Cache Performance Measures

Hit rate : fraction found in the cache

* So high that we usually talk about Miss rate = 1 - Hit Rate

Hit time : time to access the cache

Miss penalty : time to replace a block from lower level, including time to replace in CPU

* access time : time to access lower level
* transfer time : time to transfer block

Miss rate : fraction of time that we can’t find the data in cache

Average memory-access time (AMAT)

= Hit time + Miss rate x Miss penalty (ns or clocks)

Measuring and Analyzing Cache Performance

CPU time can be divided into 2 parts:

* CPU time = (CPU execution clock cycles + Memory-stall clock cycles) x Clock cycle time

Memory-stall clock cycles = Read-stall cycles + Write-stall cycles

Read-stall cycles = (Reads/Program) x Read miss rate x Read miss penalty

* Reads/Program = # of load instructions in a program
* Underlined part is # of misses

Write-stall cycles = ((Writes/Program) x Write miss rate x Write miss penalty) + Write buffer stall

* Write buffer stall if buffer is full we have to wait until there is emptiness so that we can put our data inside buffer
* Write buffer stall can be ignored using reasonable write buffer

Read and write stall cycles can be combined by using single miss rate and miss penalty:

* memory-stall clock cycles = (memory accesses / program) x miss rate x miss penalty
* memory-stall clock cycles = (Instructions/Program) x (Miss/Instruction) x Miss penalty

**Example: Cache Performance**

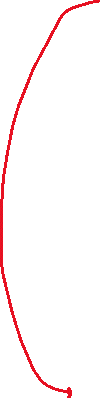
Assume that:

* Instruction miss rate %2
* Data miss rate %4
* CPI is 2 (without any memory stalls)
* Miss penalty 40 cycles
* %36 of instructions are load/store

Determine how much faster a machine would run with a perfect cache that never missed.

Text

Description automatically generated



2I cycles if the cache is perfect.

0.02I instructions will result in a miss. 🡪 For all these, I have to pay miss penalty

* 0.02I x Miss penalty = 0.8I cycles

# of instructions responsible for data miss:

* 0.36I 🡪 load – store instructions
* 4% of them result in a miss
* 0.36I x 0.04 = # of instructions result in a data miss
  + For each of them, I pay 40 cycles price (miss penalty)
  + 0.36I x 0.04 x 40 = 0.576I cycles

New execution time with stalls:

* 2I + 0.8I + 0.576I = 3.38I cycles

New perfect machine is 3.38I/2I = 1.69 times faster

**Cache Performance with Increased Clock Rate**

Suppose that clock rate of the machine used in the previous example is doubled but the memory speed, cache misses, and miss rate are same. How much faster the machine be with the faster clock?

Text

Description automatically generated with medium confidence

**Calculating AMAT**

If a direct mapped cache has a hit rate of 95%, a hit time of 4 ns, and a miss penalty of 100 ns, what is the AMAT?



In the first question, we have focused on the performance of whole program.

AMAT is important for memory performance.

We always pay the hit time price, it is always there.

For the missed ones, we pay the miss penalty.

You have to decrease 9ns to improve your memory performance.

* Hit time can be decreased by
  + using smaller cache
  + using direct map
* Miss rate can be decreased by
  + using bigger cache
  + using full associative
    - there is a trade of between hit time and miss rate, you should find the best choice
* Miss penalty can be decreased by
  + putting additional caches (If you have 1 cache and 1 main memory, miss penalty is very big)

If replacing the cache with a 2-way set associative increases the hit rate to 97%, but increases the hit time to 5 ns, what is the new AMAT?



Split vs. Unified Cache

Unified cache (mixed cache): Data and instructions are stored together (von Neuman architecture)

Split cache: Data and instructions are stored separately (Harvard architecture)

Why do instructions caches have a lower miss ratio?

Table

Description automatically generated

*percents are misses*

AMAT also depends on your program. Your miss rate depends on your program too.

We can’t say best one is 128 KB because hit time is also important.

Improve performance by:

1. Reduce the miss rate,
2. Reduce the miss penalty, or
3. Reduce the time to hit in the cache.

Decreasing one quantity, may result in increasing other one. You have to be careful.

Reducing Misses

Classifying Misses: 3 Cs

* Compulsory—The first access to a block is not in the cache, so the block must be brought into the cache. These are also called cold start misses or first reference misses.  
  (Misses in infinite cache)
  + You can’t get rid of them
* Capacity—If the cache cannot contain all the blocks needed during execution of a program, capacity misses will occur due to blocks being discarded and later retrieved.   
  (Misses due to size of cache)
  + You don’t have that data because of the capacity
  + By increasing cache capacity, you can decrease the capacity misses
* Conflict—If the block-placement strategy is set associative or direct mapped, conflict misses (in addition to compulsory and capacity misses) will occur because a block can be discarded and later retrieved if too many blocks map to its set. These are also called collision misses or interference misses.  
  (Misses due to associative and size of cache)
  + Occurs because of small associativity. Aşağıdan gelen block, daha önce aldığım blockun yerine geçiyor. Yani adresler conflict ediyor.
  + By increasing associativity, you can decrease the conflict misses
  + By increasing cache capacity, you can decrease the conflict misses (mod aldığımız sayı büyüdüğü için conflictler azalır)

Increasing cache size:

* One way to decrease misses
  + reduces capacity and conflict misses
  + no effect on compulsory misses
* However a larger cache may increase the hit time
  + larger cache 🡪 larger access time
  + if cache is too large, can’t fit it on the same chip as processor

3Cs Absolute Miss Rate

Diagram

Description automatically generated

All misses are either from conflict or capacity misses.

0.01 misses are for compulsory misses, so increasing cache size will not be any good at some point.

Increasing associativity 🡪 increasing hit time

Amount of compulsory miss doesn’t change according to your architecture strategy. But you can decrease conflict misses by architecture strategy.

Increasing block size:

* Another way to reduce the miss rate is to increase the block size
  + take advantage of spatial locality
  + decreases compulsory misses
* However, larger blocks have disadvantages
  + may increase the miss penalty (need to get more data)
  + may increase hit time (need to read more data from cache and larger MUX)
    - each word in data section of the block enters the MUX
  + may increase miss rate, since conflict misses
    - you don’t increase the cache size but increase the block size, then:
      * # of blocks will be reduced, mod aldığımız sayı küçülür, conflict missler artar.
* Increasing the block size can help, but don’t overdo it

Block Size vs. Cache Measures

Increasing Block Size generally increases Miss Penalty and decreases Miss Rate

As the block size increases the AMAT starts to decrease, but eventually increases

Diagram

Description automatically generated



Chart, line chart

Description automatically generated

How much associativity?

Increased associativity decreases miss rate

* But with diminishing returns (corresponds to increasing hit time)

Simulation of a system with 64KB D-cache, 16-word blocks, SPEC2000

* 1-way: 10.3%
* 2-way: 8.6%
* 4-way: 8.3%
* 8-way: 8.1%

Increasing Associativity

Increasing associativity helps reduce conflict misses

2:1 Cache Rule:

* The miss rate of a direct mapped cache of size N is about equal to the miss rate of a 2-way set associative cache of size N/2
* For example, the miss rate of a 32 Kbyte direct mapped cache is about equal to the miss rate of a 16 Kbyte 2-way set associative cache

Disadvantages of higher associativity

* Need to do large number of comparisons
  + If it is a 8-way set associative, you need to look 8 different positions
* Need n-to-1 multiplexor for n-way set associative
* Could increase hit time

Using 2-way set associative strategy helps you increase hit rate or helps you decrease miss rate

Set Associative Cache Organization

Diagram, engineering drawing

Description automatically generated



Data will be also separated into words and choose between words with another MUX.

AMAT vs. Associativity

Table

Description automatically generated

Hit time becomes the problematic part of the AMAT when you increase the cache size.

Replacement Policy

I couldn’t find block in cache, I have to get that block from lower level, where should I put that block?

Direct mapped: no choice

Set associative

* Prefer non-valid entry, if there is one
* Otherwise, choose among entries in the set

Least-recently used (LRU)

* Choose the one unused for the longest time
  + Simple for 2-way, manageable for 4-way, too hard beyond that

Random

* Gives approximately the same performance as LRU for high associativity

Using a 2nd Level Cache

A second level (L2) cache reduces the miss penalty by providing a large cache between the first level (L1) cache and main memory

L2 Equations:

Graphical user interface, text, application

Description automatically generated

**Adding an L2 Cache**

If a direct mapped cache has a hit rate of 95%, a hit time of 4 ns, and a miss penalty of 100 ns, what is the AMAT?



Miss penalty is large because we don’t have L2 cache here. Lower level is directly main memory.

If an L2 cache is added with a hit time of 20 ns and a hit rate of 50%, what is the new AMAT?

Text

Description automatically generated

L1 cache is subset of L2 cache. L1’de olan her şey L2’de var. L2’nin hit rate’i nasıl daha düşük oluyor?

* L2 cache’in absolute hit rate’i %95’in altında olamaz çünkü L1’in bütün dataları L2’de zaten var.
* %50 demek L1’de bulamadıklarının %50’sini L2’de buluyormuş demek. Yani aslında L2’nin hit rate’i %97.5
* Önemli olan ben L2’ye gittiklerimin ne kadarında aradığımı buluyorum.

Multilevel Caches

Primary cache attached to CPU

* Small, but fast

Level-2 cache services misses from primary cache

* Larger, slower, but still faster than main memory

Main memory services L-2 cache misses

Some high-end systems include L-3 cache

Text

Description automatically generated

Text

Description automatically generated

Multilevel Cache Considerations

Primary cache

* Focus on minimal hit time

L-2 cache

* Focus on low miss rate to avoid main memory access
* Associativity yüksek tutulabilir ki miss rate düşük olsun
* Hit time has less overall impact

Results

* L-1 cache usually smaller than a single cache
* L-1 block size smaller than L-2 block size

**CACHE PERFORMANCE SUMMARY**

AMAT = Hit time + Miss rate x Miss penalty

Split vs. Unified Cache

3 C’s of misses

* compulsory
* capacity
* conflict

Methods for improving performance

* increase (change) cache size
* increase (change) block size
* increase (change) associativity
* add a 2nd level cache