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.

	Symbolics	TI Explorer	DEC MicroVAX-II	
Benchmark	**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

	Sun			Apollo		
Benchmark	**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

	DEC VAX					
Benchmark	**μVaxII**	**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.

	opt=2		opt=1		opt=0	
Benchmark	**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

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)

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

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)*

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

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

Monday, August 4
Session 4: 4:05 p.m. – 5:45 p.m.
Chaired by *Mitchell Wand* (*Northeastern University*)

Tuesday, August 5
Session 5: 9:00 a.m. – 10:40 a.m.
Chaired by *Rodney Brooks* (*MIT*)

Session 6: 11:05 a.m. – 12:20 p.m.
Chaired by *Mark Wegman* (*IBM Research*)

Tuesday, August 5
Session 7: 2:00 p.m. – 3:15 p.m.
Chaired by *Gilles Kahn (INRIA)*

Session 8: 3:45 p.m. – 5:45 p.m.
Chaired by *L. Peter Deutsch (Xerox PARC and CodeSmith Technology, Inc.)*

Wednesday, August 6
Session 9: 9:00 a.m. – 10:40 a.m.
Chaired by *David MacQueen (Bell Laboratories)*

Session 10: 11:05 a.m. – 12:20 p.m.
Chaired by *Richard P. Gabriel (Lucid, Inc.)*

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

Monday Morning, August 6, 1984

Monday Afternoon, August 6, 1984

Monday Evening, August 6, 1984

Session 5 (8:00–10:00 P.M.) **Session chairman: Richard P. Gabriel (Stanford University)**

Tuesday Morning, August 7, 1984

Session 6 (9:00–10:40 A.M.) **Session chairman: Peter Henderson (University of Stirling)**

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)**

Lunch (12:20–2:10 P.M.)

Tuesday Afternoon, August 7, 1984

Wednesday Morning, August 8, 1984

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