CSCE 614: Computer Architecture

Assignment 2

Sridhar Mareguddi October 7, 2013
UIN: 823000772

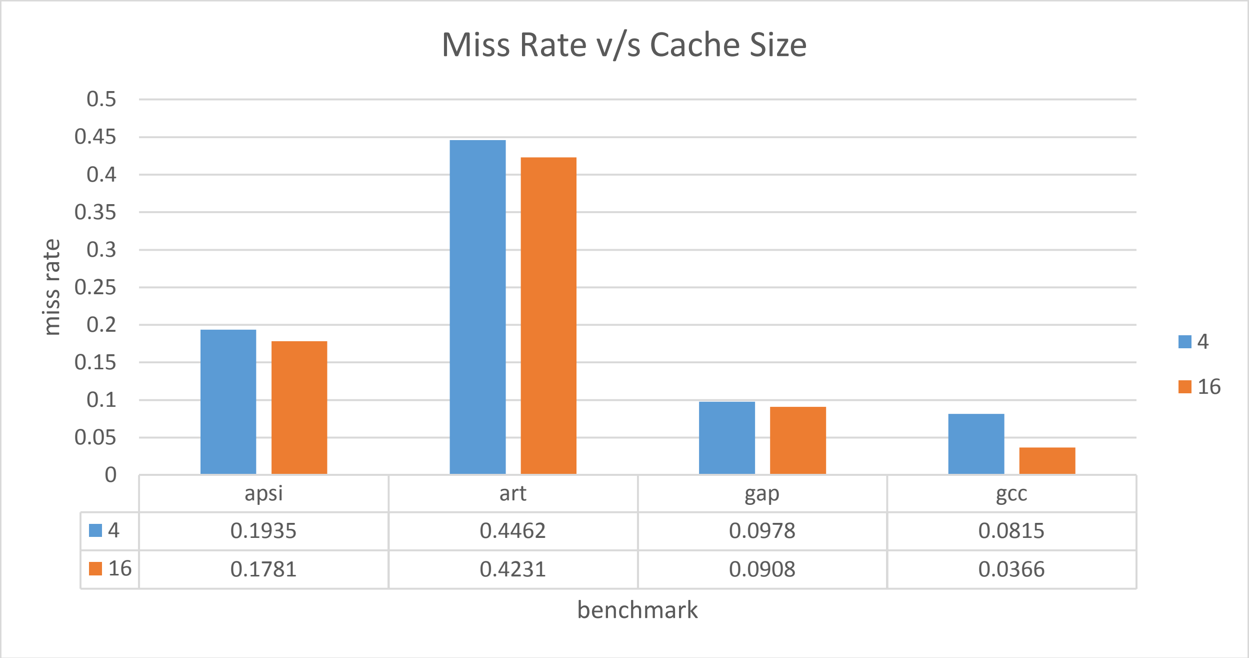
Part A

Configurations	Size	Associativity	Cache block size	Replacement policy
1 *(baseline)*	4 KB	Direct mapped	32 B	
2	4 KB	{4 , 8, fully}	32B	{LRU, Random}
3	4 KB	Direct mapped	64B	
4	16 KB	Direct mapped	32B	

Recorded Values

Number	BenchMark	Cache_Size (kB)	Sets	Block_Size	Associativity	Associativity_N	Replacement_Policy	L1 Miss Rate	L2 Miss Rate
1	gap	4	128	32	1	1	LRU	0.0978	0.231
2	gap	4	32	32	4	4	LRU	0.0914	0.2412
3	gap	4	32	32	4	4	Random	0.0936	0.2377
4	gap	4	16	32	8	8	LRU	0.0909	0.2415
5	gap	4	16	32	8	8	Random	0.0932	0.2383
6	gap	4	1	32	128	FA	LRU	0.0906	0.2419
7	gap	4	1	32	128	FA	Random	0.0932	0.2386
8	gap	4	64	64	1	1	LRU	0.0577	0.3896
9	gap	16	512	32	1	1	LRU	0.0908	0.2417
10	gcc	4	128	32	1	1	LRU	0.0815	0.148
11	gcc	4	32	32	4	4	LRU	0.0452	0.1948
12	gcc	4	32	32	4	4	Random	0.0553	0.1797
13	gcc	4	16	32	8	8	LRU	0.0422	0.1986
14	gcc	4	16	32	8	8	Random	0.0534	0.1819
15	gcc	4	1	32	128	FA	LRU	0.0403	0.201
16	gcc	4	1	32	128	FA	Random	0.053	0.1824
17	gcc	4	64	64	1	1	LRU	0.0834	0.1491
18	gcc	16	512	32	1	1	LRU	0.0366	0.1901
19	apsi	4	128	32	1	1	LRU	0.1935	0.2133
20	apsi	4	32	32	4	4	LRU	0.1748	0.2277
21	apsi	4	32	32	4	4	Random	0.1762	0.2266
22	apsi	4	16	32	8	8	LRU	0.174	0.2281
23	apsi	4	16	32	8	8	Random	0.1755	0.2269
24	apsi	4	1	32	128	FA	LRU	0.1746	0.2276
25	apsi	4	1	32	128	FA	Random	0.1758	0.2265
26	apsi	4	64	64	1	1	LRU	0.115	0.3536
27	apsi	16	512	32	1	1	LRU	0.1781	0.2232
28	art	4	128	32	1	1	LRU	0.4462	0.6841
29	art	4	32	32	4	4	LRU	0.4184	0.7283
30	art	4	32	32	4	4	Random	0.4388	0.696
31	art	4	16	32	8	8	LRU	0.4183	0.7284
32	art	4	16	32	8	8	Random	0.4538	0.6764
33	art	4	1	32	128	FA	LRU	0.4151	0.7333
34	art	4	1	32	128	FA	Random	0.4554	0.6745
35	art	4	64	64	1	1	LRU	0.4141	0.7239
36	art	16	512	32	1	1	LRU	0.4231	0.72

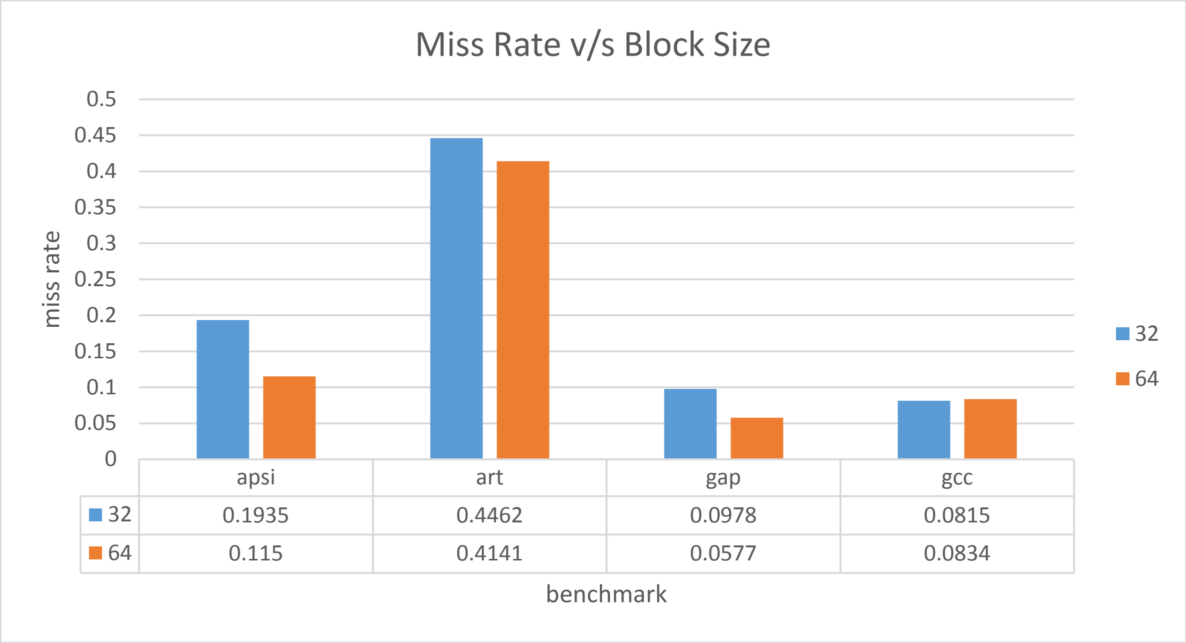
Sensitivity of L1 cache to changes in Cache size



	apsi	art	gap	gcc
🟦 4	0.1935	0.4462	0.0978	0.0815
🟧 16	0.1781	0.4231	0.0908	0.0366

As we can see from the figure, the increase in cache size results in the reduction in miss rate because more blocks can be accommodated in a larger cache and this reduces *capacity misses*. This can be seen for all the benchmarks. The disadvantage of larger cache is longer hit time and higher cost.

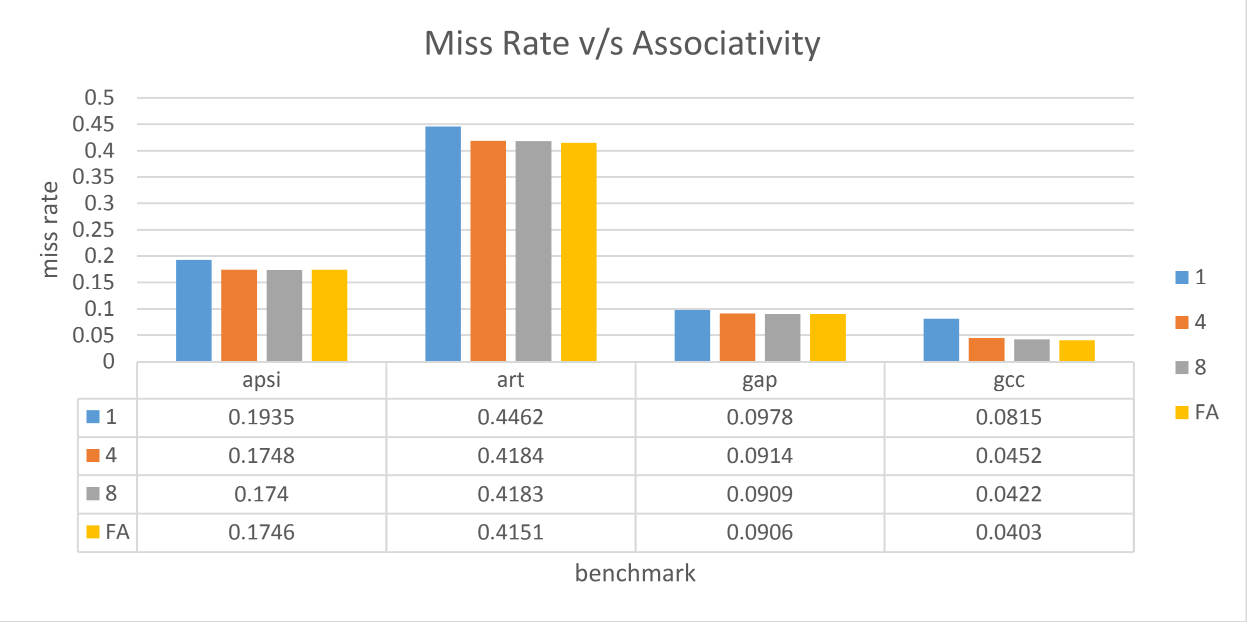
Sensitivity of L1 caches to changes in Block size



	apsi	art	gap	gcc
🟦 32	0.1935	0.4462	0.0978	0.0815
🟧 64	0.115	0.4141	0.0577	0.0834

From the above graph, we can see that the number of miss rate tends to be decrease when we increase the block size. It is because the larger blocks take advantage of *spatial locality* and reduce compulsory miss. However when the block size becomes large enough approaching the cache size, the increase in block size reduce the number of blocks in a cache, increases conflict and capacity misses. Note that there is no significant improvement in the miss rate when the block size is increased from 32 to 64 bytes when the cache size is fixed because it has increased conflict misses.

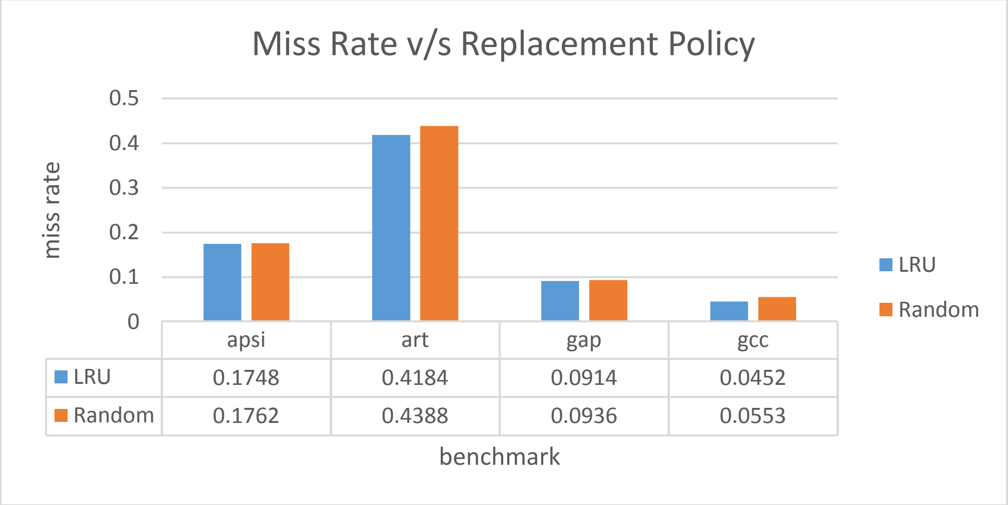
Sensitivity of L1 caches to changes in Associativity



	apsi	art	gap	gcc
1	0.1935	0.4462	0.0978	0.0815
4	0.1748	0.4184	0.0914	0.0452
8	0.174	0.4183	0.0909	0.0422
FA	0.1746	0.4151	0.0906	0.0403

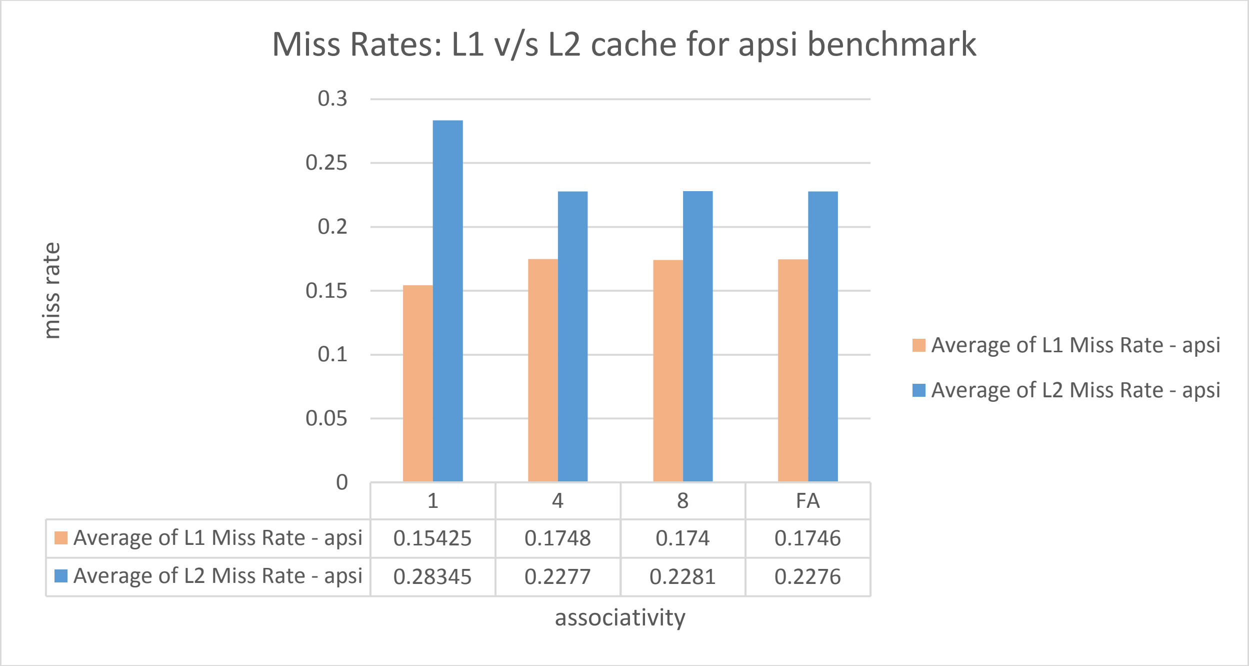
As seen from the above graph, the miss rate decreases with the increase in the cache size because each set has now more blocks and hence associativity reduces the conflict misses. Increasing associativity maximizes cache hits but may increases cache hit penalty due to slower cycle time due to the inclusion of comparator and other hardware. Current practices utilizes 1- to 4-way set associative caches. There will be no significant reduction in miss rate when the associativity is increased beyond 4.

Sensitivity of L1 caches to changes in Replacement policy

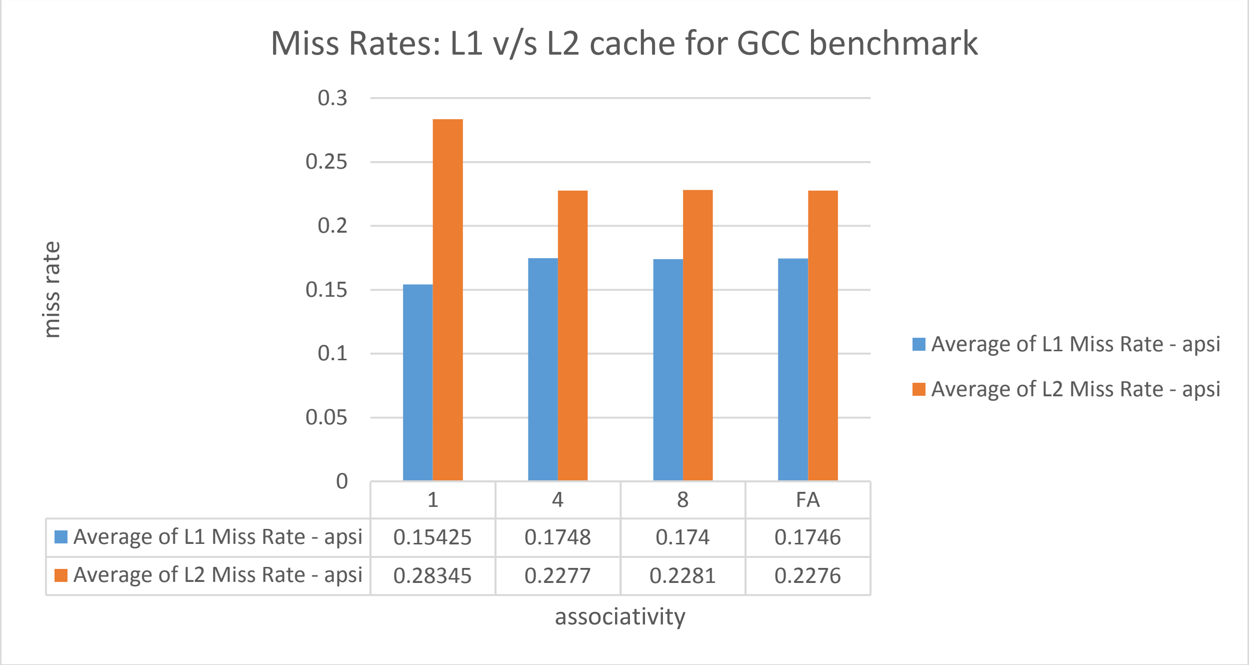


	apsi	art	gap	gcc
■ LRU	0.1748	0.4184	0.0914	0.0452
■ Random	0.1762	0.4388	0.0936	0.0553

Effect of L1 on L2 cache:



	1	4	8	FA
■ Average of L1 Miss Rate - apsi	0.15425	0.1748	0.174	0.1746
■ Average of L2 Miss Rate - apsi	0.28345	0.2277	0.2281	0.2276



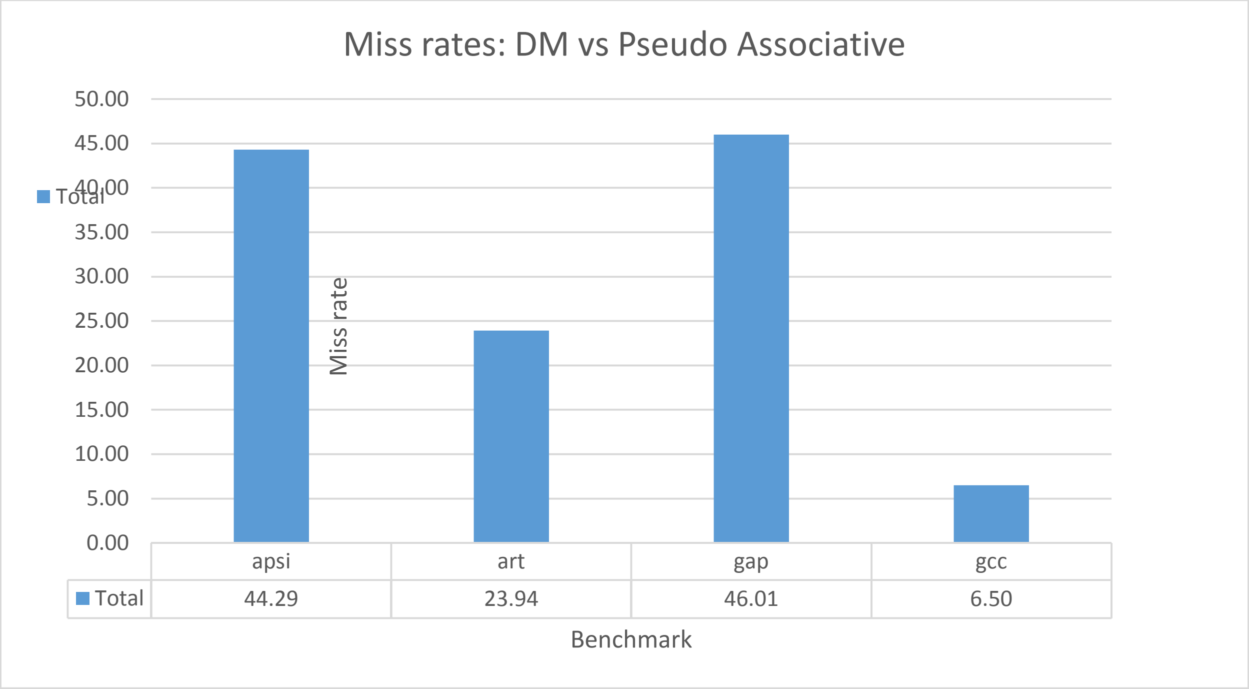
As we can see from the figure, L1 miss rate has reduced and L2 miss rate have increased. Assuming the L1 hit rate to say 90%, the access to L2 cache is only 10%. Assuming L2 cache to be inclusive, all the data in L1 cache is included in L2 and the new blocks are less. The probability of finding L1 missed data in L2 is hence less. Thus the miss rate increases in L2 as miss rate decreases in L1.

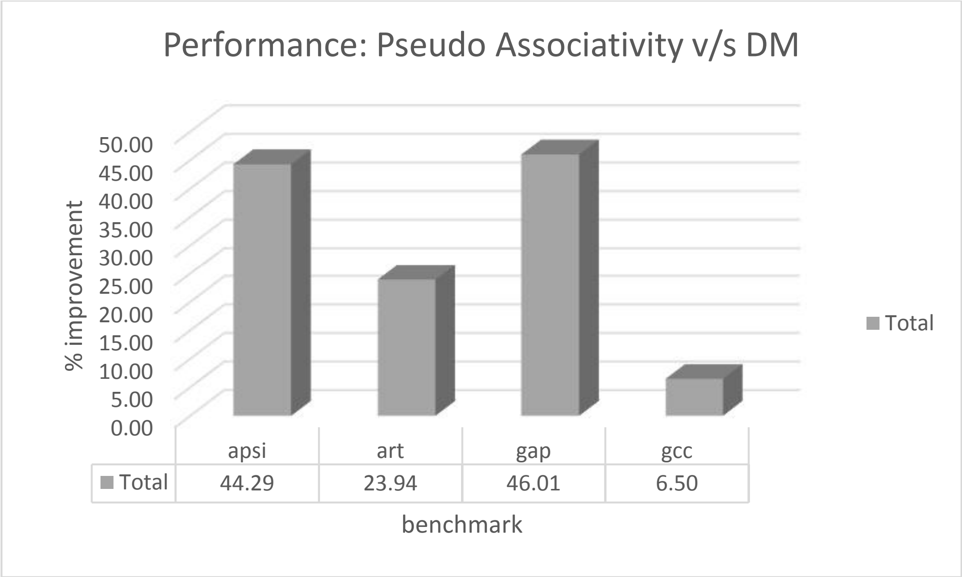
Part B

This implementation resolves conflicts encountered in directly mapped cache by allowing alternate hashing functions, which results in significantly better use of cache *area*. Using simplescalar simulation, we demonstrate that its miss rate is much better than that of a directly mapped cache.

Whenever there is a conflicting address, instead of fetching the new block from the lower level memory, a different hashing function is dynamically applied in order to place or locate the data in a different set. This is new set is obtained by flipping the MSB bit if of the index.

Number	BenchMark	Cache_Size (kB)	Sets	Block_Size	Assoc	MISS RATE: FALSE	MISS RATE: Pseudo	Improvement %
1	apsi	4	128	32	1	0.1935	0.1078	44.29
2	art	4	128	32	1	0.4462	0.3394	23.94
3	gcc	4	128	32	1	0.0815	0.0762	6.50
4	gap	4	128	32	1	0.0978	0.0528	46.01





Files Included along with this report:

- Log files of simplescalar
- Excel sheet with all recorded values and pivot table graphs
- cache.c cache.h sim-cache.c and sim-outorder.c files with Pseudo-Column Associativity algorithm implementation