NC STATE UNIVERSITY

Dept. of Electrical and Computer Engineering

ECE 563 : Fall 2019

Project#1 : Flexible Cache Simulator, Memory Hierarchy Design

By

VISHNU SURESH MENON

NCSU ID : 200317998

**Flexible Cache Simulator Project Report**

**Section 1: Effect of parameters on overall cache performance and its noteworthy trends**

1. L1 Cache size vs. miss rate (For different associativity) [Without L2]

Configuration: Block size = 16B, No L2 Cache

|  |  |
| --- | --- |
| Benchmark : gcc\_trace.txt | |
| Associativity = 1 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1922 |
| 12 | 0.1102 |
| 13 | 0.0846 |
| 16 | 0.0546 |
| Associativity = 2 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.16 |
| 12 | 0.0875 |
| 13 | 0.0638 |
| 16 | 0.0485 |
| Associativity = 4 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1473 |
| 12 | 0.0755 |
| 13 | 0.0595 |
| 16 | 0.0472 |
| Associativity = 8 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1401 |
| 12 | 0.0728 |
| 13 | 0.0587 |
| 16 | 0.0472 |
| Associativity = Full | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1417 |
| 12 | 0.0702 |
| 13 | 0.0630 |
| 16 | 0.0471 |

|  |  |
| --- | --- |
| Benchmark : perl\_trace.txt | |
| Associativity = 1 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.2599 |
| 12 | 0.1085 |
| 13 | 0.0672 |
| 16 | 0.0306 |
| Associativity = 2 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.2236 |
| 12 | 0.0771 |
| 13 | 0.0398 |
| 16 | 0.0262 |
| Associativity = 4 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.2236 |
| 12 | 0.0505 |
| 13 | 0.0337 |
| 16 | 0.0261 |
| Associativity = 8 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.2004 |
| 12 | 0.0458 |
| 13 | 0.0321 |
| 16 | 0.0261 |
| Associativity = Full | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1948 |
| 12 | 0.0433 |
| 13 | 0.0313 |
| 16 | 0.0261 |

|  |  |
| --- | --- |
| Benchmark : go\_trace.txt | |
| Associativity = 1 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.2623 |
| 12 | 0.1128 |
| 13 | 0.1028 |
| 16 | 0.0880 |
| Associativity = 2 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1317 |
| 12 | 0.0994 |
| 13 | 0.0988 |
| 16 | 0.0810 |
| Associativity = 4 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1046 |
| 12 | 0.0986 |
| 13 | 0.0984 |
| 16 | 0.0819 |
| Associativity = 8 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1016 |
| 12 | 0.0985 |
| 13 | 0.0984 |
| 16 | 0.0803 |
| Associativity = Full | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.0993 |
| 12 | 0.0984 |
| 13 | 0.0984 |
| 16 | 0.0824 |

|  |  |
| --- | --- |
| Benchmark : vortex\_trace.txt | |
| Associativity = 1 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1431 |
| 12 | 0.0654 |
| 13 |  |
| 16 | 0.0240 |
| Associativity = 2 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.1230 |
| 12 | 0.0436 |
| 13 | 0.0311 |
| 16 | 0.0222 |
| Associativity = 4 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.0992 |
| 12 | 0.0357 |
| 13 | 0.0278 |
| 16 | 0.0221 |
| Associativity = 8 | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.0931 |
| 12 | 0.0339 |
| 13 | 0.0268 |
| 16 | 0.0221 |
| Associativity = Full | |
| Log2(L1 Cache size) | Miss rate |
| 10 | 0.0675 |
| 12 | 0.0318 |
| 13 | 0.0261 |
| 16 | 0.0221 |

Trends : From the above tables and graphs its observed that for a given set associativity miss rate decreases exponentially as the size of the L1 cache increases. This observation is noticed among all the above-mentioned benchmarks. Also, it is noticed that as the associativity is increased the miss rate reduced significantly for all the benchmarks tested which is a result of the reduction in conflict miss rates.

1. Associativity vs. miss rate

Configuration: Block size = 16B, L1 Cache size = 1024B, No L2 Cache

|  |  |
| --- | --- |
| Benchmark : gcc\_trace.txt | |
| Associativity | Miss Rate |
| 2 | 0.16 |
| 4 | 0.1473 |
| 8 | 0.1401 |
| Benchmark : perl\_trace.txt | |
| Associativity | Miss Rate |
| 2 | 0.2236 |
| 4 | 0.2236 |
| 8 | 0.2004 |

|  |  |
| --- | --- |
| Benchmark : go\_trace.txt | |
| Associativity | Miss Rate |
| 2 | 0.1317 |
| 4 | 0.1046 |
| 8 | 0.1016 |
| Benchmark : vortex\_trace.txt | |
| Associativity | Miss Rate |
| 2 | 0.123 |
| 4 | 0.0992 |
| 8 | 0.0931 |

Trends: As the set-associativity is varied for a fixed L1 cache size configuration the trend observed is the reduction in the miss rate of L1 cache. The same trend is observed for all the benchmark trace and it is mainly due to the significant decrease in the conflict miss rates. As the set associativity is increased the degree of freedom for a block to be placed in a given set will rise and hence the chance for a conflict miss reduces resulting in lower miss rate.

1. L2 Cache size vs. miss rate (keep L1 size constant)

Configuration : Block size = 16B, L1 Associativity = 2, L2 Associativity = 1, N=1 and P=1

|  |  |  |  |
| --- | --- | --- | --- |
| Benchmark : gcc\_trace.txt | | | |
| Log2(L1 Cache size) = 10 | | Log2(L1 Cache size) = 12 | |
| Log2(L2 Cache size) | Miss rate | Log2(L2 Cache size) | Miss rate |
| 13 | 0.3975 | 13 | 0.703 |
| 14 | 0.3283 | 14 | 0.5982 |
| 15 | 0.3095 | 15 | 0.5653 |
| 16 | 0.303 | 16 | 0.5538 |
| Benchmark : perl\_trace.txt | | | |
| Log2(L1 Cache size) = 10 | | Log2(L1 Cache size) = 12 | |
| Log2(L2 Cache size) | Miss rate | Log2(L2 Cache size) | Miss rate |
| 13 | 0.1791 | 13 | 0.5012 |
| 14 | 0.1329 | 14 | 0.3806 |
| 15 | 0.1245 | 15 | 0.3602 |
| 16 | 0.1171 | 16 | 0.3396 |
| Benchmark : go\_trace.txt | | | |
| Log2(L1 Cache size) = 10 | | Log2(L1 Cache size) = 12 | |
| Log2(L2 Cache size) | Miss rate | Log2(L2 Cache size) | Miss rate |
| 13 | 0.753 | 13 | 0.9941 |
| 14 | 0.7482 | 14 | 0.9907 |
| 15 | 0.7285 | 15 | 0.9644 |
| 16 | 0.6142 | 16 | 0.8103 |
| Benchmark : vortex\_trace.txt | | | |
| Log2(L1 Cache size) = 10 | | Log2(L1 Cache size) = 12 | |
| Log2(L2 Cache size) | Miss rate | Log2(L2 Cache size) | Miss rate |
| 13 | 0.2505 | 13 | 0.692 |
| 14 | 0.2007 | 14 | 0.5664 |
| 15 | 0.1861 | 15 | 0.5258 |
| 16 | 0.1805 | 16 | 0.51 |

Trends: The L2 miss rate depends on L2 size and associativity. The above tables and graphs speculate that the miss rate reduces and remains stable to almost constant value when the L2 cache size is increased linearly keeping L1 cache size constant. The probability of capacity misses reduces as the L2 cache size is improved and thus it contributes to lessen the miss rate to an extent. Also, its noticed from the above experiment that the miss rate is low for a lower L1 cache size keeping the L2 cache size constant.

1. N address tags vs miss rate (Keep L1 constant, P constant)

Configuration: Block size = 32B, L1 Associativity = 2, L2 Associativity = 1, L2 Cache size = 32768B

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Benchmark : gcc\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 P = 2 | | Log2(L1 Cache size) = 12 P = 4 | | Log2(L1 Cache size) = 13 P = 8 | |
| N | Miss rate | N | Miss rate | N | Miss rate |
| 1 | 0.2598 | 1 | 0.5294 | 1 | 0.7731 |
| 2 | 0.239 | 2 | 0.4744 | 2 | 0.7022 |
| 4 | 0.238 | 4 | 0.47 | 4 | 0.6939 |
| Benchmark : perl\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 P = 2 | | Log2(L1 Cache size) = 12 P = 4 | | Log2(L1 Cache size) = 13 P = 8 | |
| N | Miss rate | N | Miss rate | N | Miss rate |
| 1 | 0.1452 | 1 | 0.3246 | 1 | 0.6445 |
| 2 | 0.1365 | 2 | 0.2934 | 2 | 0.5441 |
| 4 | 0.1336 | 4 | 0.284 | 4 | 0.5224 |
| Benchmark : go\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 P = 2 | | Log2(L1 Cache size) = 12 P = 4 | | Log2(L1 Cache size) = 13 P = 8 | |
| N | Miss rate | N | Miss rate | N | Miss rate |
| 1 | 0.5071 | 1 | 0.9496 | 1 | 0.9757 |
| 2 | 0.5057 | 2 | 0.9478 | 2 | 0.9737 |
| 4 | 0.4988 | 4 | 0.9346 | 4 | 0.959 |
| Benchmark : vortex\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 P = 2 | | Log2(L1 Cache size) = 12 P = 4 | | Log2(L1 Cache size) = 13 P = 8 | |
| N | Miss rate | N | Miss rate | N | Miss rate |
| 1 | 0.1345 | 1 | 0.4219 | 1 | 0.7025 |
| 2 | 0.1292 | 2 | 0.4007 | 2 | 0.6702 |
| 4 | 0.1264 | 4 | 0.3887 | 4 | 0.6488 |

Trends: Here as the number of address tags increases for a fixed number of cache line size the miss rate reduces initially and then gets stabilized to almost a constant value.

1. P address tags vs miss rate (Keep L1 constant, N constant)

Configuration: Block size = 32B, L1 Associativity = 2, L2 Associativity = 1, L2 Cache size = 32768B

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Benchmark : gcc\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 N = 2 | | Log2(L1 Cache size) = 12 N = 4 | | Log2(L1 Cache size) = 13 N = 8 | |
| P | Miss rate | P | Miss rate | P | Miss rate |
| 2 | 0.239 | 2 | 0.4594 | 2 | 0.6732 |
| 4 | 0.2662 | 4 | 0.47 | 4 | 0.676 |
| Benchmark : perl\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 N = 2 | | Log2(L1 Cache size) = 12 N = 4 | | Log2(L1 Cache size) = 13 N = 8 | |
| P | Miss rate | P | Miss rate | P | Miss rate |
| 2 | 0.1365 | 2 | 0.2823 | 2 | 0.5186 |
| 4 | 0.1411 | 4 | 0.284 | 4 | 0.5186 |
| Benchmark : go\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 N = 2 | | Log2(L1 Cache size) = 12 N = 4 | | Log2(L1 Cache size) = 13 N = 8 | |
| P | Miss rate | P | Miss rate | P | Miss rate |
| 2 | 0.5057 | 2 | 0.9346 | 2 | 0.9575 |
| 4 | 0.5057 | 4 | 0.9346 | 4 | 0.9575 |
| Benchmark : vortex\_trace.txt | | | | | |
| Log2(L1 Cache size) = 10 N = 2 | | Log2(L1 Cache size) = 12 N = 4 | | Log2(L1 Cache size) = 13 N = 8 | |
| P | Miss rate | P | Miss rate | P | Miss rate |
| 2 | 0.1292 | 2 | 0.3755 | 2 | 0.6116 |
| 4 | 0.1387 | 4 | 0.3887 | 4 | 0.6246 |

**Section 2: Influence of Parameters on AAT**

|  |  |
| --- | --- |
| Associativity = 2 | |
| Log2(L1 Cache size) | AAT |
| 10 | 3.6054 |
| 12 | 2.11895 |
| 13 | 1.63229 |
| 16 | 1.3188 |
| Associativity = 4 | |
| Log2(L1 Cache size) | AAT |
| 10 | 3.3948 |
| 12 | 1.92357 |
| 13 | 1.59372 |
| 16 | 1.34301 |
| Associativity = 8 | |
| Log2(L1 Cache size) | AAT |
| 10 | 3.34664 |
| 12 | 1.96638 |
| 13 | 1.6785 |
| 16 | 1.4417 |

1. Influence of L1 size on AAT (Without L2)

Cache Configuration:

BLOCKSIZE : 16B

L1 SIZE : 1KB, 4KB, 8KB and 64KB

L1 ASSOC : 2, 4, 8

NO L2 CACHE

Benchmark : gcc\_trace

The above graphs and table illustrate the variation of cache performance with the change in the L1 cache size parameter. The trend reveals that bigger the cache size lower is the average access time since increasing the cache capacity reduces the probability of capacity misses which diminishes the miss rate. As we know that AAT depends upon the cache hit time, miss rate and miss penalty, the parameter miss rate is affected upon varying the cache size which in turn reduces the AAT.

|  |  |
| --- | --- |
| Associativity = 2 | |
| Log2(L1 Cache size) | AAT |
| 10 | 2.30116 |
| 12 | 1.969 |
| 13 | 1.6545 |
| 16 | 1.424 |
| Associativity = 4 | |
| Log2(L1 Cache size) | AAT |
| 10 | 2.309 |
| 12 | 2.012 |
| 13 | 1.68 |
| 16 | 1.4665 |
| Associativity = 8 | |
| Log2(L1 Cache size) | AAT |
| 10 | 2.380 |
| 12 | 2.0665 |
| 13 | 1.796 |
| 16 | 1.5672 |

1. Influence of L1 size on AAT (With L2)

Cache Configuration:

BLOCKSIZE : 16B

L1 SIZE : 1KB, 4KB, 8KB and 64KB

L1 ASSOC : 2, 4, 8

L2 SIZE : 4KB

L2 ASSOC : 4

Benchmark : gcc\_trace

This trend is similar as mentioned above with the only difference that L2 cache is included along with L1 cache. But as we compare this trend with the above trend(without L2), the AAT values are less for this configuration compared to the above one without the presence of L2. This reveals the fact that including a one more level of cache memory reduces the average access time which reinforce the fact that multilevel caches improve the cache performance which is due to the reduction in miss penalty. Therefore, adding another level of cache between the L1 cache and memory helps to reduce the widening gap between processor and memory and also improves the speed of overall cache to cope up with the speed of processor speed.

1. Influence of Blocksize on AAT

Cache Configuration:

BLOCKSIZE : 16B, 32B, 64B, 128B

|  |  |
| --- | --- |
| Benchmark : gcc\_trace.txt | |
| Log2(L1 SIZE) = 10 | |
| Log2(BLOCKSIZE) | AAT |
| 4 | 3.39974 |
| 5 | 3.40158 |
| 6 | 3.93946 |
| 7 | 5.442 |
| Log2(L1 SIZE) = 12 | |
| Log2(BLOCKSIZE)) | AAT |
| 4 | 1.9431 |
| 5 | 1.67785 |
| 6 | 1.83111 |
| 7 | 2.56225 |
| Log2(L1 SIZE) = 13 | |
| Log2(BLOCKSIZE) | AAT |
| 4 | 1.6327 |
| 5 | 1.33093 |
| 6 | 1.33804 |
| 7 | 1.7494 |
| Log2(L1 SIZE) = 16 | |
| Log2(BLOCKSIZE) | AAT |
| 4 | 1.6555 |
| 5 | 1.25745 |
| 6 | 1.08502 |
| 7 | 1.07928 |

L1 SIZE : 1KB, 4KB, 8KB and 64KB

L1 ASSOC : 4

NO L2 CACHE

Benchmark : gcc\_trace

As per the above data mentioned in the table and graph it is depicted that the variation in block size influences to reduce the miss rate and resulting in the improvement of average access time. Larger block size means lesser the chances for compulsory misses as the large blocks take advantage of the spatial locality ie; increasing the size of the memory block helps to accommodate maximum bytes of data helping to capture most number of memory references or requests from the processor. But there exists a tradeoff in varying the block size after an extent since it increases the miss penalty. The reason behind the tradeoff is due to the fact that larger blocks means less memory blocks within a cache capacity which introduces higher probability of conflict misses and sometimes capacity misses too if the cache size is small. Thus, increasing miss penalty is outweighed by the reduction in miss rate thereby increasing the AAT after a while as illustrated in the graph.

|  |  |
| --- | --- |
| Log2(L1 Cache size) = 10 | |
| Log2(L2 Cache size) | AAT |
| 13 | 8.62109 |
| 14 | 6.90451 |
| 15 | 6.76925 |
| 16 | 6..8725 |
| Log2(L1 Cache size) = 12 | |
| Log2(L2 Cache size) | AAT |
| 13 | 16.5975 |
| 14 | 12.4835 |
| 15 | 12.106 |
| 16 | 12.1466 |
| Log2(L1 Cache size) = 13 | |
| Log2(L2 Cache size) | AAT |
| 13 | 19.0768 |
| 14 | 16.3963 |
| 15 | 15.886 |
| 16 | 15.884 |
| Log2(L1 Cache size) = 14 | |
| Log2(L2 Cache size) | AAT |
| 14 | 22.3302 |
| 15 | 22.3639 |
| 16 | 22.3125 |
| 17 | 22.2065 |

1. Influence of L2 on AAT

Cache Configuration:

BLOCKSIZE : 32B

L1 SIZE : 1KB, 4KB, 8KB and 16KB

L1 ASSOC : 4

L2 SIZE : 8KB, 16KB, 32KB, 64KB

L2 ASSOC : 8

Benchmark : gcc\_trace

This trend is also same as the variation of the L1 size versus the AAT. As the L2 cache size increases the reduction in capacity misses reduces the miss rate and in turn the average access time.

**Section 3 : Best Memory Hierarchy Configuration**

So as discussed in the above sections, performed the analysis of miss rate and average access time by variation of different parameters like the L1 cache size, L1 associativity, L2 cache size, L2 associativity, the L2 address tags and L2 data blocks of the sectored cache etc.

The general trend from the various analysis performed above it is observed that the miss rate depends upon the cache size and set associativity of the cache structure. As the L1 cache size is increased keeping the set associativity constant the miss rate reduces which is a result of the improvement of capacity misses and whereas keeping the L1 cache size constant and varying the associativity also shows an improvement in miss rate which is a result of the reduction in conflict misses.

Thus, to get a combined effect of both the capacity misses and conflict misses its better to have a configuration with higher associativity and maximum cache capacity.

Cache configuration

BLOCKSIZE : 32B

L1 SIZE : 8KB

L1 ASSOC : 4

L2 SIZE : 0

Benchmark : gcc\_trace

From the influence of blocksize versus AAT it is observed that the blocksize of 32B with L1 size of 8KB gives the least AAT with the constant associativity of 4. This is based on the above analysis and trends which is observed within a set of configurations evaluated so far. Hence the reason for choosing the above configuration.

This similar configuration is applicable for all other benchmark traces as all other shows similar trend.

**Section 4 : Different Benchmarks**

From the above analysis it is imperative that keeping the L1 and L2 cache set associative and larger cache size configurations are better for benchmark traces gcc, perl and vortex based on the above explanation mentioned in Section3.

**Section 5: Advantages of the Decoupled Sectored Cache**

The idea of sectored cache originated from the concept of increasing the block size in order to reduce the CPU overhead of miss penalty at the same time maintaining reduced tag store size. The goal of the decoupled sectored cache is to improve the hit rate of sectored cache as the address tag location associated with a cache line location is dynamically chosen at fetch time among several possible locations. The decoupled sectored cache provides a good hit ratio of a non-sectored cache with a low hardware cost. There exists no static binding between one tag entry versus one data block i.e.; there exist a freedom in decoupled sectored one.

The reason behind implementing decoupled sector cache in L2 rather than L1 is considering the factor of memory bandwidth. The memory bandwidth from L2 to the next level of memory hierarchy is comparatively less than from L1 to L2 which means the miss penalty of L2 is higher than L1 resulting in higher average access time and lower performance. So, if the L2 is implemented in decoupled sectored cache structure then multiple blocks which are fetched dynamically by one address tag are transmitted simultaneously thereby increasing the memory bandwidth.

**THE END**