Cache Performance of the Integer SPEC Benchmarks on a RISC [†]

Dionisios N. Pnevmatikatos
Mark D. Hill

Computer Sciences Department
University of Wisconsin-Madison
1210 W. Dayton Street
Madison, WI 53706

ABSTRACT

SPEC is a new set of benchmark programs designed to measure a computer system's performance. The performance measured by benchmarks is strongly affected by the existence and configuration of cache memory. In this paper we evaluate the cache miss ratio of the Integer SPEC benchmarks. We show that the cache miss ratio depends strongly on the program, and that large caches are not completely exercised by these benchmarks.

1. Introduction

The SPEC benchmarks are a new set of benchmark programs designed to evaluate the performance of computer systems [9]. In the design of this set, considerable effort is made so that the results are indicative from the expected performance in environments like Computer Aided Software Engineering and Computer Aided Engineering. To achieve this goal, public domain real applications from the above areas are used. The performance measured is scaled to the performance of the VAX 11/780 for each program in the set, and the ratios are averaged using the geometric mean as proposed in [1]. The first release of the SPEC benchmarks consists of ten programs: GNU C compiler, Eqntott, Xlisp, Espresso, Spice 2g6, Doduc, Nasa7, Matrix300, Fpppp and TomcatV. The first four are integer-intensive programs written in C, and the rest are floating-point-intensive programs written in Fortran.

Since most high-performance computer systems include one or more caches to reduce the average memory access time and therefore to improve performance [7], a question that can be asked is what is the effect of caches on the performance measured by the benchmark programs. Knowing the cache behavior of the benchmarking programs, one can predict the performance change for a given change in the memory hierarchy. However, the performance of a system will not be accurately predictable if the cache behavior shows sharp changes around some parameter values, because small changes in a cache configuration can yield large changes in the measured performance. Also important are cache configuration sizes where miss ratios become negligible, since improving cache parameters beyond these sizes will not improve the performance measured by these benchmarks.

To evaluate the cache performance of the SPEC benchmarks, the four integer-intensive programs were traced and simulated for a large number of cache configurations. The programs were compiled on a DECstation 3100 based on the MIPS R2000 architecture running the Ultrix operating system.

2. Tracing and simulation

The traces were obtained using the Abstract Execution method [6], for speed and compression reasons. AE poses some restrictions on the traces that can be obtained, the most important of which are that they can only be single process traces, that the operating system calls cannot be traced, and that the GNU C compiler must be used. Even though AE can trace through library calls, the standard C libraries were not traced in this simulation. However, we feel that the programs do not use them very much and the difference in the results would be small. All programs were compiled with the optimizing flag enabled. For the cache simulation, the *All Associativity* simulation algorithm [5] as implemented in Tycho [4] was used.

The cache parameters used throughout the simulation were: cache size from 1 Kbyte up to 1 Mbyte, associativity from 1 (direct mapped) up to 8 way set-associative, and block size from 16 up to 256 bytes. All steps through these parameters are in factors of two. The replacement algorithm used was LRU and the set-mapping function was bit selection. All the above cache configurations were simulated for instruction, data and unified caches. The reported miss ratio is the total number of misses for the entire execution of the program, divided by the total number of references. Since the entire execution of the program is traced, the cold misses are included in the miss ratio. For Gcc and Espresso, that are run for several input files, the misses for all input files are summed and divided by the total number of references made. Due to time limitations, Espresso was simulated for only two of the eight input files that constitute the benchmark. The complete results are given in tabular form in the Appendix. In the results, a zero miss ratio means that the actual miss ratio is less than 0.0001. The total number of references for each program and the breakdown to instruction and data accesses is given in Table I.

Table I

Program	Instruction	Data	Total
Eqntott	1,129,738,935	288,685,224	1,418,424,159
Xlisp	1,280,576,271	592,700,976	1,873,277,247
GNU C Compiler	1,030,734,885	404,430,283	1,435,165,168
Espresso	1,415,583,519	394,076,087	1,809,659,606

3. Simulation results

This section present the four integer SPEC benchmarks in some detail and the corresponding simulation results. Additional insight can be obtained from Figures 1, 2 and 3 that give some of the results in a graphical form. Figure 1 plots the miss ratio as a function of cache size, Figure 2 plots the change in the miss ratio when the block size is doubled and Figure 3 plots the change in miss ratio when degree of associativity is doubled. For the rest of this section, the misses are divided into three categories [2]: *conflict* misses due to too many active blocks mapping to a fraction of the sets, *capacity* misses due to the fixed cache size and *compulsory* misses, the ones that first reference an item.

3.1. Eqntott

Eqntott is a program that translates a logical representation of a boolean equation to a truth table. The primary computation it performs is sorting, using the quicksort algorithm. Profiling this program shows that 95% of the time is spent in the quicksort routine. The size of data for the benchmark input is approximately 1.8 Mbytes.

The performance of the instruction cache for Eqntott can be seen clearly in the tables: for instruction caches of size larger than 2Kbytes the miss ratio is smaller than 0.0001. Since 95% of the time is spent in the quicksort routine, any cache of size larger than that will give the same performance.

The behavior of the data cache for Eqntott is a result of the nature of quicksort, which does not show a particularly high temporal locality, and of the large data size. Because of the low temporal locality of the algorithm, the capacity misses dominate and give a relatively high miss ratio for caches of size up to 256 Kbytes (0.03 for 128 Kbyte eight-way set-associative cache with 16 bytes block size). In the same range, the miss ratio is almost insensitive to changes in associativity as can be seen in Figure 3. For larger caches, associativity improves significantly the miss ratio. This behavior differs considerably from the one reported in previous studies [3], [5]. Furthermore, the miss ratio for medium and large size caches drops very quickly as the block size is increased, as the compulsory misses are reduced.

The unified cache performance of Eqntott, because of its degenerate instruction behavior, can be easily calculated by multiplying the data miss ratio by the fraction of data references of the program. This simple relation holds only when the number of sets is enough so that the conflicts between instruction and data accesses are rare.

3.2. Xlisp

Xlisp is a small implementation of a lisp interpreter, which also provides some object-oriented programming hooks. The input file is a small backtracking program that solves the Eight Queens problem. The small size of the program can be seen in the table for associativity 8: a cache size larger than 16 Kbytes for instruction and 32 Kbytes for data cache give miss ratio less than 0.0001.

For Xlisp, a small instruction cache gives a high miss ratio, because the interpreter shows low temporal locality (0.10 for 2Kbyte direct mapped cache with 16 bytes block size). However the miss ratio drops fast with the cache size and with the associativity, since the active part of the code is small.

The data accesses of Xlisp show greater temporal locality, so the miss ratio for small data caches is smaller than for instruction. It shows also significant spatial locality as can be seen in Figure 2. Increasing the block size improves the miss ratio considerably. Changes in the associativity are within previously reported limits, except for caches that are large enough to hold the entire working set, in which case the improvement can be larger than a factor of two (change from direct mapped to two-way set-associative cache of size larger than 32Kbytes).

The unified cache behavior follows the lines of the instruction and data cache behavior. The magnitude of the miss ratio is larger because of interference between instruction and data accesses. That interference can be reduced by increasing the associativity.

3.3. GNU C Compiler

The GNU C compiler or Gcc is the version 1.35 of the compiler as distributed by the Free Software Foundation. The benchmark consists of converting 19 preprocessed source files into optimized 68020 assembly language using the Sun-3 assembly format. This program is expected to predict accurately the compiling performance of the system in a software engineering environment.

Because Gcc is a big program, the miss ratios it shows are among the highest ones. The behavior of Gcc as can be seen in Figures 1, 2 and 3 is uniform and consistent with earlier studies [3], [5], [8] over a wide range of parameters. In that sense, Gcc is a well-behaved program and its behavior can be easily and accurately predicted or interpolated from incomplete data without large errors. Because of its size and structure, its demands from the memory hierarchy are significant.

3.4. Espresso

Finally, Espresso is a tool for generation and optimization of Programmable Logic Arrays from the Berkeley CAD software distribution. The main computations it performs are set operations like union and intersection, the sets being represented as arrays of bits. The set operations are then implemented as logical operations on these arrays. The benchmark consists of converting a set of eight input models into the corresponding optimized PLA output. For the two of the inputs that were traced, the breakdown in instruction and data references is shown in Table II.

Table II

Espresso			
Input File	Instr	Data	Total
bca	446,395,312	117,213,963	563,609,275
tial	969,188,207	276,862,124	1,246,050,331

Since Espresso is a relatively small program that spends a lot of its time in inner loops operating on arrays, the instruction cache miss ratio is small. The miss ratio is dominated by conflict misses, as can be seen by the effect of increasing the associativity. For Espresso, any two-way set-associative instruction cache of size larger than 32Kbytes has miss ratio less than 0.0001.

The array manipulation also limits the temporal locality of the data accesses and results in high miss ratio for data caches. However, the active data size is about 64 Kbytes, so caches of this or larger size have essentially the same performance.

Finally, the unified cache performance lies between the performance of split instruction and data caches. However, we should note that even though the results for Espresso appear to be consistent and reasonable, one should be careful when generalizing since only two of the eight inputs were simulated.

4. Conclusions

The cache performance of Integer SPEC benchmarks varies greatly depending on the program. With the major exception of Eqntott, the trends for small caches are similar. However, large caches show almost zero miss ratio for Xlisp and Espresso (0.002 for 64 Kbyte two-way set associative cache with 16 bytes block size). The behavior of Eqntott, because of its special structure, deviates greatly from the other programs. It favors configurations with a small instruction cache and a large data or unified cache. Overall, a split instruction/data cache of size 128 Kbytes each will give very low miss ratios for all programs and further improvements will not be clearly reflected in the SPEC results.

It might be arguable that, since the traces are not multiprogrammed as in most previous studies, the cache performance can be different. However, SPEC measure the performance in a practically single process environment, while the actual use of the machines is multiprogrammed. The Integer SPEC benchmarks, although a significant improvement over past standard benchmarks, will fail to completely exercise high performance memory hierarchies designed for such systems.

Acknowledgments

References

[1]   Philip Flemming and John Wallace, "How not to lie with statistics: The correct way to summarize benchmark results," *Communications of ACM*, Vol 29, 3, March 1986, 218-221.

[2]   Mark D. Hill, "Aspects of Cache Memory and Instruction Buffer Performance," Ph. D. Thesis, Univ. of California at Berkeley, Technical Report UCB/CSD 87/381, November 1987.

[3]   Mark D. Hill, "A Case For Direct-Mapped Caches," *IEEE Computer*, Vol 21, pp. 25-40, December 1989.

[4]   Mark D. Hill, "Tycho," Unpublished UNIX style manual page.

[5]   Mark D. Hill and Alan J. Smith, "Evaluating Associativity in CPU Caches," in *IEEE Transactions On Computer*, Vol 38, 12, December 1989.

[6]   James Larus, "Abstract Execution: A Technique for Efficiently Tracing Programs," Univ. Of Wisconsin, Madison, Technical Report #912, February 1990.

[7]   Alan J. Smith, "Cache Memories," *Computing Surveys*, Vol 14, 3, September 1982.

[8]   Alan J. Smith, "Line (Block) Size Choice for CPU Cache Memories," *IEEE Transactions on Computers*, Vol C-36, 9, September 1987, 1063-1075

[9]   SPEC newsletter, Vol 1, 1, Fall 1989.

Figure 1. Miss ratio as a function of cache size for two-way set associative caches and for block size 16 and 32 bytes. Note that the y-axis scale vary.

GNU C Compiler

Eqntott

Xlisp

Espresso
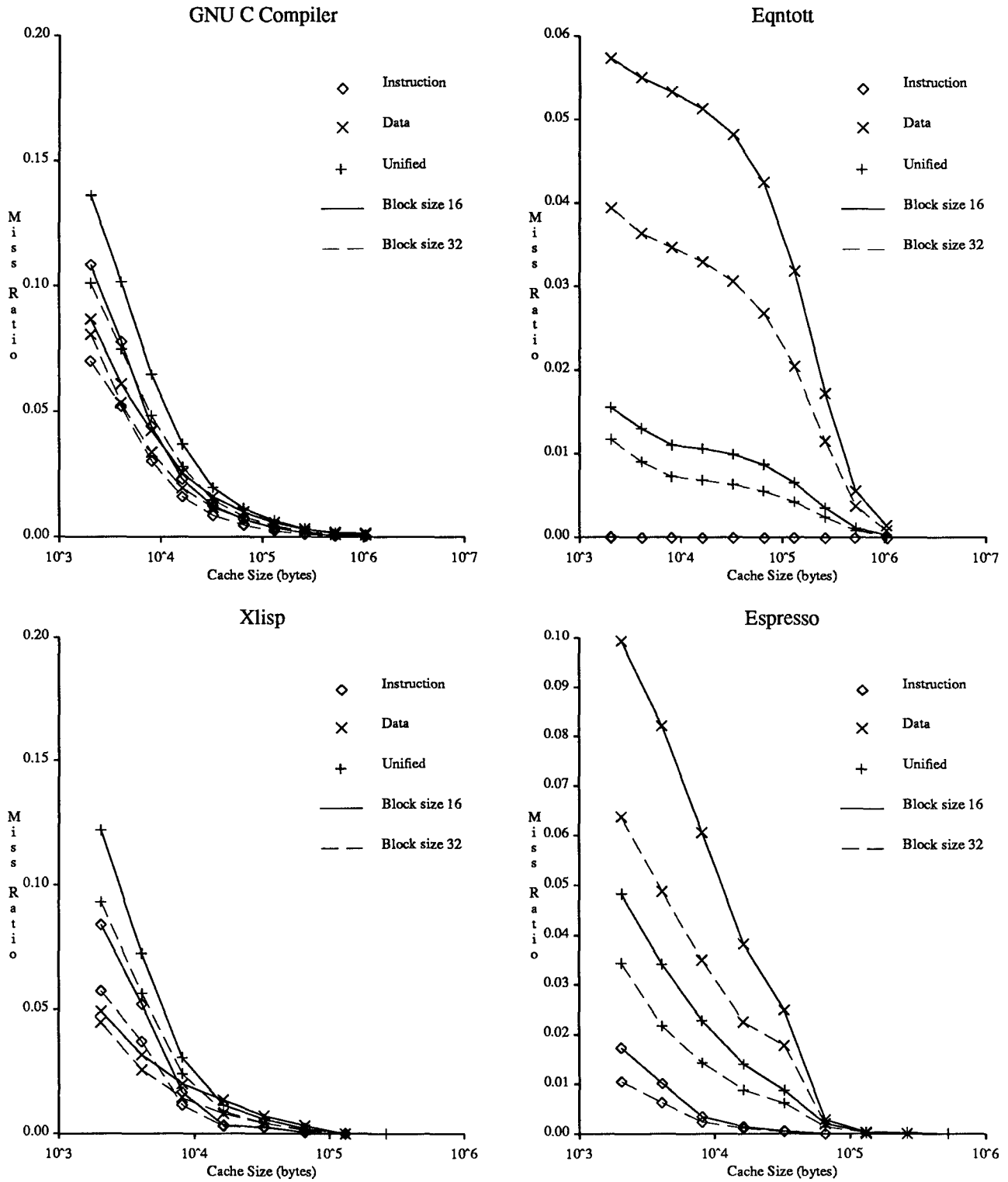
Figure 2. Change in miss ratio as a function of block size. Each point is the ratio of the miss ratio for a cache with block size 2*B divided by the miss ratio for a cache of block size B. This value gives the benefits of doubling the block size for two-way set-associative caches of size 16 and 64 Kbytes. Smaller values mean that doubling block size is more worthwhile. The instruction cache is omitted where the miss ratio is zero (Eqntott both sizes, Espresso 32Kbytes).

Figure 3. Change in miss ratio as a function of associativity. Each point is the ratio of miss ratio for a cache of associativity 2*A divided by the miss ratio for a cache of associativity A. This value gives the benefits of doubling the associativity for block size 32 bytes. Smaller values mean that doubling the associativity is more worthwhile. The instruction cache is omitted where the miss ratio is zero (Eqntott both sizes, Espresso 32Kbytes).

Appendix. Complete tables of miss ratios. Note that a value 0 means that the miss ratio is less than 0.0001.

Eqntott : Associativity 1				
Instruction				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
1K	0.0158	0.0116	0.0066	0.0066	0.0067
2K	0.0158	0.0116	0.0066	0.0066	0.0066
4K	0	0	0	0	0
8K	0	0	0	0	0
16K	0	0	0	0	0
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
1K	0.1073	0.1086	0.1748	0.2823	0.4642
2K	0.0817	0.0728	0.0958	0.1545	0.2699
4K	0.0675	0.0533	0.0604	0.0892	0.1567
8K	0.0595	0.0431	0.0431	0.0560	0.0817
16K	0.0545	0.0373	0.0338	0.0387	0.0498
32K	0.0496	0.0326	0.0275	0.0277	0.0301
64K	0.0432	0.0280	0.0232	0.0215	0.0206
128K	0.0336	0.0217	0.0179	0.0158	0.0140
256K	0.0207	0.0136	0.0116	0.0107	0.0093
512K	0.0103	0.0068	0.0060	0.0060	0.0053
1M	0.0031	0.0021	0.0018	0.0019	0.0017
Unified				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
1K	0.0543	0.0577	0.0776	0.1938	0.2540
2K	0.0387	0.0368	0.0420	0.0682	0.1223
4K	0.0192	0.0178	0.0214	0.0353	0.0714
8K	0.0148	0.0121	0.0132	0.0200	0.0412
16K	0.0125	0.0094	0.0092	0.0122	0.0266
32K	0.0109	0.0075	0.0067	0.0076	0.0102
64K	0.0091	0.0061	0.0052	0.0053	0.0057
128K	0.0070	0.0046	0.0038	0.0036	0.0036
256K	0.0044	0.0029	0.0025	0.0024	0.0024
512K	0.0022	0.0015	0.0014	0.0014	0.0014
1M	0.0007	0.0005	0.0005	0.0005	0.0006

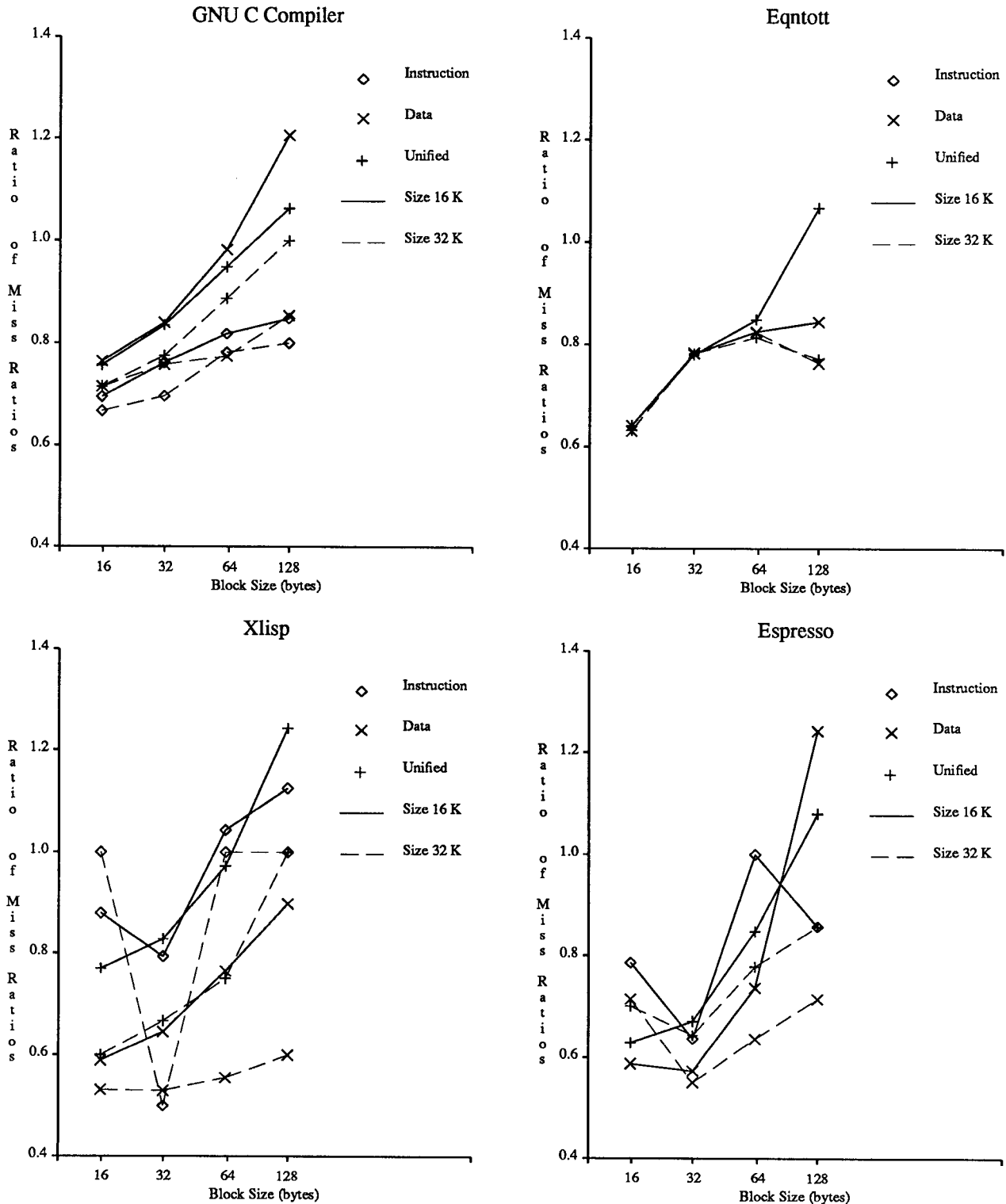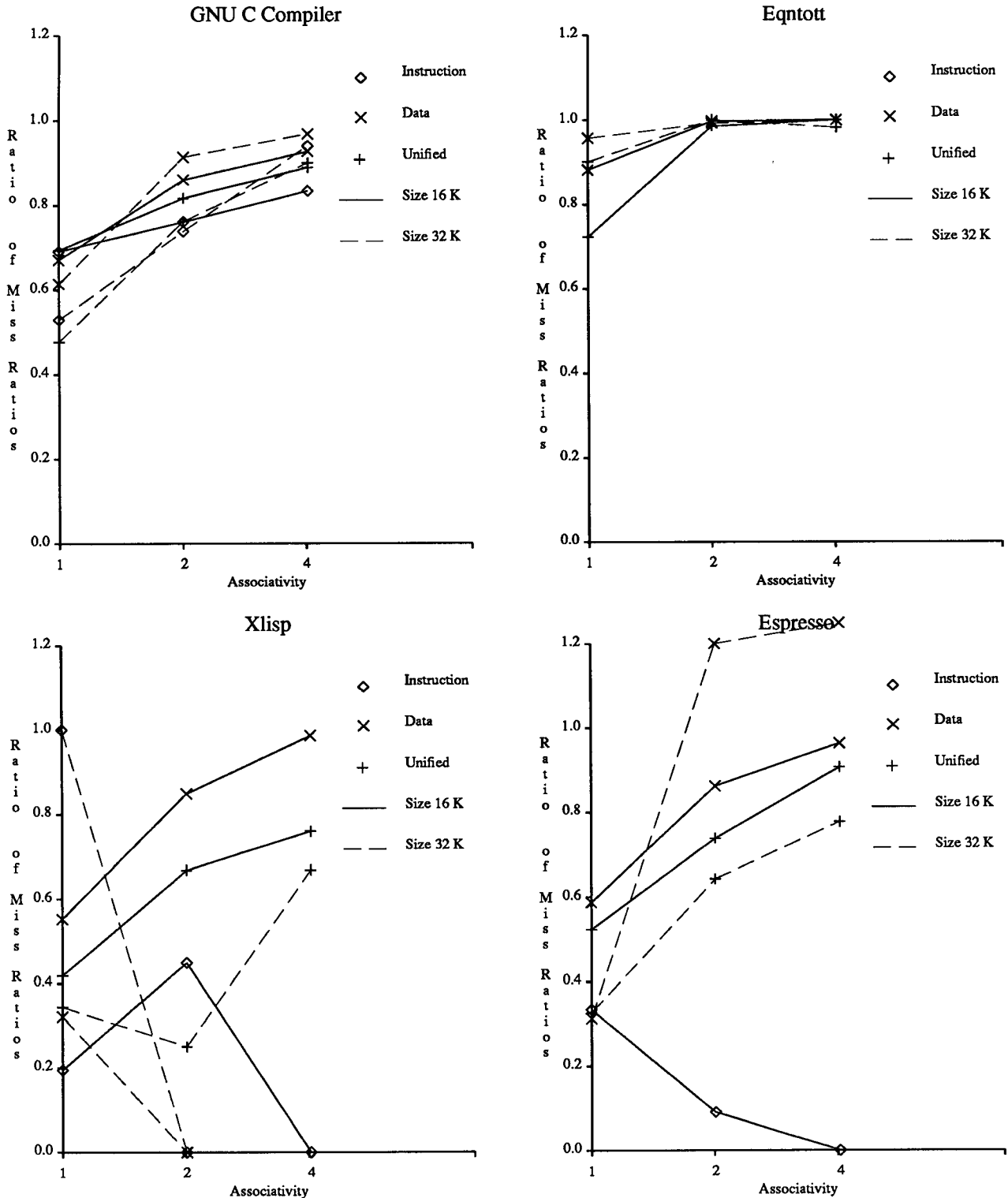| Eqntott : Associativity 2				
Instruction				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
2K	0	0	0	0	0
4K	0	0	0	0	0
8K	0	0	0	0	0
16K	0	0	0	0	0
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
2K	0.0573	0.0394	0.0422	0.0610	0.1307
4K	0.0550	0.0363	0.0303	0.0317	0.0554
8K	0.0533	0.0346	0.0273	0.0241	0.0304
16K	0.0513	0.0329	0.0257	0.0212	0.0179
32K	0.0482	0.0306	0.0238	0.0196	0.0153
64K	0.0425	0.0268	0.0210	0.0173	0.0132
128K	0.0318	0.0204	0.0166	0.0142	0.0108
256K	0.0172	0.0115	0.0104	0.0096	0.0077
512K	0.0055	0.0037	0.0037	0.0041	0.0037
1M	0.0014	0.0008	0.0007	0.0008	0.0008
Unified				
Size	Block size (bytes)			
(bytes)	16	32	64	128	256
2K	0.0155	0.0117	0.0139	0.0589	0.1061
4K	0.0130	0.0090	0.0079	0.0101	0.0234
8K	0.0110	0.0072	0.0058	0.0055	0.0092
16K	0.0106	0.0068	0.0053	0.0045	0.0048
32K	0.0099	0.0063	0.0049	0.0040	0.0037
64K	0.0087	0.0055	0.0043	0.0035	0.0027
128K	0.0065	0.0042	0.0034	0.0029	0.0022
256K	0.0035	0.0024	0.0021	0.0020	0.0016
512K	0.0011	0.0008	0.0008	0.0008	0.0008
1M	0.0003	0.0002	0.0001	0.0002	0.0002

Eqntott : Associativity 4					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0	0	0	0	0
8K	0	0	0	0	0
16K	0	0	0	0	0
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0543	0.0352	0.0277	0.0237	0.0373
8K	0.0531	0.0342	0.0266	0.0215	0.0179
16K	0.0513	0.0328	0.0254	0.0205	0.0157
32K	0.0481	0.0305	0.0237	0.0193	0.0145
64K	0.0425	0.0266	0.0208	0.0172	0.0131
128K	0.0316	0.0203	0.0165	0.0139	0.0105
256K	0.0141	0.0099	0.0097	0.0094	0.0075
512K	0.0031	0.0022	0.0026	0.0032	0.0031
1M	0.0010	0.0005	0.0003	0.0003	0.0004
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0112	0.0073	0.0058	0.0062	0.0159
8K	0.0109	0.0070	0.0055	0.0045	0.0048
16K	0.0105	0.0067	0.0052	0.0042	0.0033
32K	0.0099	0.0062	0.0049	0.0039	0.0030
64K	0.0087	0.0055	0.0043	0.0035	0.0027
128K	0.0065	0.0041	0.0034	0.0028	0.0021
256K	0.0029	0.0020	0.0020	0.0019	0.0015
512K	0.0006	0.0005	0.0005	0.0007	0.0006
1M	0.0002	0.0001	0	0	0

Eqntott : Associativity 8					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0	0	0	0	0
16K	0	0	0	0	0
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0530	0.0342	0.0265	0.0214	0.0164
16K	0.0512	0.0328	0.0254	0.0205	0.0155
32K	0.0481	0.0304	0.0237	0.0193	0.0145
64K	0.0425	0.0266	0.0207	0.0171	0.0131
128K	0.0317	0.0202	0.0164	0.0138	0.0104
256K	0.0117	0.0088	0.0095	0.0094	0.0075
512K	0.0017	0.0011	0.0012	0.0022	0.0027
1M	0.0009	0.0005	0.0002	0.0001	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0109	0.0070	0.0054	0.0044	0.0038
16K	0.0105	0.0067	0.0052	0.0042	0.0032
32K	0.0099	0.0062	0.0049	0.0039	0.0030
64K	0.0087	0.0054	0.0042	0.0035	0.0027
128K	0.0065	0.0041	0.0034	0.0028	0.0021
256K	0.0024	0.0018	0.0019	0.0019	0.0015
512K	0.0004	0.0002	0.0003	0.0005	0.0006
1M	0.0002	0	0	0	0

Xlisp : Associativity 1

Instruction

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
1K	0.1516	0.0985	0.0710	0.0575	0.0564
2K	0.1007	0.0681	0.0492	0.0415	0.0410
4K	0.0595	0.0415	0.0299	0.0237	0.0244
8K	0.0293	0.0195	0.0144	0.0134	0.0146
16K	0.0229	0.0149	0.0106	0.0098	0.0114
32K	0.0049	0.0035	0.0024	0.0020	0.0018
64K	0.0006	0.0004	0.0002	0.0002	0.0002
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Data

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
1K	0.1194	0.1170	0.1195	0.1336	0.1630
2K	0.0784	0.0756	0.0772	0.0859	0.1036
4K	0.0503	0.0447	0.0426	0.0472	0.0546
8K	0.0274	0.0218	0.0200	0.0209	0.0243
16K	0.0199	0.0143	0.0122	0.0130	0.0150
32K	0.0164	0.0111	0.0083	0.0077	0.0081
64K	0.0090	0.0053	0.0034	0.0029	0.0027
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Unified

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
1K	0.2015	0.1642	0.1506	0.1751	0.2434
2K	0.1464	0.1206	0.1078	0.1187	0.1583
4K	0.0895	0.0712	0.0609	0.0656	0.0871
8K	0.0495	0.0399	0.0353	0.0402	0.0559
16K	0.0282	0.0208	0.0175	0.0179	0.0233
32K	0.0122	0.0093	0.0078	0.0074	0.0092
64K	0.0049	0.0035	0.0029	0.0030	0.0043
128K	0.0004	0.0003	0.0002	0.0002	0.0002
256K	0.0004	0.0003	0.0002	0.0002	0.0002
512K	0.0004	0.0003	0.0002	0.0002	0.0002
1M	0.0004	0.0003	0.0002	0.0002	0.0002

Xlisp : Associativity 2

Instruction

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
2K	0.0843	0.0575	0.0422	0.0357	0.0359
4K	0.0521	0.0371	0.0296	0.0249	0.0243
8K	0.0166	0.0114	0.0103	0.0094	0.0095
16K	0.0033	0.0029	0.0023	0.0024	0.0027
32K	0.0027	0.0024	0.0019	0.0021	0.0021
64K	0.0004	0.0004	0.0002	0.0002	0.0002
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Data

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
2K	0.0493	0.0447	0.0455	0.0516	0.0601
4K	0.0318	0.0257	0.0233	0.0225	0.0270
8K	0.0200	0.0144	0.0118	0.0099	0.0104
16K	0.0134	0.0079	0.0051	0.0039	0.0035
32K	0.0070	0.0041	0.0026	0.0020	0.0017
64K	0.0032	0.0017	0.0009	0.0005	0.0003
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Unified

Size (bytes)	Block size (bytes)				
	16	32	64	128	256
2K	0.1221	0.0934	0.0755	0.0676	0.0722
4K	0.0726	0.0565	0.0482	0.0438	0.0455
8K	0.0306	0.0240	0.0205	0.0189	0.0224
16K	0.0113	0.0087	0.0072	0.0070	0.0087
32K	0.0058	0.0042	0.0032	0.0031	0.0033
64K	0.0020	0.0012	0.0008	0.0006	0.0006
128K	0.0003	0.0002	0.0001	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Xlisp : Associativity 4					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0244	0.0199	0.0175	0.0175	0.0181
8K	0.0091	0.0076	0.0076	0.0074	0.0085
16K	0.0012	0.0013	0.0013	0.0015	0.0015
32K	0	0	0	0.0001	0.0002
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0266	0.0187	0.0138	0.0136	0.0190
8K	0.0169	0.0113	0.0084	0.0064	0.0057
16K	0.0126	0.0067	0.0041	0.0027	0.0019
32K	0.0067	0.0034	0.0018	0.0009	0.0005
64K	0.0002	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0518	0.0440	0.0394	0.0361	0.0379
8K	0.0233	0.0182	0.0167	0.0164	0.0218
16K	0.0078	0.0058	0.0047	0.0044	0.0049
32K	0.0035	0.0020	0.0014	0.0011	0.0011
64K	0.0005	0.0003	0.0002	0.0002	0.0002
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Xlisp : Associativity 8					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0011	0.0014	0.0017	0.0030	0.0044
16K	0	0	0.0002	0.0006	0.0008
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0159	0.0105	0.0076	0.0058	0.0045
16K	0.0124	0.0066	0.0037	0.0023	0.0018
32K	0.0069	0.0036	0.0020	0.0010	0.0005
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0153	0.0119	0.0115	0.0120	0.0167
16K	0.0059	0.0044	0.0038	0.0036	0.0044
32K	0.0034	0.0018	0.0012	0.0008	0.0008
64K	0.0004	0.0002	0.0001	0.0001	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

GNU C Compiler : Associativity 1					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
1K	0.1414	0.0919	0.0635	0.0465	0.0371
2K	0.1182	0.0769	0.0530	0.0387	0.0295
4K	0.0876	0.0580	0.0410	0.0304	0.0229
8K	0.0605	0.0407	0.0297	0.0225	0.0173
16K	0.0339	0.0230	0.0167	0.0132	0.0105
32K	0.0210	0.0139	0.0101	0.0080	0.0063
64K	0.0131	0.0087	0.0063	0.0048	0.0038
128K	0.0063	0.0042	0.0030	0.0022	0.0018
256K	0.0034	0.0022	0.0016	0.0012	0.0011
512K	0.0013	0.0009	0.0007	0.0005	0.0004
1M	0.0006	0.0004	0.0003	0.0002	0.0002
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
1K	0.1640	0.1646	0.1835	0.2276	0.2987
2K	0.1190	0.1153	0.1237	0.1496	0.1979
4K	0.0841	0.0771	0.0789	0.0907	0.1204
8K	0.0547	0.0481	0.0472	0.0537	0.0707
16K	0.0344	0.0289	0.0274	0.0294	0.0366
32K	0.0218	0.0176	0.0159	0.0161	0.0198
64K	0.0146	0.0114	0.0099	0.0093	0.0105
128K	0.0089	0.0067	0.0056	0.0052	0.0058
256K	0.0045	0.0032	0.0025	0.0022	0.0022
512K	0.0029	0.0020	0.0016	0.0013	0.0013
1M	0.0017	0.0010	0.0007	0.0006	0.0006
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
1K	0.2235	0.1823	0.1783	0.2039	0.2692
2K	0.1826	0.1434	0.1345	0.1437	0.1811
4K	0.1422	0.1105	0.1021	0.1036	0.1210
8K	0.1067	0.0829	0.0782	0.0782	0.0873
16K	0.0518	0.0402	0.0341	0.0352	0.0421
32K	0.0330	0.0254	0.0213	0.0228	0.0282
64K	0.0218	0.0168	0.0142	0.0154	0.0194
128K	0.0120	0.0092	0.0078	0.0086	0.0101
256K	0.0063	0.0049	0.0042	0.0040	0.0043
512K	0.0036	0.0030	0.0027	0.0028	0.0032
1M	0.0023	0.0020	0.0019	0.0021	0.0025

GNU C Compiler : Associativity 2					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
2K	0.1083	0.0699	0.0486	0.0352	0.0272
4K	0.0776	0.0520	0.0371	0.0273	0.0209
8K	0.0441	0.0302	0.0230	0.0183	0.0145
16K	0.0229	0.0159	0.0121	0.0099	0.0084
32K	0.0123	0.0085	0.0061	0.0051	0.0043
64K	0.0069	0.0046	0.0032	0.0025	0.0020
128K	0.0036	0.0024	0.0017	0.0013	0.0010
256K	0.0014	0.0010	0.0008	0.0006	0.0005
512K	0.0006	0.0004	0.0003	0.0002	0.0002
1M	0.0003	0.0002	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
2K	0.0867	0.0806	0.0824	0.0992	0.1473
4K	0.0608	0.0533	0.0513	0.0575	0.0789
8K	0.0424	0.0337	0.0305	0.0323	0.0408
16K	0.0254	0.0194	0.0163	0.0160	0.0193
32K	0.0159	0.0117	0.0090	0.0076	0.0079
64K	0.0098	0.0070	0.0053	0.0041	0.0035
128K	0.0057	0.0040	0.0030	0.0022	0.0018
256K	0.0030	0.0020	0.0013	0.0010	0.0007
512K	0.0018	0.0010	0.0006	0.0004	0.0003
1M	0.0014	0.0007	0.0004	0.0002	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
2K	0.1362	0.1010	0.0832	0.0813	0.0963
4K	0.1015	0.0746	0.0599	0.0555	0.0603
8K	0.0646	0.0483	0.0399	0.0371	0.0384
16K	0.0369	0.0279	0.0233	0.0221	0.0235
32K	0.0196	0.0146	0.0119	0.0114	0.0114
64K	0.0112	0.0080	0.0062	0.0055	0.0055
128K	0.0064	0.0044	0.0034	0.0028	0.0026
256K	0.0031	0.0022	0.0016	0.0013	0.0011
512K	0.0014	0.0009	0.0007	0.0005	0.0005
1M	0.0007	0.0004	0.0003	0.0002	0.0002

GNU C Compiler : Associativity 4

Instruction

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0717	0.0487	0.0350	0.0261	0.0200
8K	0.0369	0.0261	0.0206	0.0170	0.0136
16K	0.0171	0.0121	0.0094	0.0081	0.0072
32K	0.0089	0.0060	0.0044	0.0036	0.0032
64K	0.0054	0.0034	0.0024	0.0018	0.0015
128K	0.0028	0.0020	0.0014	0.0011	0.0008
256K	0.0009	0.0007	0.0005	0.0005	0.0004
512K	0.0003	0.0002	0.0001	0	0
1M	0.0003	0.0001	0	0	0

Data

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0533	0.0437	0.0411	0.0461	0.0628
8K	0.0389	0.0287	0.0243	0.0249	0.0326
16K	0.0224	0.0167	0.0137	0.0129	0.0149
32K	0.0145	0.0103	0.0076	0.0059	0.0054
64K	0.0091	0.0064	0.0047	0.0034	0.0026
128K	0.0050	0.0034	0.0025	0.0018	0.0013
256K	0.0026	0.0016	0.0011	0.0008	0.0005
512K	0.0016	0.0009	0.0006	0.0004	0.0002
1M	0.0013	0.0007	0.0003	0.0002	0

Unified

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0942	0.0686	0.0540	0.0467	0.0478
8K	0.0577	0.0436	0.0357	0.0315	0.0302
16K	0.0301	0.0228	0.0190	0.0179	0.0180
32K	0.0154	0.0114	0.0091	0.0082	0.0084
64K	0.0088	0.0061	0.0046	0.0038	0.0036
128K	0.0053	0.0036	0.0026	0.0020	0.0016
256K	0.0024	0.0017	0.0012	0.0010	0.0008
512K	0.0010	0.0006	0.0004	0.0004	0.0003
1M	0.0006	0.0003	0.0002	0	0

GNU C Compiler : Associativity 8

Instruction

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0323	0.0241	0.0197	0.0167	0.0136
16K	0.0144	0.0101	0.0079	0.0071	0.0066
32K	0.0081	0.0054	0.0039	0.0030	0.0027
64K	0.0051	0.0032	0.0022	0.0016	0.0013
128K	0.0026	0.0019	0.0014	0.0010	0.0007
256K	0.0005	0.0003	0.0004	0.0004	0.0004
512K	0.0003	0.0001	0	0	0
1M	0.0003	0.0001	0	0	0

Data

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0375	0.0270	0.0223	0.0224	0.0303
16K	0.0214	0.0155	0.0122	0.0107	0.0120
32K	0.0121	0.0087	0.0066	0.0051	0.0045
64K	0.0087	0.0062	0.0046	0.0033	0.0023
128K	0.0048	0.0032	0.0024	0.0017	0.0012
256K	0.0025	0.0016	0.0010	0.0007	0.0005
512K	0.0016	0.0009	0.0005	0.0003	0.0002
1M	0.0013	0.0007	0.0003	0.0002	0

Unified

Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0532	0.0411	0.0345	0.0307	0.0290
16K	0.0272	0.0203	0.0172	0.0168	0.0172
32K	0.0135	0.0099	0.0079	0.0071	0.0073
64K	0.0081	0.0055	0.0041	0.0033	0.0029
128K	0.0050	0.0033	0.0023	0.0017	0.0014
256K	0.0019	0.0014	0.0011	0.0009	0.0007
512K	0.0008	0.0005	0.0003	0.0003	0.0002
1M	0.0006	0.0003	0.0002	0	0

Espresso : Associativity 1						Espresso : Associativity 2				
Instruction						Instruction				
Size	Block size (bytes)					Size	Block size (bytes)				
(bytes)	16	32	64	128	256	(bytes)	16	32	64	128	256
1K	0.0323	0.0195	0.0121	0.0105	0.0135						
2K	0.0242	0.0146	0.0090	0.0081	0.0065	2K	0.0173	0.0105	0.0065	0.0048	0.0038
4K	0.0181	0.0109	0.0068	0.0066	0.0054	4K	0.0101	0.0062	0.0040	0.0033	0.0028
8K	0.0083	0.0050	0.0032	0.0025	0.0021	8K	0.0034	0.0023	0.0015	0.0014	0.0015
16K	0.0053	0.0033	0.0021	0.0016	0.0014	16K	0.0014	0.0011	0.0007	0.0007	0.0006
32K	0.0034	0.0021	0.0013	0.0012	0.0010	32K	0.0005	0.0004	0.0003	0.0003	0.0003
64K	0.0031	0.0019	0.0012	0.0011	0.0009	64K	0	0	0	0	0
128K	0.0030	0.0019	0.0012	0.0010	0.0009	128K	0	0	0	0	0
256K	0.0029	0.0018	0.0012	0.0010	0.0008	256K	0	0	0	0	0
512K	0	0	0	0	0	512K	0	0	0	0	0
1M	0	0	0	0	0	1M	0	0	0	0	0
Data						Data					
Size	Block size (bytes)					Size	Block size (bytes)				
(bytes)	16	32	64	128	256	(bytes)	16	32	64	128	256
1K	0.1591	0.1405	0.1724	0.2256	0.3287						
2K	0.1219	0.0948	0.1079	0.1326	0.1890	2K	0.0993	0.0637	0.0532	0.0636	0.0998
4K	0.0989	0.0722	0.0742	0.0845	0.1106	4K	0.0821	0.0488	0.0351	0.0317	0.0398
8K	0.0764	0.0541	0.0546	0.0596	0.0737	8K	0.0606	0.0350	0.0217	0.0171	0.0200
16K	0.0576	0.0383	0.0384	0.0375	0.0412	16K	0.0383	0.0225	0.0129	0.0095	0.0118
32K	0.0268	0.0183	0.0114	0.0095	0.0108	32K	0.0249	0.0178	0.0094	0.0053	0.0035
64K	0.0095	0.0064	0.0045	0.0047	0.0062	64K	0.0028	0.0020	0.0011	0.0007	0.0005
128K	0.0043	0.0030	0.0018	0.0014	0.0015	128K	0.0003	0.0001	0	0	0
256K	0.0001	0	0	0	0	256K	0	0	0	0	0
512K	0	0	0	0	0	512K	0	0	0	0	0
1M	0	0	0	0	0	1M	0	0	0	0	0
Unified						Unified					
Size	Block size (bytes)					Size	Block size (bytes)				
(bytes)	16	32	64	128	256	(bytes)	16	32	64	128	256
1K	0.1229	0.1143	0.1250	0.1490	0.2128						
2K	0.0832	0.0684	0.0705	0.0834	0.1209	2K	0.0483	0.0344	0.0312	0.0375	0.0612
4K	0.0511	0.0387	0.0374	0.0449	0.0628	4K	0.0342	0.0217	0.0164	0.0167	0.0228
8K	0.0328	0.0239	0.0228	0.0248	0.0371	8K	0.0228	0.0143	0.0097	0.0091	0.0105
16K	0.0239	0.0168	0.0158	0.0163	0.0240	16K	0.0140	0.0088	0.0059	0.0050	0.0054
32K	0.0137	0.0097	0.0075	0.0074	0.0108	32K	0.0087	0.0061	0.0038	0.0028	0.0026
64K	0.0063	0.0043	0.0032	0.0034	0.0052	64K	0.0020	0.0014	0.0009	0.0007	0.0006
128K	0.0044	0.0028	0.0019	0.0017	0.0018	128K	0.0003	0.0002	0.0002	0.0001	0.0001
256K	0.0026	0.0016	0.0011	0.0010	0.0008	256K	0.0002	0	0	0	0
512K	0.0003	0.0002	0.0001	0.0001	0.0001	512K	0	0	0	0	0
1M	0.0003	0.0002	0.0001	0.0001	0.0001	1M	0	0	0	0	0

Espresso : Associativity 4					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0092	0.0059	0.0038	0.0032	0.0025
8K	0.0017	0.0012	0.0008	0.0006	0.0010
16K	0.0002	0.0001	0	0	0.0001
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0792	0.0460	0.0280	0.0208	0.0214
8K	0.0608	0.0344	0.0191	0.0117	0.0092
16K	0.0332	0.0194	0.0105	0.0061	0.0039
32K	0.0257	0.0158	0.0081	0.0042	0.0023
64K	0.0026	0.0024	0.0012	0.0007	0.0004
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
4K	0.0325	0.0201	0.0131	0.0109	0.0131
8K	0.0210	0.0127	0.0079	0.0060	0.0056
16K	0.0107	0.0065	0.0039	0.0028	0.0026
32K	0.0070	0.0043	0.0023	0.0014	0.0011
64K	0.0011	0.0009	0.0005	0.0004	0.0003
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0

Espresso : Associativity 8					
Instruction					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0012	0.0008	0.0005	0.0005	0.0011
16K	0.0001	0	0	0	0
32K	0	0	0	0	0
64K	0	0	0	0	0
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Data					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0616	0.0345	0.0186	0.0108	0.0080
16K	0.0320	0.0187	0.0100	0.0056	0.0034
32K	0.0264	0.0155	0.0079	0.0040	0.0022
64K	0.0029	0.0030	0.0015	0.0008	0.0004
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0
Unified					
Size	Block size (bytes)				
(bytes)	16	32	64	128	256
8K	0.0202	0.0123	0.0076	0.0058	0.0050
16K	0.0099	0.0059	0.0034	0.0023	0.0020
32K	0.0064	0.0038	0.0020	0.0011	0.0007
64K	0.0008	0.0007	0.0004	0.0002	0.0001
128K	0	0	0	0	0
256K	0	0	0	0	0
512K	0	0	0	0	0
1M	0	0	0	0	0