.82	2.50
FFT	4.62	3.72	3.88	6.29	2.57	2.90
Puzzle	5.06	4.14	2.92	6.60	3.77	3.02
Triang	137.78	107.96	72.92	178.13	95.44	78.46
Fprint	1.24	0.96	0.82	1.94	1.33	1.03
Fread	3.90	3.00	2.12	4.24	3.09	2.38
Tprint	1.94	1.38	0.98	2.20	1.52	1.19
Frpoly-r-15	3.90	2.92	1.94	5.43	4.15	2.84
Frpoly-r2-15	44.90	35.12	24.88	23.30	17.50	13.96
Frpoly-r3-15	7.65	5.86	4.28	11.79	8.52	6.40
Geo mean	2.33	2.98	4.18	1.63	2.44	3.04

Table 2: Common Lisp benchmarks for Apollo and Sun systems

Benchmark	DEC VAX					
	μ VaxII	3600	11/780	11/785	8650	8700
Tak	1.16	0.45	1.23	0.97	0.25	0.21
Stak	4.70	1.00	3.04	2.36	0.63	0.48
Ctak	9.91	2.67	6.64	4.80	1.32	1.34
Takl	7.12	2.02	5.49	4.12	1.14	0.99
Takr	2.51	0.87	2.37	1.42	0.41	0.28
Boyer	44.27	12.32	28.72	18.66	5.63	4.77
Browse	52.80	13.50	32.57	21.96	6.13	6.33
Destru	4.08	1.45	3.81	2.78	0.65	0.73
Trav-Init	14.69	4.89	11.49	7.39	2.33	2.22
Trav-Run	129.00	33.38	95.84	69.63	19.60	17.61
Deriv	12.28	3.75	8.82	6.20	1.73	1.70
DDeriv	21.24	5.42	12.36	8.04	2.51	2.14
Div2-Iter	4.56	1.30	3.26	2.34	0.64	0.55
Div2-Recur	6.80	2.08	5.72	3.77	1.10	0.89
FFT	35.58	9.27	23.08	15.60	4.57	4.31
Puzzle	23.38	8.01	19.43	13.04	3.78	3.65
Triang	435.00	101.49	271.35	191.67	50.86	49.04
Fprint	5.03	1.80	3.97	2.66	0.77	0.77
Fread	6.00	2.28	5.37	2.63	0.89	0.82
Tprint	2.68	0.90	1.69	1.09	0.31	0.33
Frpoly-r-15	16.84	5.68	10.84	6.72	2.46	2.13
Frpoly-r2-15	39.90	14.38	33.73	21.11	6.26	6.10
Frpoly-r3-15	23.80	7.99	14.00	8.56	3.02	2.67
Geo mean	0.74	2.42	1.00	1.49	5.13	5.63

Table 3: Common Lisp benchmarks for various VAX systems

Will Clinger provides the following information on a Scheme version of the above benchmarks:

Gabriel Benchmark timings for MacScheme+Toolsmith Version 1.0

Machines:

- Macintosh II with 5 Mby RAM, 40 Mby internal hard disk; Finder 6.0b2, System 4.1; disk cache turned off
- Macintosh Plus with 1 Mby RAM, 2 Mby QuickDrive (external RAMdisk); Finder 5.5, System 4.1; disk cache turned off

Optimization levels:

- opt=2 Native code.
- opt=1 Interpreted byte code. (The default.)
- opt=0 Unoptimized byte code. (Best for debugging.)

Notes on the implementation:

Values are represented as tagged pointers, with the tag in the high bits. The tag bits are masked off on each access in anticipation of future Macintoshes, though masking is unnecessary with current hardware.

Fixnums are represented in a 30-bit two's complement immediate format. All arithmetic is generic. All flonums are boxed.

The compiler algorithm and abstract machine architecture used at optimization level 0 have been proved correct relative to a denotational semantics for Scheme. Higher levels are based on this correctness proof but add a few optimizations that have not been subjected to formal proof.

Multitasking is supported through a programmable interrupt system and first class continuations. As in PC Scheme, continuation frames are always allocated on a stack, but are copied into the heap on each task switch and on each call to call-with-current-continuation. Unlike PC Scheme, continuations in the heap are invoked without being copied back into the stack. The CTAK benchmark (as modified for Scheme) tests this mechanism.

All benchmarks use generic arithmetic.

All benchmarks perform full tag checking and bounds checks.

CPSTAK Continuation-passing version of TAK. An excellent test of first class procedures and tail recursion, both of which are used heavily in Scheme.

CTAK In Scheme this is a very heavy test of first class continuations (call-with-current-continuation). The results are not comparable with results for other dialects.

TAKR The Macintosh Plus timings show the effect of self-recursion optimization. The Macintosh II timings show the additional effect of the 68020's 256-byte on-chip instruction cache.

FFT Version 1.0 does not use the Macintosh II's floating point coprocessor.

PUZZLE In Scheme, the two-dimensional arrays are represented as vectors of vectors.

FPRINT, FREAD, TPRINT For opt=2, the standard I/O library was compiled at optimization level 2. For opt=1 and opt=0, the standard I/O library was compiled at optimization level 1.

Real (elapsed) times include gc times. The timer's resolution is 1/60 second.

Benchmark	opt=2		opt=1		opt=0	
	GC	Real	GC	Real	GC	Real
Tak	0	2.250	0	8.500	0	12.300
	0	10.433	0	32.767	0	49.700
CPStak	0	7.333	0	15.250	0.117	14.933
	9.016	38.383	3.900	61.350	0	66.383
Ctak	0.317	21.417	0.250	34.767	0.400	33.000
	39.749	123.917	19.801	155.483	21.233	146.583
Takl	0	10.650	0	44.933	0.667	78.383
	0	49.617	0	174.983	40.270	327.150
Takr	0	3.567	0	10.117	0	12.433
	0	15.050	0	37.867	6.284	51.467
Boyer	0.700	52.383	1.584	124.700	7.702	163.617
	-	-	-	-	-	-
Browse	.984	126.967	1.116	269.117	3.649	311.883
	150.989	651.650	80.452	1111.983	242.215	1409.900
Destructive	0	5.417	0	19.200	0.367	29.483
	5.300	29.967	3.767	76.167	17.984	124.250
Traverse-init	0	30.217	0	117.567	7.333	200.100
	-	-	-	-	-	-
Traverse	0	396.650	0	1036.317	59.814	1449.950
	-	-	-	-	-	-
Deriv	0	12.550	0.567	32.400	1.150	36.183
	12.067	64.300	7.566	130.917	16.067	151.567
Dderiv	0.200	16.767	0.583	39.817	1.717	45.650
	16.517	82.483	11.400	162.517	24.649	193.433
Div-iter	0	2.283	0	10.017	1.166	19.400
	6.250	18.950	3.900	43.617	16.433	85.033
Div-rec	0	5.333	0	16.067	0.583	22.650
	6.367	30.917	3.932	67.817	13.919	99.800
FFT	3.951	1012.750	12.303	1114.550	12.282	1125.983
	424.823	3364.967	226.996	3649.350	234.114	3752.750
Puzzle	0	31.917	0	129.150	5.832	199.433
	22.164	169.183	10.949	499.900	151.020	880.667
Triangle	2.634	719.983	2.083	2216.117	108.538	2939.717
	-	-	-	-	-	-
Fprint	0	3.950	0	6.150	0	6.167
	0	12.050	0	18.683	0	18.833
Fread	0	11.600	0	21.750	0	21.767
	7.783	47.700	10.000	86.433	10.000	87.183
Tprint	0	4.500	0	5.617	0	5.617
	0	19.883	0	23.967	0	23.967

Table 4: Scheme benchmarks for MacScheme+Toolsmith V1.0

PROCEEDINGS OF THE
LISP AND FUNCTIONAL PROGRAMMING CONFERENCES

Order From:

ACM Order Department
PO Box 64145
Baltimore, MD 21264

Order No.	Date	Location	ACM Members:	Others:
552800	1980	Stanford, CA	\$15	\$21
552820	1982	Pittsburgh, PA	\$18	\$26
552840	1984	Austin, TX	\$20	\$27
552860	1986	Cambridge, MA	\$21	\$28

The 1980 LISP Conference
Stanford, August 25-27, 1980

Invited Address: 9:00 to 9:30, August 25, 1980	
John McCarthy, Stanford University	v
 Session 1: 9:30 to 10:20, August 25, 1980	Chaired by Richard Fateman, U. C. Berkeley
<i>Symbolic Computing with and without LISP</i>	1
J. Campbell, University of Exeter and J. Fitch, University of Bath	
<i>Prose and CONS - Multics Emacs: A Commercial Text-processing System in LISP</i>	6
B. Greenberg, Honeywell	
 Session 2: 10:45 to 12:00, August 25, 1980	Chaired by Daniel Friedman, Indiana University
<i>Explicit Parallelism in LISP-like Languages</i>	13
G. Prini, University of Pisa	
<i>Continuation-Based Multiprocessing</i>	19
M. Wand, Indiana University	
<i>MULTI - A LISP Based Multiprocessing System</i>	29
D. McKay and S. Shapiro, SUNY at Buffalo	
 Session 3: 1:30 to 3:10, August 25, 1980	Chaired by Carolyn Takcott, Stanford University
<i>The Function-Class</i>	38
T. Kurokawa, Japan	
<i>A Constructive Alternative to Axiomatic Data Type Definitions</i>	46
R. Cartwright, Cornell University	
<i>A Semantic Comparison of LISP and SCHEME</i>	56
S. Muchnick, UC Berkeley and U. Pleban, University of Kansas	
<i>MODLISP</i>	65
J. Davenport and R. Jenks, IBM	
 Session 4: 3:45 to 5:00, August 25, 1980	Chaired by Alan Kay, Xerox PARC
<i>Extending Object Oriented Programming in Smalltalk</i>	75
I. Goldstein and D. Bobrow, Xerox PARC	
<i>A System of Communicating Residential Environments</i>	82
E. Sandewall, H. Sorenson, and C. Stromberg, Linkoping	
<i>A Session with Tinker: Interleaving Program Testing with Program Writing</i>	90
H. Lieberman and C. Hewitt, MIT	
 Session 5: 9:00 to 10:15, August 26, 1980	Chaired by David Wise, Indiana University
<i>Computing with Text-Graphics Forms</i>	100
F. Lakin, Xerox	
<i>Design of the APIARY for Actor Systems</i>	107
C. Hewitt, MIT	
<i>Address/Memory Management for a Gigantic LISP Environment</i>	119
J. White, MIT	

Session 6: 10:45 to 12:00, August 26, 1980		Chaired by Carl Hewitt, MIT
<i>SKIM - The S, K, I Reduction Machine</i>		128
T. Clarke, P. Gladstone, C. MacLean and A. Norman, Trinity College		
<i>HOPE: An Experimental Applicative Language</i>		136
R. Burstall, D. MacQueen, and D. Sannella, University of Edinburgh		
<i>Computing Cyclic List Structures</i>		144
L. Morris, Syracuse University and J. Schwarz, Bell Labs		
Session 7: 1:30 to 3:10, August 26, 1980		Chaired by Anthony Hearn, University of Utah
<i>An Efficient Environment Allocation Scheme in an Interpreter for a Lexically-Scoped LISP</i>		154
D. McDermott, Yale University		
<i>The Dream of a Lifetime: A Lazy Variable Extent Mechanism</i>		163
G. Steele Jr. and G. Sussman, MIT		
<i>Strategies for Data Abstraction in LISP</i>		173
B. Steele, MIT		
<i>Special Forms in LISP</i>		179
K. Pitman, MIT		
Session 8: 3:45 to 5:00, August 26, 1980		Chaired by Bruce Anderson, Essex University
<i>Panel Discussion</i>		
Session 9: 8:35 to 10:15, August 27, 1980		Chaired by Gianfranco Prini, University of Pisa
<i>Multiprocessing via Intercommunicating LISP Systems</i>		188
M. Model, Brandeis University		
<i>Divide and CONCer: Data Structuring in Applicative Multiprocessing Systems</i>		196
R. Keller, University of Utah		
<i>Compilation Techniques for a Control-Flow Concurrent LISP System</i>		203
J. Marti, University of Utah		
<i>On Compiling Embedded Languages in LISP</i>		208
P. Emanuelson and A. Haraldsson, Linkoping		
Session 10: 10:45 to 12:25, August 27, 1980		Chaired by Eichi Goto, University of Tokyo
<i>A LISP Compiler Producing Compact Code</i>		216
W. Rowan, UC San Francisco		
<i>Local Optimization in a Compiler for Stack-based LISP Machines</i>		223
L. Masinter and L. P. Deutsch, Xerox PARC		
<i>ByteLisp and its Alto Implementation</i>		231
L. P. Deutsch, Xerox PARC		
<i>Overview and Status of DoradoLISP</i>		243
R. Burton, L. Masinter, D. Bobrow, W. Haugeland, R. Kaplan, and B. Sheil, Xerox PARC		

The 1982 ACM Symposium on LISP and Functional Programming

Pittsburgh, PA, August 15 - 18, 1982

Monday, August 16, 1982

WELCOME 9:00 am

Symposium Chairman: David M.R. Park (University of Warwick)

Local Arrangements Chairman: Guy L. Steele, Jr. (Carnegie-Mellon University)

Session 1: 9:30 am - 12:30 pm Chairman: Nico Habermann (Carnegie-Mellon University)

*** Programming with Infinite Data Structures**

David Turner (University of Canterbury)

Super Combinators: A New Implementation Method for Applicative Languages

R.J.M. Hughes (Oxford University) 1

A Fixed-Program Machine for Combinator Expression Evaluation

Steven S. Muchnick (University of California, Berkeley)

Neil D. Jones (Aarhus University) 11

Expressions as Processes

J.R. Kennaway and M.R. Sleep (University of East Anglia) 21

Session 2: 2:30 pm - 6:00 pm Chairman: Joseph E. Stoy (Oxford University)

Combinatory Foundation of Functional Programming

Corrado Böhm (University of Rome) 29

Functional Specifications of a Text Editor

Gary Feldman (Carnegie-Mellon University) 37

Experiments with a Supercompiler

Valentin F. Turchin, Robert M. Nirenberg & Dimitri V. Turchin

(The City University of New York) 47

A Parallel Prolog: the construction of a data driven model	
<i>Michael J. Wise (University of New South Wales)</i>	56
How to Define a Language using PROLOG	
<i>Christopher D.S. Moss (Imperial College)</i>	67
Logic Enhancement: A Method for Extending Logic Programming Languages	
<i>Paul R. Eggert (University of California, Santa Barbara & System Dev. Corp.)</i> <i>D. Val Schorre (System Development Corporation)</i>	74
 Session 3: 8:00 pm - 10:00 pm Chairman: John Allen (The LISP Company)	
Implementation of Interlisp on the VAX	
<i>Raymond L. Bates, David Dyer and Johannes A.G.M. Koomen</i> <i>(University of Southern California, Information Sciences Institute)</i>	81
PSL: A Portable LISP System	
<i>Martin L. Griss, Eric Benson and Gerald Q. Maguire, Jr. (University of Utah)</i>	88
An Overview of Common LISP	
<i>Guy L. Steele, Jr. (Carnegie-Mellon University)</i>	98
S-1 Common Lisp Implementation	
<i>Rodney A. Brooks (M.I.T.)</i> <i>Richard P. Gabriel (Stanford University & Lawrence Livermore National Laboratory)</i> <i>Guy L. Steele, Jr. (Carnegie-Mellon University)</i>	108
T: A Dialect of Lisp or, LAMBDA: The Ultimate Software Tool	
<i>Jonathan A. Rees and Norman I. Adams IV (Yale University)</i>	114

**The 1986 ACM Conference on
Lisp and Functional Programming
Massachusetts Institute of Technology, August 4-6, 1986**

Monday, August 4

Session 1: 9:00 a.m. - 10:30 a.m.
Chaired by *William L. Scherlis (CMU)*

Laws in Miranda <i>Simon Thompson (University of Kent at Canterbury)</i>	1
A Simple Applicative Language: Mini-ML <i>Dominique Clement (SEMA, Sophia-Antipolis), Joelle Despeyrouz, Thierry Despeyrouz, Gilles Kahn (INRIA, Sophia-Antipolis)</i>	13
Integrating Functional and Imperative Programming <i>David K. Gifford, John M. Lucassen (Massachusetts Institute of Technology)</i>	28

Session 2: 10:55 a.m. - 12:35 p.m.
Chaired by *Daniel Weinreb (Symbolics, Inc.)*

Experience with an Uncommon LISP <i>Cyril N. Alberga, Chris Bosman-Clark, Martin Mikelsons, Mary S. Van Deusen (IBM T. J. Watson Research Center), Julian Padget (University of Bath)</i>	39
Desiderata for the Standardization of LISP <i>Julian Padget (University of Bath), et al.</i>	54
Design of an Optimizing, Dynamically Retargetable Compiler for Common Lisp <i>Rodney A. Brooks (Massachusetts Institute of Technology, Lucid, Inc.), David B. Posner, James L. McDonald, Jon L. White, Eric Benson, Richard P. Gabriel (Lucid, Inc.)</i>	67
The Implementation of PC Scheme <i>David H. Bartley, John C. Jensen (Texas Instruments Incorporated)</i>	86

Session 3: 2:00 p.m. - 3:40 p.m.
Chaired by *John H. Williams (IBM Research)*

Code Generation Techniques for Functional Languages <i>Jon Fairbairn, Stuart C. Wray (University of Cambridge)</i>	94
An Architecture for Mostly Functional Languages <i>Tom Knight (Symbolics, Inc. and Massachusetts Institute of Technology)</i>	105
Mechanisms for Efficient Multiprocessor Combinator Reduction <i>M. Lemaître, M. Castan, M. -H. Durand, G. Durrieu, B. Lecussan, (ONERA-CERT)</i>	113
The CURRY Chip <i>John D. Ramsdell (The MITRE Corporation)</i>	122

Monday, August 4

Session 4: 4:05 p.m. – 5:45 p.m.

Chaired by *Mitchell Wand (Northeastern University)*

Variations on Strictness Analysis <i>Adrienne Bloss, Paul Hudak (Yale University)</i>	132
Expansion-Passing Style: Beyond Conventional Macros <i>R. Kent Dybvig, Daniel P. Friedman, Christopher T. Haynes (Indiana University)</i>	143
Hygienic Macro Expansion <i>Eugene Kohlbecker, Daniel P. Friedman, Matthias Felleisen, Bruce Duba (Indiana University)</i>	151
Exact Real Arithmetic: A Case Study in Higher Order Programming <i>Hans-J. Boehm, Robert Cartwright, Mark Riggle (Rice University), Michael J. O'Donnell (University of Chicago)</i>	162

Tuesday, August 5

Session 5: 9:00 a.m. – 10:40 a.m.

Chaired by *Rodney Brooks (MIT)*

Reconfigurable, Retargetable Bignums: A Case Study in Efficient, Portable Lisp System Building <i>Jon L. White (Lucid, Inc.)</i>	174
LISP on a Reduced-Instruction-Set-Processor <i>Peter Steenkiste, John Hennessy (Stanford University)</i>	192
Partitioning Parallel Programs for Macro-Dataflow <i>Vivek Sarkar, John Hennessy (Stanford University)</i>	202
NORMA: A Graph Reduction Processor <i>Mark Scheevel (Burroughs Corporation)</i>	212

Session 6: 11:05 a.m. – 12:20 p.m.

Chaired by *Mark Wegman (IBM Research)*

The Four-Stroke Reduction Engine <i>Chris Clack, Simon L. Peyton Jones (University College London)</i>	220
On the Use of LISP in Implementing Denotational Semantics <i>Peter Lee, Uwe Pleban (The University of Michigan)</i>	233
Semantics Directed Compiling for Functional Languages <i>Hanne R. Nielson, Flemming Nielson (Aalborg University Centre)</i>	249

Tuesday, August 5

Session 7: 2:00 p.m. – 3:15 p.m.

Chaired by *Gilles Kuhn (INRIA)*

Connection Graphs <i>Alan Bowden (Massachusetts Institute of Technology)</i>	258
Implementing Functional Languages in the Categorical Abstract Machine <i>Michel Mauny, Ascander Suarez (INRIA)</i>	266
Connection Machine LISP: Fine-Grained Parallel Symbolic Processing <i>Guy L. Steele, Jr., W. Daniel Hillis (Thinking Machines Corporation)</i>	279

Session 8: 3:45 p.m. – 5:45 p.m.

Chaired by *L. Peter Deutsch (Xerox PARC and CodeSmith Technology, Inc.)*

Panel: Object Oriented Programming in Lisp

Wednesday, August 6

Session 9: 9:00 a.m. – 10:40 a.m.

Chaired by *David MacQueen (Bell Laboratories)*

The Mystery of the Tower Revealed: A Non-Reflective Description of the Reflective Tower <i>Mitchell Wand (Northeastern University), Daniel P. Friedman (Indiana University)</i>	298
A Type-Inference Approach to Reduction Properties and Semantics of Polymorphic Expressions (Summary) <i>John C. Mitchell (AT&T Bell Laboratories)</i>	308
Equations, Sets, and Reduction Semantics for Functional and Logic Programming <i>Bharat Jayaraman, Frank S. K. Silbermann (University of North Carolina at Chapel Hill)</i>	320
Towards a Semantic Theory for Equational Programming Languages <i>Satish R. Thatte (The University of Michigan)</i>	332

Session 10: 11:05 a.m. – 12:20 p.m.

Chaired by *Richard P. Gabriel (Lucid, Inc.)*

A Protocol for Distributed Reference Counting <i>Claus-Werner Lermen, Dieter Maurer (Universitat des Saarlandes)</i>	343
A Semantic Model of Reference Counting and its Abstraction (Detailed Summary) <i>Paul Hudak (Yale University)</i>	351
Distributed Copying Garbage Collection <i>Martin Rudalics (Johannes Kepler Universität)</i>	364

The 1984 ACM Symposium on LISP and Functional Programming
 Austin, Texas
 August 5-8, 1984

Monday Afternoon, August 6, 1984

Session 3 (2:10-3:50 P.M.) Session chairman: John Williams (International Business Machines)
 Laziness is Better than Laziness: Lazy Evaluation and Garbage Collection at Compile Time 45
Philip Wadler (Oxford University)

Stream Processing 53
Allen Goldberg (University of California, Santa Cruz, and Kestrel Institute)
and Robert Paige (Rutgers University)

Rewriting Systems on FP Expressions that Reduce the Number of Sequences They Yield 63
Francoise Bellegarde (Centre de recherche en informatique de Nancy)

Schema Recognition for Program Transformations 74
John S. Givler and Richard B. Kieburtz (Oregon Graduate Center)

Coffee break (3:50-4:15 P.M.)

Session 4 (4:15-5:30 P.M.) Session chairman: Guy L. Steele Jr. (Tartan Laboratories)
 Forms: an Object and Time Oriented System for Music Composition and Synthesis 85
Pierre Colite and Xavier Roder (Institut de Recherche et Coordination en Acoustique/Musique)

Arctic: A Functional Language for Real-Time Control 96
Roger B. Deunenberg (Carnegie-Mellon University)

muFP: A Language for VLSI Design 104
Mary Sheeran (Oxford University)

Break for dinner (5:30-8:00 P.M.)

Monday Morning, August 6, 1984

Session 1 (9:00-10:40 A.M.) Session chairman: Daniel P. Friedman (Indiana University)
 Opening remarks
General chairman: Robert S. Boyer (University of Texas at Austin)
Local arrangements chairman: Edward S. Schneider (Burroughs Corporation)

Welcome

K. Mani Chandy (Chairman, Computer Sciences, University of Texas at Austin)

Invited talk: How to Teach LISP
Patrick H. Winston (Massachusetts Institute of Technology)

A Critique of Common Lisp 1
Rodney A. Brooks and Richard P. Gabriel (Stanford University)

Coffee break (10:40-11:05 A.M.)

Session 2 (11:05 A.M.-12:20 P.M.) Session chairman: Jerome Chailoux (INRIA)
 Implementation of Multilisp: Lsp on a Multiprocessor 9
Robert H. Halstead, Jr. (Massachusetts Institute of Technology)

Engines Build Process Abstractions 18
Christopher T. Haynes and Daniel P. Friedman (Indiana University)

Queue-based Multi-processing Lisp 25
Richard P. Gabriel and John McCarthy (Stanford University)

Lunch (12:30-2:10 P.M.)

Monday Evening, August 6, 1984

Session 5 (8:00-10:00 P.M.)	Session chairman: Richard P. Gabriel (Stanford University)	
I.F.LISP: a Portable and Efficient LISP System	113	
<i>Jérôme Chailoux (INRIA), Mathieu Devin (Ecole des Mines de Paris), and Jean-Marie Hullot (INRIA)</i>		
DIALISP - A LISP Machine	123	
<i>G. Siegfou, A. Puvion, V. Ristricceanu, and A. Rimbaun (Polytechnical Institute of Bucharest)</i>		
Recent Developments in ISI-Interlisp	129	
<i>Raymond L. Bates, David Dyer, and Mark Feber (University of Southern California Information Sciences Institute)</i>		
TACO: A Fast Interpreter-Centered Lisp System on Lisp Machine ELJS	140	
<i>Hiroshi G. Okuno, Ikuo Takeuchi, Nobuyasu Osaio, Yasushi Hibino, and Kazuyumi Watanabe (Nippon Telegraph and Telephone Public Corporation)</i>		
The Design of an Instruction Set for Common Lisp	150	
<i>Skof Wholey and Scott E. Fahlman (Carnegie-Mellon University)</i>		

Tuesday Morning, August 7, 1984

Session 6 (9:00-10:40 A.M.)	Session chairman: Peter Henderson (University of Stirling)	
Some Practical Methods for Rapid Combinator Reduction	159	
<i>W. R. Smye, T. J. W. Clarke, and A. C. Norman (University of Cambridge)</i>		
Experiments in Diffused Combinator Reduction	167	
<i>Paul Hudak and Benjamin Goldberg (Yale University)</i>		
Invited talk: Expression Evaluation in the ICON Programming Language	177	
<i>Ralph E. Griswold (University of Arizona)</i>		
Coffee break (10:40-11:05 A.M.)		
Session 7 (11:05 A.M.-12:20 P.M.)	Session chairman: William L. Scherlis (Carnegie-Mellon University)	
Invited paper: A Proposal for Standard ML	184	
<i>Robin Milner (University of Edinburgh)</i>		
Modules for Standard ML	198	
<i>David MacQueen (AT&T Bell Laboratories)</i>		
Compiling a Functional Language	208	
<i>Luca Cardelli (AT&T Bell Laboratories)</i>		
A Compiler for Lazy ML	218	
<i>Lennart Augustsson (Chalmers University of Technology)</i>		

Lunch (12:20-2:10 P.M.)

Tuesday Afternoon, August 7, 1984

Session 8 (2:10-3:50 P.M.) Session chairman: **Martin L. Griss (Hewlett-Packard)**

Recursion is More Efficient than Iteration 228
Fimmanuel Saint-James (Université Pierre et Marie Curie)

Garbage Collection in a Large Lisp System 235
David A. Moon (Symbolics, Inc.)

Steps Toward Better Debugging Tools for Lisp 247
Henry Lieberman (Massachusetts Institute of Technology)

Trading Data Space for Reduced Time and Code Space
in Real-Time Garbage Collection on Stock Hardware 256
Rodney A. Brooks (Stanford University)

Coffee break (3:50-4:15 P.M.)

Session 9 (4:15-5:55 P.M.) Session chairman: **Dana Scott (Carnegie-Mellon University)**

Type Inference and Type Checking for Functional Programming Languages: 263
A Reduced Computation Approach
Takuya Katayama (Tokyo Institute of Technology)

A Powerful Strategy for Deriving Efficient Programs by Transformation 273
Alberto Pettorossi (Edinburgh University)

Busy and Lazy FP with Infinite Objects 282
Walter Dösch and Bernhard Möller (Technische Universität München)

Continuations and Coroutines 293
Christopher T. Haynes, Daniel P. Friedman, and Mitchell Wand (Indiana University)

Break before banquet (5:55-7:30 P.M.)

Banquet (7:30 P.M.)

Invited speaker: Raymond Smullyan

Wednesday Morning, August 8, 1984

Session 10 (9:00-10:40 A.M.) Session chairman: **David Warren (Silogic Incorporated)**

Invited talk: Early LISP History (1956-1959) 299
Herbert Stoyan (University of Erlangen-Nürnberg)

Making Control and Data Flow in Logic Programs Explicit 311
Gert Smulka (Cornell University)

TABLOG: The Deductive-Tableau Programming Language 323
*Yonithan Malachi (Stanford University), Zohar Manna (Stanford University),
and Richard Waldinger (SRI International)*

Coffee break (10:40-11:05 P.M.)

Session 11 (11:05 A.M.-12:20 P.M.) Session chairman: **Robert Cartwright (Rice University)**

The Implementation of Procedurally Reflective Languages 331
Jim des Rivères and Brian Cartmill Smith (Xerox Palo Alto Research Center)

Refutation: Reflection without Metaphysics 348
Daniel P. Friedman and Mitchell Wand (Indiana University)

The Scheme 311 Compiler: An Exercise in Denotational Semantics 356
William Clinger (Indiana University)

End of conference (12:20 P.M.)

EDITORIAL POLICY

All submissions to Lisp Pointers, with the exception of technical articles, should be made in camera-ready text and sent to the appropriate department head. Technical articles may be submitted to the Technical Articles Editor in either hard copy or in TEX source files by Arpanet link, tar format cartridge tape, or tar format reel-to-reel. All submissions should be single-spaced with no page numbers. Without a special waiver from the appropriate department head, submissions will be limited to ten pages. This can be achieved by printing longer articles two-up. Camera-ready text is defined to be no more than 7 1/2 X 10 inches or 19 X 25 centimeters, centered on an 8 1/2 X 11 inch page. Articles that contain too much blank space will be rejected. It is the author's responsibility to retain a working copy of the submission, as contributions will not be returned to authors. Authors not fluent in writing English are requested to have their work reviewed and corrected for style and syntax prior to submission.

Although Lisp Pointers is not refereed, acceptance is subject to the discretion of the appropriate department head. The scope of topics for Lisp Pointers includes all dialects of Lisp and Scheme. We encourage research articles, tutorials, and summarizations of discussions in other forums. Lisp Pointers is not a forum for detailed discussions on proposed changes to the Common Lisp standard.

Lisp Pointers is a non-profit publications sponsored by companies interested in the Lisp topic. No company has directed or controlled its publication. The opinions expressed herein are solely those of the authors, editors, and other contributors. These opinions do not necessarily reflect the opinions of either the companies affiliated with these individuals or the companies sponsoring this newsletter. There is no cost for a subscription to Lisp Pointers for either an individual or a library. Lisp Pointers is published four times per year (July-August-September, October-November-December, January-February-March, April-May-June). Submission deadlines are July 1, October 1, January 1, and April 1.

Free Subscription to Lisp Pointers (postal mail only)

LISP POINTERS
Mary S. Van Deusen, Editor
IBM Watson Research
PO Box 704
Yorktown Heights, NY 10598

Name: _____

Mailing Address: _____

If the mailing list for Lisp Pointers is ever made available for commercial use, please do not make my name available.

CURRENT AND PREVIOUS SPONSORS OF LISP POINTERS

IBM Thomas J. Watson Research
Institut National De Recherche En Informatique Et En Automatique (I.N.R.I.A.) **
Xerox PARC
Microelectronics and Computer Technology Corporation (MCC)
Texas Instruments
Digital Equipment Corporation

**** Continuing Sponsor**