NC State University

Department of Electrical and Computer Engineering

ECE 463/563: Fall 2018 (Rotenberg)

Project #1: Cache Design, Memory Hierarchy Design

by

<< SALONI SHAMBHUWANI >>

NCSU Honor Pledge: "I have neither given nor received unauthorized aid on this test or assignment."

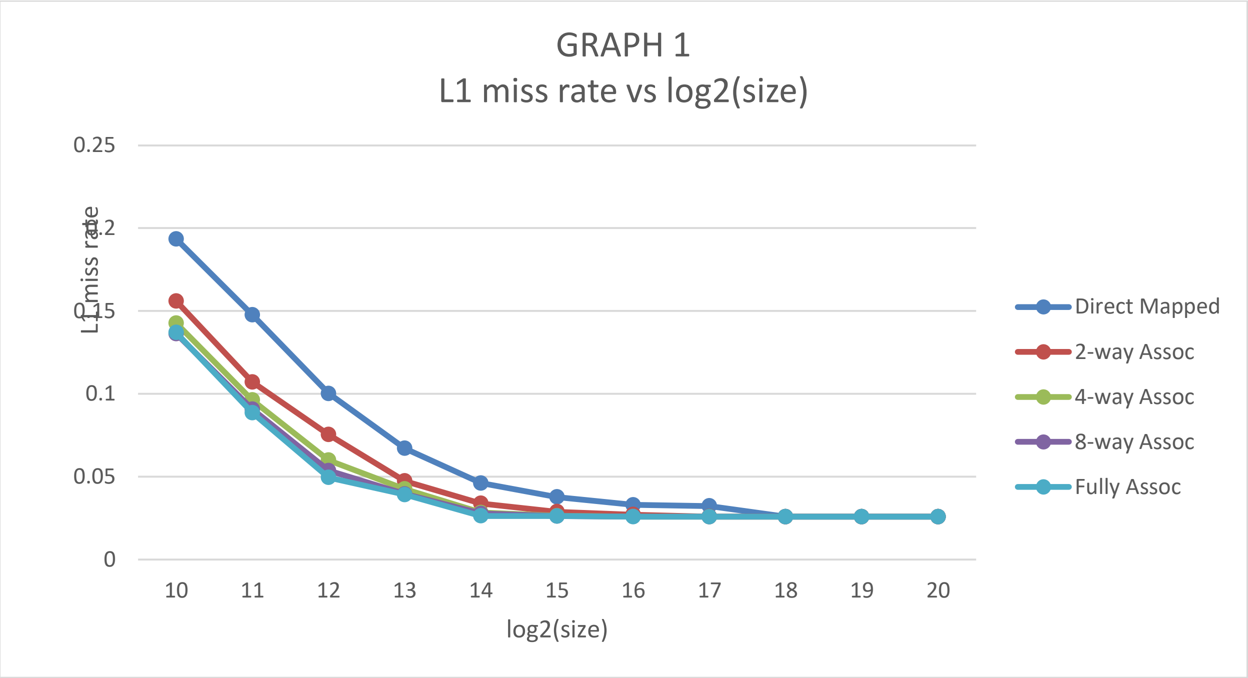
Student’s electronic signature: Saloni Shambhuwani
(sign by typing your name)

Course number: ___563________
(463 or 563 ?)

Graph #1: L1 Miss Rate vs. $\log_2$(L1 SIZE), without VC and L2

Cache Configuration:
L1 Cache Size = 1KB, 2KB, 4KB, …, 1MB
L1 Cache Associativity: Direct-mapped, 2-way SA, 4-way SA, 8-way SA, Fully associative
No VC, No L2 Cache

Plot:



Details:

1. Discuss trends in the graph. For a given associativity, how does increasing cache size affect miss rate? For a given cache size, what is the effect of increasing associativity?

   Answer:
   For a given associativity, increasing cache size tends to reduce miss rate. Increasing cache size reduces the number of capacity and conflict misses but not the compulsory misses. After a certain point, say here, once the L1 cache size is 2^18 bytes, increasing the size of cache beyond that does not affect the miss rate any more(miss rate does not reduce further) since, there are only compulsory misses left.

   For a given cache size, increasing associativity reduces the miss rate. By increasing associativity, we conflict misses reduce. Thus, the miss rate reduces as we move from direct mapped cache to a fully associative cache.

2. Estimate the *compulsory miss rate* from the graph

   Answer:

For a given associativity, increasing cache size tends to reduce miss rate. As we go on increasing cache size, after a point, we get rid of capacity misses. As we go on increasing associativity the conflict miss reduce. Thus increasing cache size and associativity will eliminate all the conflict and capacity misses leaving only compulsory misses.
From the graph compulsory miss rate can be estimated to be 0.0258.

Compulsory miss rates cannot be reduced by standard techniques, except software or hardware prefetching.

3. For each associativity, estimate the *conflict miss rate* from the graph.

Answer:
To get estimate of conflict miss rate, we compare each associativity (cache under test) with the full associative cache. A fully associative cache has no conflict misses. Thus the average difference in miss rate between an associative cache and the fully associative cache of same size will give us the estimate of conflict miss rate for that associativity.

Below is the estimate of conflict miss rate:

Size/Assoc	1KB	2KB	4KB	8KB	16KB	32KB	64KB	128KB	256KB	512KB	1MB
Direct Mapped	0.056	0.059	0.051	0.027	0.19	0.11	0.007	0.006	0.00	0.00	0.00
2-way associativity	0.019	0.018	0.026	0.008	0.007	0.003	0.001	0.0001	0.00	0.00	0.00
4-way associativity	0.006	0.007	0.010	0.003	0.002	0.0002	0.0001	0.00	0.00	0.00	0.00
8-way associativity		0.002	0.004	0.000	0.001	0.00	0.00	0.00	0.00	0.00	0.00

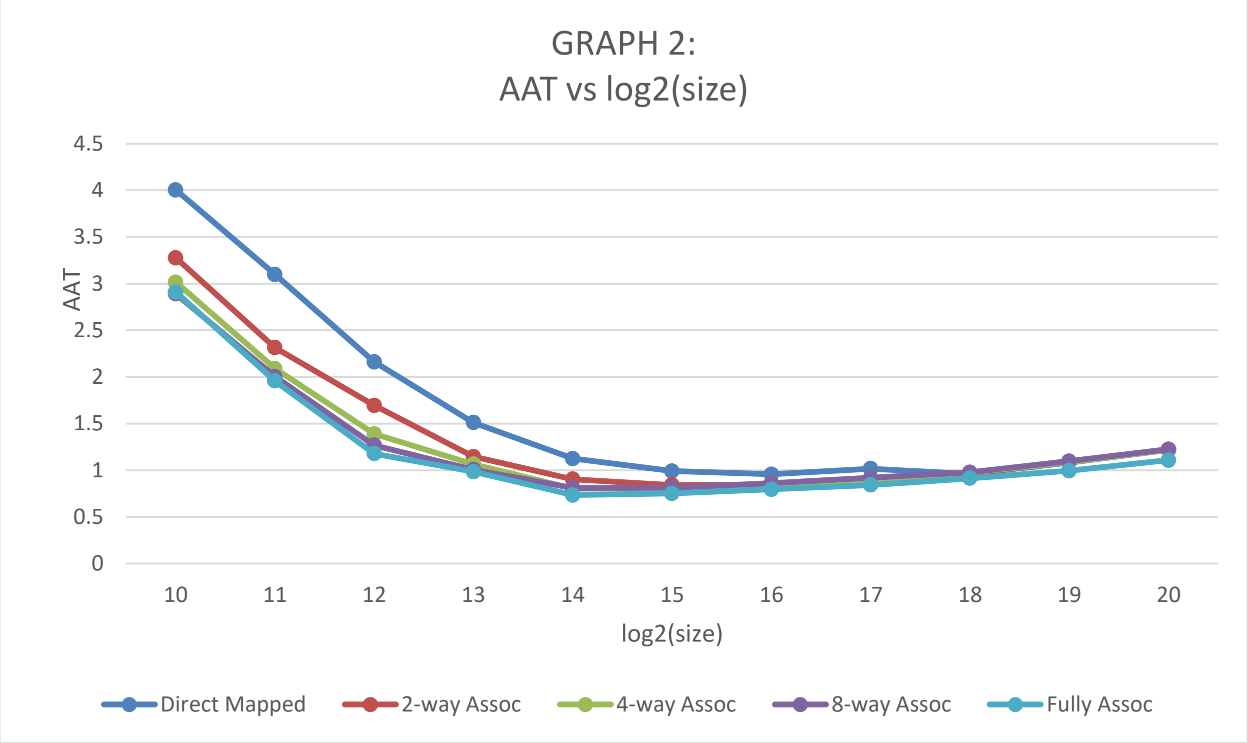
A fully-associative cache does not suffer from conflict misses; it suffers only compulsory and capacity misses.

Graph #2: AAT vs. log2(L1 SIZE), without VC or L2

Cache Configuration:

Blocksize = 32
L1 Cache Size = 1KB, 2KB, 4KB, …, 1MB
L1 Cache Associativity: Direct-mapped, 2-way SA, 4-way SA, 8-way SA, Fully associative
No VC, No L2 Cache

Plot:



Details:

1. For a memory hierarchy with only an L1 cache and BLOCKSIZE = 32, which configuration yields the best (i.e., lowest) AAT?

Answer:

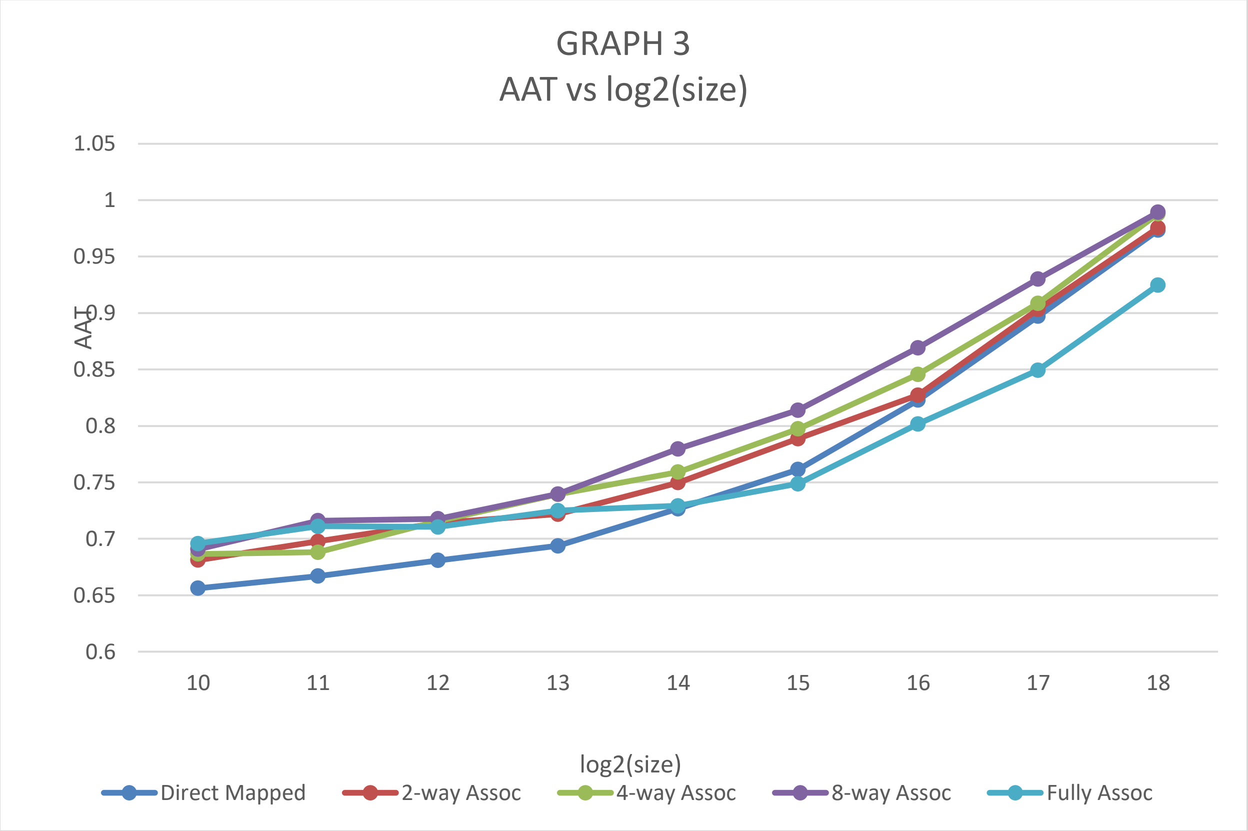
From the graph, Cache size of 16KB and fully associative mapping leads to best (i.e. the lowest) AAT. AAT depends upon the cache hit time, miss rate and the miss penalty. The miss penalty is the same for all the configurations. Hit time and the miss rate depend both on the associativity and the cache size. Hit time increases with the cache size, and the miss rate decreases with size. It is thus a trade-off between the size and associativity. Fully associative cache causes increase in AAT as cache size increases further.

With a cache of 16KB and full associativity, the trade-off gives the optimal value of the AAT.

Graph #3: AAT vs. log2(L1 SIZE), with L2 Cache included

Cache Configuration:
Blocksize = 32
L1 Cache Size = 1KB, 2KB, 4KB, …, 1MB
L1 Cache Associativity: Direct-mapped, 2-way SA, 4-way SA, 8-way SA, Fully associative
L2 Cache Size = 512KB
L2 Associativity: 8-way set associative
No VC

Plot:



Details:

1. With the L2 cache added to the system, which L1 cache configurations result in AATs close to the best AAT observed in GRAPH #2 (*e.g.*, within 5%)?

Answer:
The lowest AAT observed in Graph #2 is 0.737 ns. We can observe from the AAT values for the following configurations in Graph#3 which are less than 0.774ns are within 5% of the lowest AAT in Graph #2:
Yes -AAT is within 5% of the best AAT from Graph #2;
No -AAT is not within a 5% range.

Assoc/Cache Size	1KB	2KB	4KB	8KB	16KB	32KB	64KB	128KB	256KB
Direct-Mapped	Yes	Yes	Yes	Yes	Yes	No	No	No	No

2-Way SA	Yes	Yes	Yes	Yes	Yes	No	No	No	No
4-Way SA	Yes	Yes	Yes	Yes	Yes	No	No	No	No
8-Way SA	Yes	Yes	Yes	Yes	No	No	No	No	No
Fully-Associative	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No

2. With the L2 cache added to the system, which L1 cache configuration yields the best (*i.e.*, lowest) AAT? How much lower is this optimal AAT compared to the optimal AAT in GRAPH #2?

Answer:

The lowest AAT in the graph is 0.724 ns, for a direct-mapped L1 cache of 8KB size.
In Graph#2 which is without L2 cache, the lowest observed AAT is 0.737 ns.
Thus, the memory hierarchy with L2 included has an AAT which is 0.0128 ns lower, which is 1.76% lower than the one without L2.
Thus, the presence of L2 reduces the miss penalty in the hierarchy leading to a decrease in AAT.

3. Compare the *total area* required for the optimal-AAT configurations with L2 cache (GRAPH #3) versus without L2 cache (GRAPH #2).

Answer:

Graph#2

Total area required for the optimal-AAT configurations without L2 cache is 0.063446019 mm^2.

Graph#3

Total area required for the optimal-AAT configurations with L2 cache is
Area of L1 Cache + Area of L2 Cache
= 0.053293238 + 2.640142073
= 2.693435311 mm^2.

The increase in area when L2 cache is included in the memory hierarchy = 2.693435311 - 0.063446019 = 2.629989292 mm^2

Percentage increase = 4145.24%.

Thus, for a small improvement in the optimal AAT, there is huge increase in the total memory hierarchy area.

1. Compare the *total area* required for the optimal-AAT configurations with L2 cache (GRAPH #3) versus without L2 cache (GRAPH #2).

Answer:

Graph#2

Total area required for the optimal-AAT configurations without L2 cache is 0.063446019 mm^2.

Graph#3

Total area required for the optimal-AAT configurations with L2 cache is

Area of L1 Cache + Area of L2 Cache
= 0.053293238 + 2.640142073
= 2.693435311 mm^2.

The increase in area when L2 cache is included in the memory hierarchy = 2.693435311 - 0.063446019 = 2.629989292 mm^2

Percentage increase = 4145.24%.

Thus, for a small improvement in the optimal AAT, there is huge increase in the total memory hierarchy area.

GRAPH #4: L1 miss rate versus $\log_2$(Blocksize)

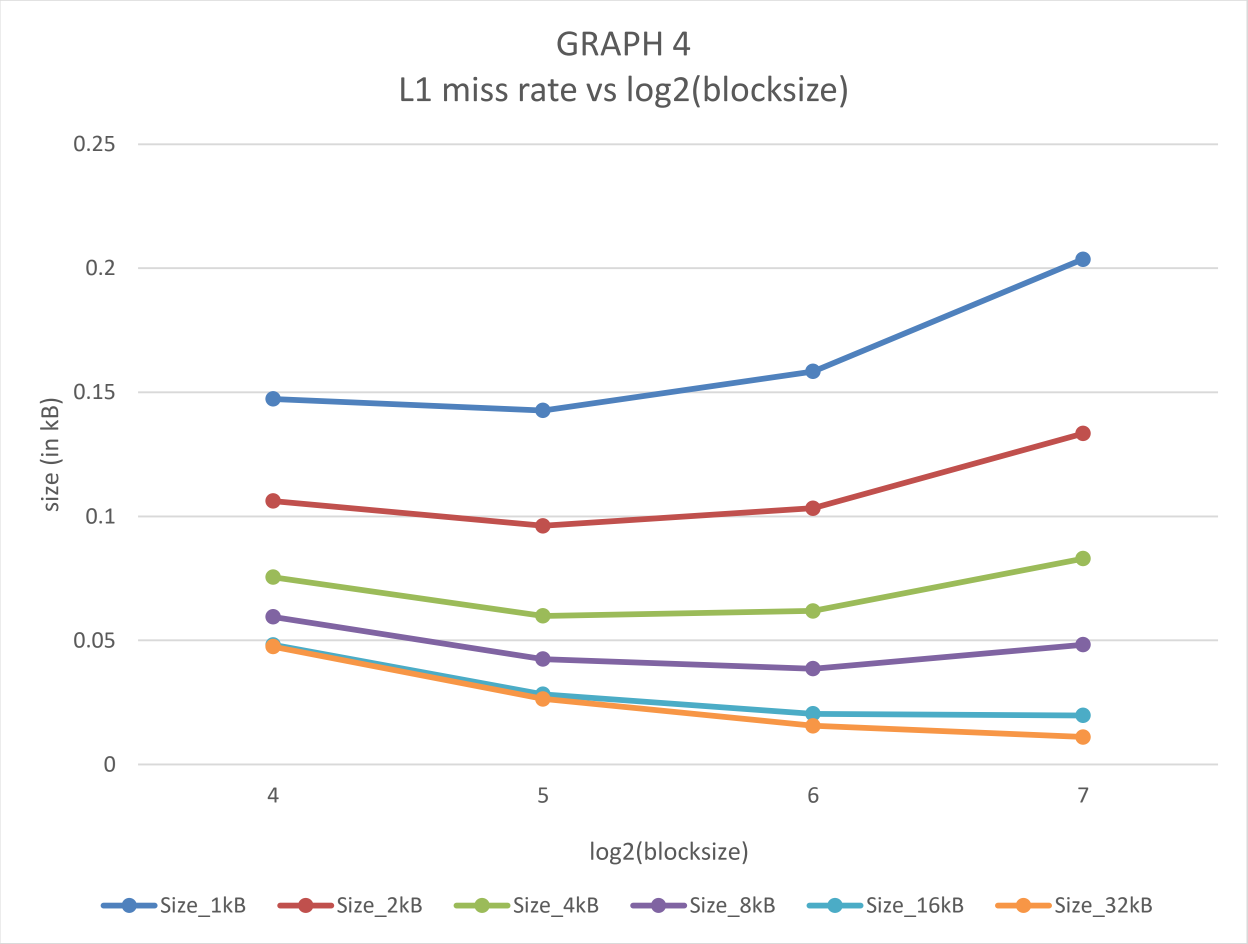
L1 cache:

a. Size: 1KB, 2KB, …. , 32KB (in powers of two)
b. Assoc: 4
c. Blocksize = 16, 32, 64 and 128

Victim cache: None

L2 cache: None

Plot:



Details:

1. Discuss trends in the graph. Do smaller caches prefer smaller or larger block sizes? Do larger caches prefer smaller or larger block sizes? Why? As block size is increased from 16 to 128, is the tradeoff between *exploiting more spatial locality* versus *increasing cache pollution* evident in the graph, and does the balance between these two factors shift with different cache sizes?

Answer:

From the graphs it can be observed that the smaller cache prefer small size. The cache block-size affects the spatial locality present in the cache. If the address trace has good spatial locality, a larger block-size will help to reduce the miss rate while a large block-size leads to fewer unique blocks being present in the cache. A larger block-size will cause lesser variety of blocks in the cache leading to an increase in miss rate.

Larger caches prefer larger block sizes to an extent since it allows to exploit more spatial locality. After certain point, increasing blocksize does not improve miss rate rather large blocksizes cause a phenomenon called cache pollution where in unnecessary blocks occupy the cache leading to higher miss rate.

The effect on miss rate of an increase in blocksize depends upon the trade-off between the positive impact of the increase in spatial locality and the negative impact of possible cache pollution because of bringing in lesser variety of memory blocks. The overall effect of varying blocksize depends upon the total cache size. For a smaller cache size, the cache pollution effect begins to dominate, whereas for a larger cache size the spatial locality provides a greater positive impact than the negative effect of cache pollution.

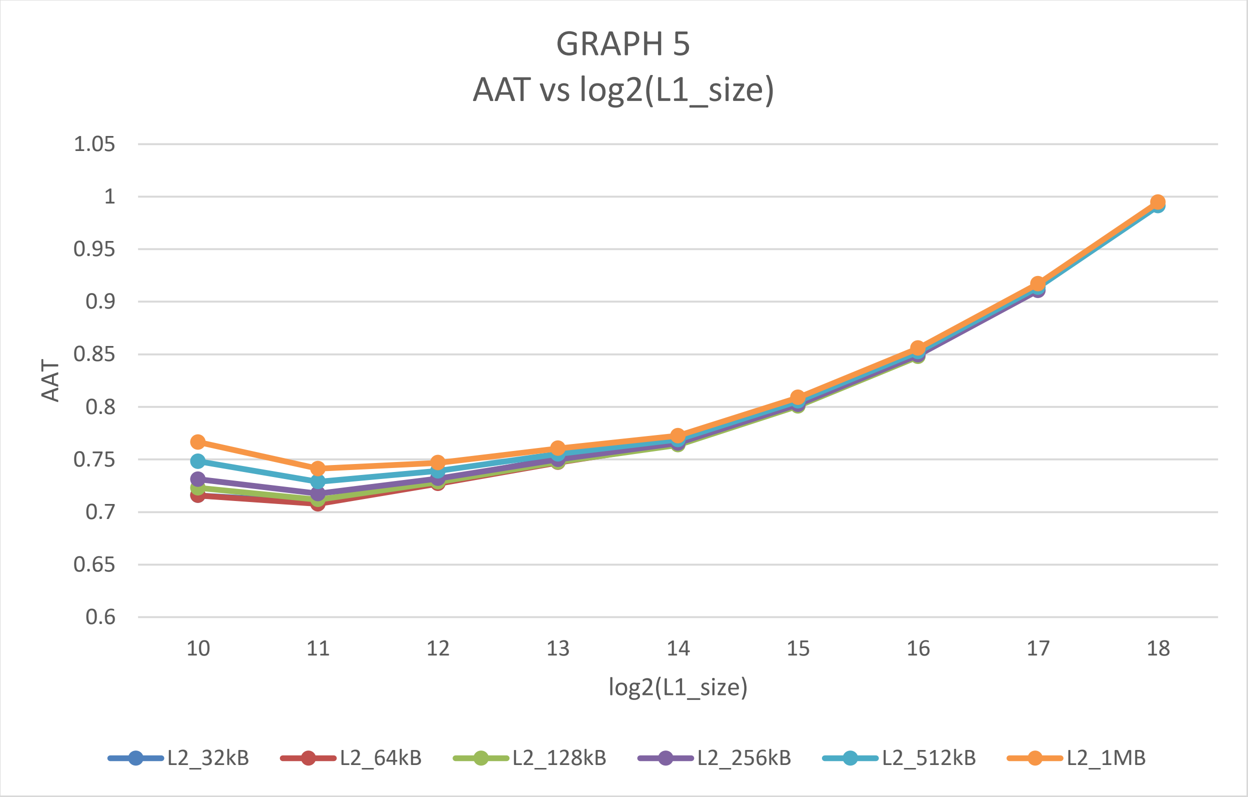
Yes, as the blocksize is increased from 16 to 128, the tradeoff between exploiting spatial locality versus cache pollution is evident from the graph. Yes, the balance shifts with different cache sizes. Smaller caches get polluted sooner than larger caches with increase of the block size.

GRAPH #5: AAT versus $\log_2$(L1 size)

Configuration:

1. L1 cache:
   a. Size: 1KB, 2KB, …. , 256KB (in powers of two)
   b. Assoc: 4
   c. Blocksize = 32
2. Victim cache: None
3. L2 cache:
   a. Size: 32KB, 64KB, 128KB, 256KB, 512 KB, 1MB
   b. Assoc: 8
   c. Blocksize = 32

Plot:



Details:

1.Which memory hierarchy configuration yields the best (*i.e.*, lowest) AAT?

Answer:
L1 : 2KB cache with 4-way set associative
L2 : 64KB 8-way set associative
The lowest AAT value is found to be **0.710 ns**. The AAT in this hierarchy involves a complex trade-off between the L1 hit time (which depends on L1 cache size), the L1 miss rate (depends onL1 size and associativity), the L2 hit time (depends on L2 size), and the L2 miss rate (again depends on L2 size and associativity

2. Which memory hierarchy configuration has the smallest total area, that yields an AAT within 5% of the best AAT?

Answer:
L1 : 1KB cache with 4-way set associative
L2 : 32KB cache with 8-way set associative

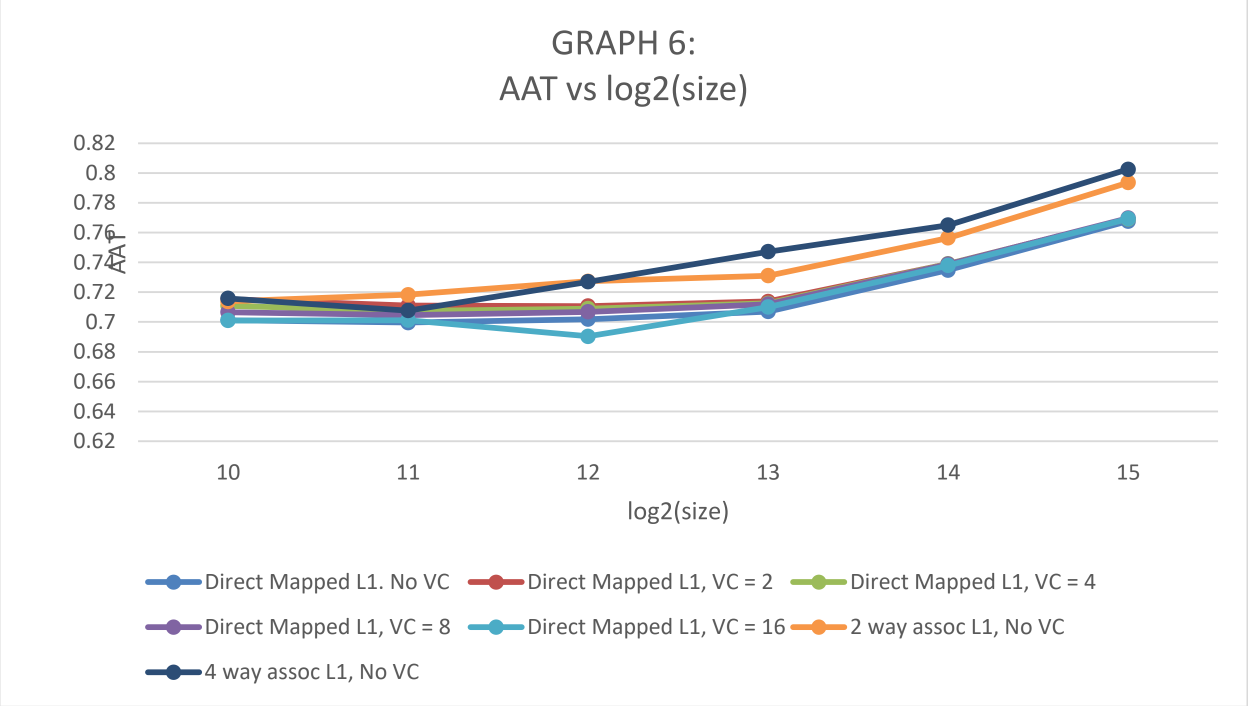
The area for this configuration is 0.257 mm^2. Also, the AAT is 0.718 ns, which is within 1.1% of the lowest AAT. The trade-off here is between larger area needed for a bigger cache and the decrease in miss rate observed on account of the larger size.

GRAPH #6: AAT versus log$_2$(size)

Configuration:

1. L1 cache:
   a. Size: 1KB, 2KB, …. , 32KB (in powers of two)
   b. Assoc: Vaies as per situation
   c. Blocksize = 32
2. Victim cache: Entry varies as per case
3. L2 cache:
   a. Size: 64KB
   b. Assoc: 8
   c. Blocksize = 32

Plot:



Details:

1. Discuss trends in the graph. Does adding a Victim Cache to a direct-mapped L1 cache yield performance comparable to a 2-way set-associative L1 cache of the same size?
…for which L1 cache sizes? …for how many Victim Cache entries?

Answer:

Ideally a victim cache adds associativity to a direct mapped cache thus reducing the conflict misses and hence the AAT. But the general trend in the graph is that adding Victim cache to direct mapped L1 does not yield performance comparable to 2-way set associative L1 cache of same size. This could possibly be due to nature of trace file where in addresses do not conflict and hence adding the victim cache provides no improvement in AAT.
The performance of a **1KB 2-way set associative L1 cache** is thus comparable to a 1KB L1 cache with a 2-entry or a 4-entry Victim Cache. As the size of the L1 cache increases, the Victim Cache configuration performs better than the set associative cache.

2. Which memory hierarchy configuration yields the best (i.e., lowest) AAT?

Answer:
The best value of AAT found is 0.7022ns with following memory hierarchy with blocksize of 32 bytes L1 : 2KB direct mapped
L2 : 64KB 8-way Set associative
No Victim

Although the miss rate for this configuration is not the least among the various configurations, the overall AAT is lower because of no additional delay on account of VC access.

3. Which memory hierarchy configuration has the smallest total area, that yields an AAT within 5% of the best AAT?

Answer:

The smallest AAT is found to be 0.7038 for following memory hierarchy with 32 Bytes blocksize: L1: 1KB direct mapped
L2: 64KB 8-way set associative.

Looking at total cache areas for configuration with an AAT within 5%, *i.e.* AAT upto 0.737 ns, the smallest total area is found to be for a 1KB 2-way set associative L1 cache with no Victim Cache, with 64KB 8-way set associative L2 Cache. The area is found to be 0.369 mm^2. The AAT for this configuration is 0.716 ns, which is within 2% of the lowest AAT.