Estratégias para a substituição de elementos na cache

Como escolher o elemento a ser substituído, em caches *n-way set associative*, $n > 1$, quando todos os elementos de um conjunto estão em uso?

Least recently used (LRU)

- É escolhido o elemento que não é acedido há mais tempo
- Difícil de implementar eficientemente

LRU aproximada

- Implementação simplificada de LRU
- Usada quando $n \geq 4$

Escolha aleatória

- *Miss rate* cerca de 10% pior para cache 2-*way set associative*
- Pode dar melhores resultados do que LRU aproximada

Princípios de localidade

A cache tenta tirar partido de dois tipos de localidade que se observam no funcionamento dos programas

Localidade temporal

Uma posição de memória acedida tenderá a ser acedida outra vez em breve

Localidade espacial

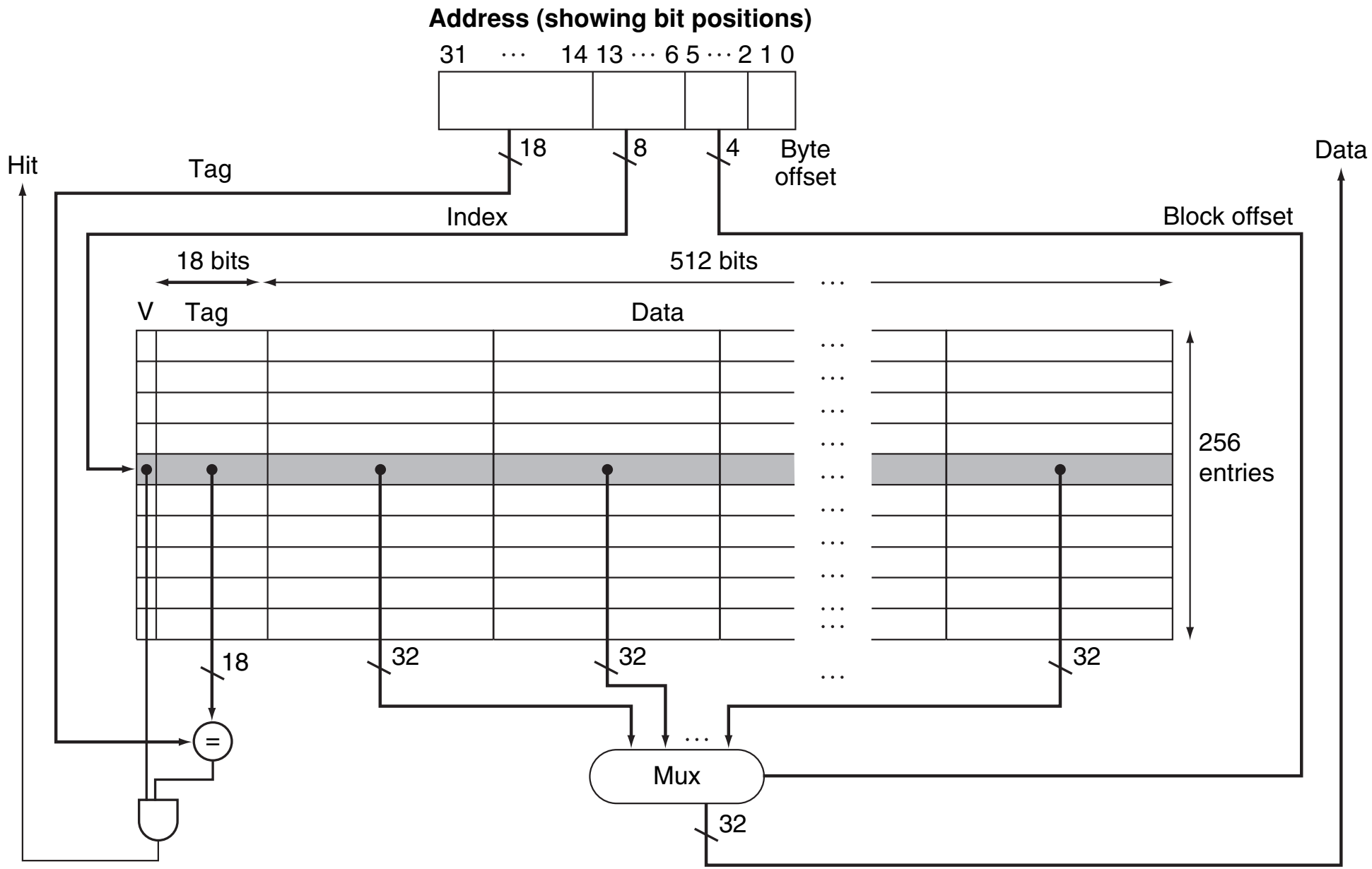
As posições de memória perto de uma posição de memória acedida tenderão a ser acedidas em breve

- Linha de cache ou bloco (de cache)
  Unidade mínima de informação presente na cache, consiste em uma ou mais palavras

  Uma posição da cache contém uma linha (ou bloco)

Cache *direct-mapped* com blocos de 16 palavras

Intrinsity FastMATH



Características de uma cache

- Número de conjuntos da cache

  Determina o conjunto em que um bloco poderá estar

- Número de blocos por conjunto

  Determina em quantas posições um bloco poderá estar

- Número de palavras por bloco (dimensão de uma linha de cache)

- Número de *bytes* por palavra

- Número de bits de um endereço

  Em conjunto com as restantes características, determina a dimensão do *tag*

- *Tag*

  Identifica o bloco presente em cada posição

Relações numa cache

endereço — endereço/número de um *byte*

$$\text{palavra} = \frac{\text{endereço}}{\textit{bytes} \text{ por palavra}}$$

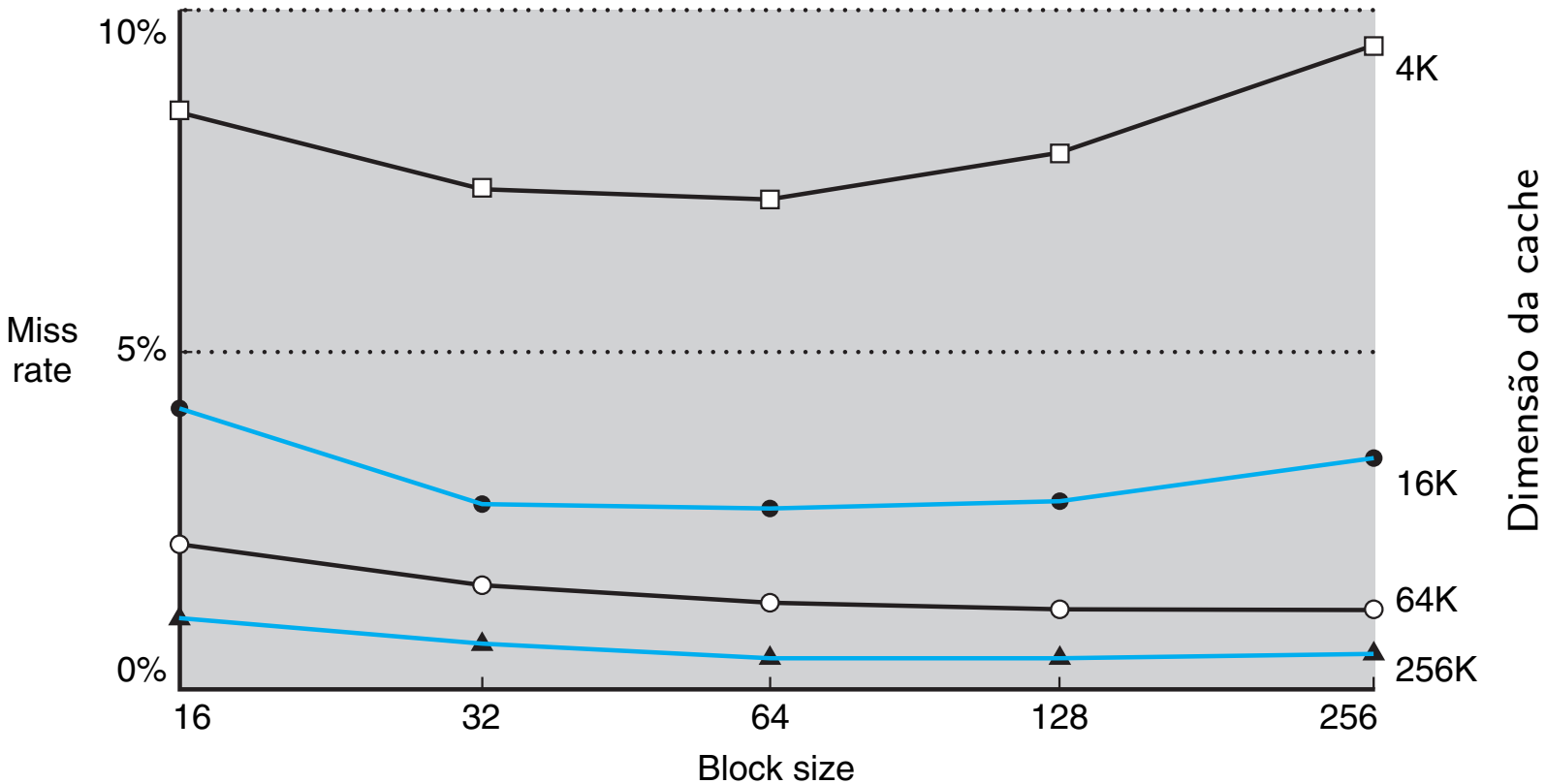
$$\text{bloco} = \frac{\text{palavra}}{\text{palavras por bloco}} = \frac{\text{endereço}}{\textit{bytes} \text{ por bloco}}$$

índice ou conjunto = bloco % nº de conjuntos

$$\textit{tag} = \frac{\text{bloco}}{\text{nº de conjuntos}}$$

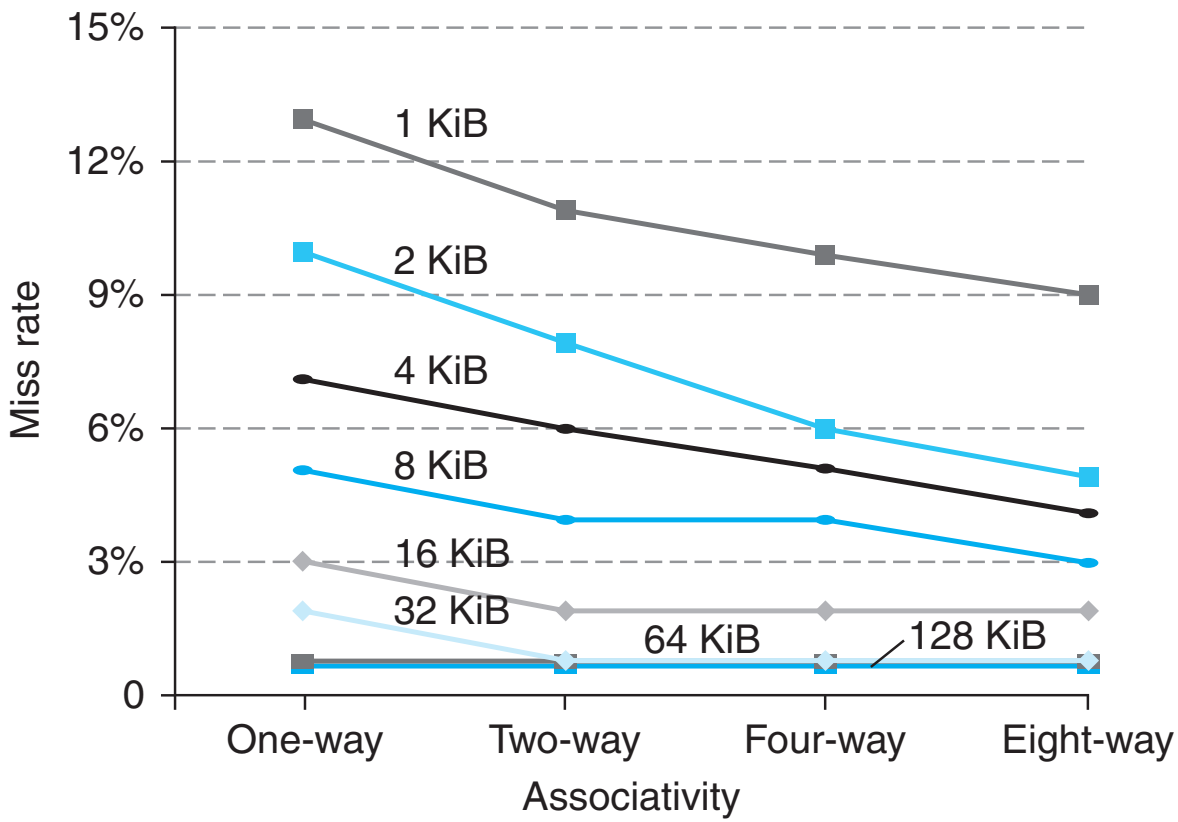
Influência da dimensão do bloco

Variação da *miss rate* com a dimensão dos blocos (em *bytes*) para várias dimensões da cache (SPEC92)



Influência da associatividade da cache

Variação da *miss rate* com a associatividade da cache para várias dimensões da cache (10 programas do SPEC2000)



(Blocos com 16 palavras)

Tipos de *miss*

Os 3 Cs

Compulsory (ou *cold-start*)
A primeira vez que é acedido um endereço pertencente a um bloco, ele não está na cache

- Relacionados com o tamanho dos blocos

Capacidade
Misses devidos ao bloco ter sido retirado da cache por a cache não ter capacidade para todos os blocos usados pelo programa

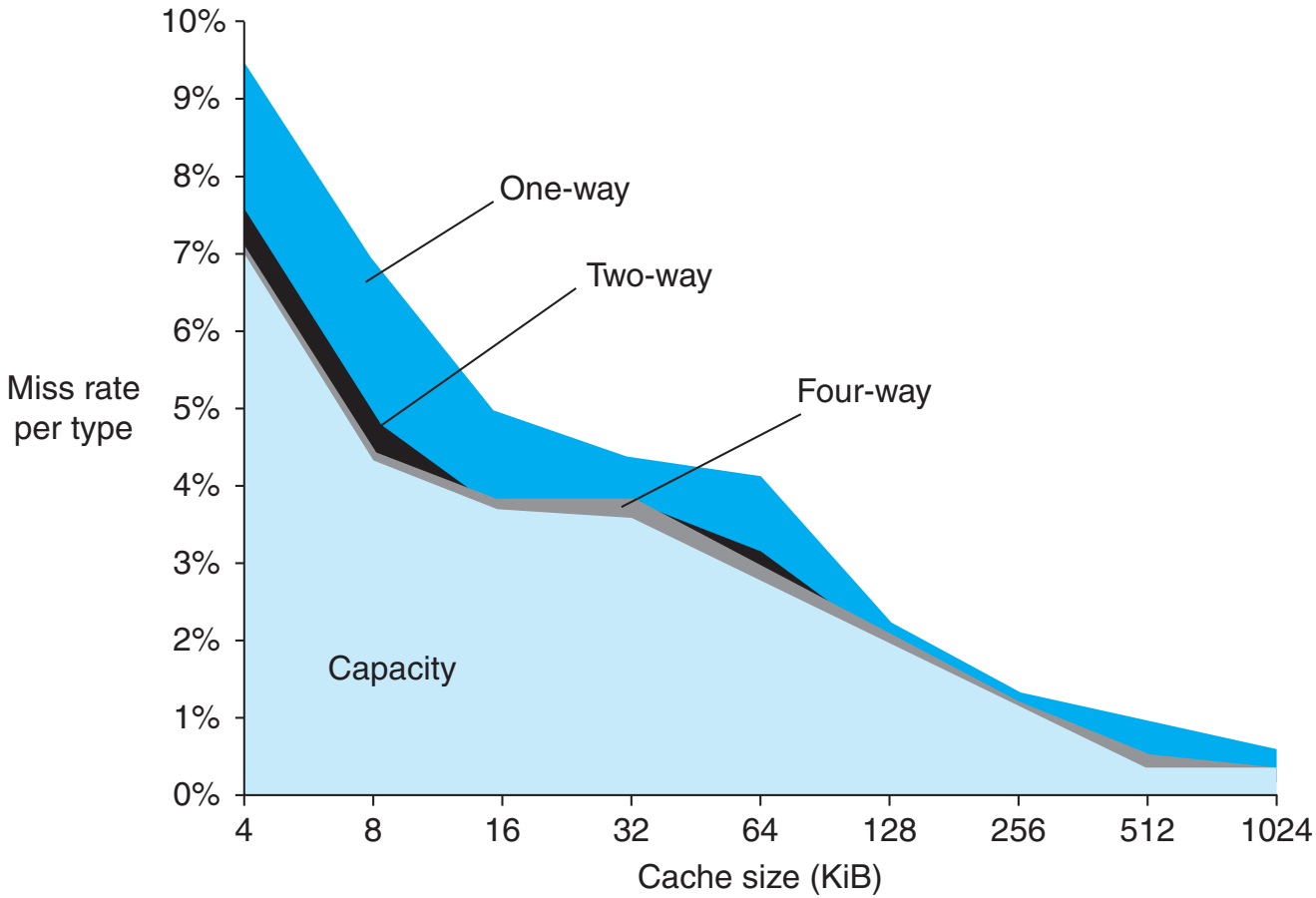
- Relacionados com o tamanho da cache

Conflito (ou colisão)
Misses devidos ao bloco ter sido retirado da cache por estar a ocupar a posição onde deverá passar a estar outro bloco, e que não ocorreriam se a cache fosse *fully associative*

- Relacionados com a associatividade da cache

Tipificação dos *misses*

Misses por tipo para várias associatividades e várias dimensões da cache (10 programas do SPEC2000)



A *compulsory miss rate* é cerca de 0.006% e não é visível no gráfico

Os *conflict misses* aparecem acima dos *capacity misses*

Os valores para 8-*way set associative* e *fully associative* não se distinguem

Escrita na memória

Como proceder?

Se o bloco está na cache (*hit*)

- Actualizar só a cache? — Estratégia *write-back*
- Actualizar a cache e a memória? — Estratégia *write-through*

Se o bloco não está na cache (*miss*)

- Ler o bloco para a cache? — Estratégia *write-allocate*
  - *Passa-se à situação do hit*
- Não ler o bloco para a cache? — Estratégia *no write-allocate*
  - É actualizada (só) a memória

Escrita na memória

Estratégias na presença de cache

Write-through

Escritas são imediatamente propagadas para o nível abaixo da memória

- Com ou sem *write-allocate*
- Pode escrever na cache antes de o bloco estar presente

Write-back (ou *copy back*)

Escritas só se reflectem no nível abaixo da memória quando o bloco é substituído

- Usa (em geral) *write-allocate*
- Substituição de um bloco modificado (*dirty*) só depois de copiado para o nível inferior (ou para um *write-back buffer*)

Pode ser usado um *write buffer* para guardar as alterações a efectuar

Acessos à memória e tempo de CPU

Ignorando os acessos à memória

$$\text{Tempo de CPU} = \text{nº ciclos execução} \times \text{duração de 1 ciclo}$$

Contando com os acessos à memória

$$\text{Tempo de CPU} = (\text{nº ciclos execução} + \text{nº ciclos } \textit{memory-stall}) \times \text{duração de 1 ciclo}$$

$$\text{nº ciclos } \textit{memory-stall} = \text{nº ciclos } \textit{read-stall} + \text{nº ciclos } \textit{write-stall}$$

$$\text{nº ciclos } \textit{read-stall} = \text{nº } \textit{reads} \times \textit{read miss-rate} \times \textit{read miss-penalty}$$

$$\text{nº ciclos } \textit{write-stall} = \text{nº } \textit{writes} \times \textit{write miss-rate} \times \textit{write miss-penalty} + \textit{write-buffer stalls}$$

Average memory access time

$$\text{AMAT} = \textit{hit time} + \textit{miss rate} \times \textit{miss penalty}$$

Caches com vários níveis

Cada nível apresenta uma *miss rate* local

Miss rate global

Percentagem de acessos que obrigam a acesso à memória principal

Exemplo

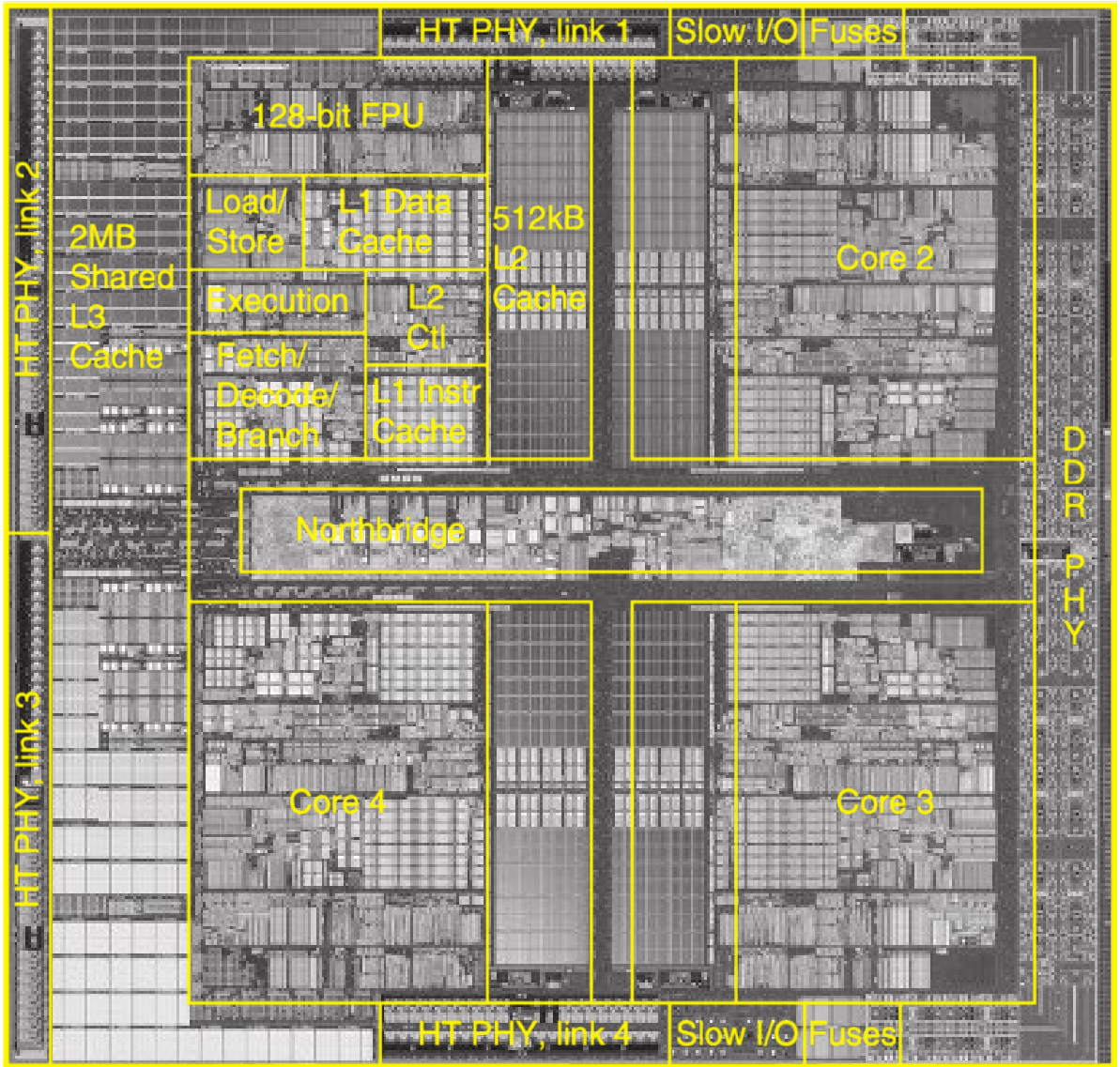
Cache com 2 níveis

$$\textit{miss rate}_{\text{L1}} = \frac{\text{N}^{\text{o}}\ \textit{misses}_{\text{L1}}}{\text{N}^{\text{o}}\ \text{acessos à cache L1}}$$

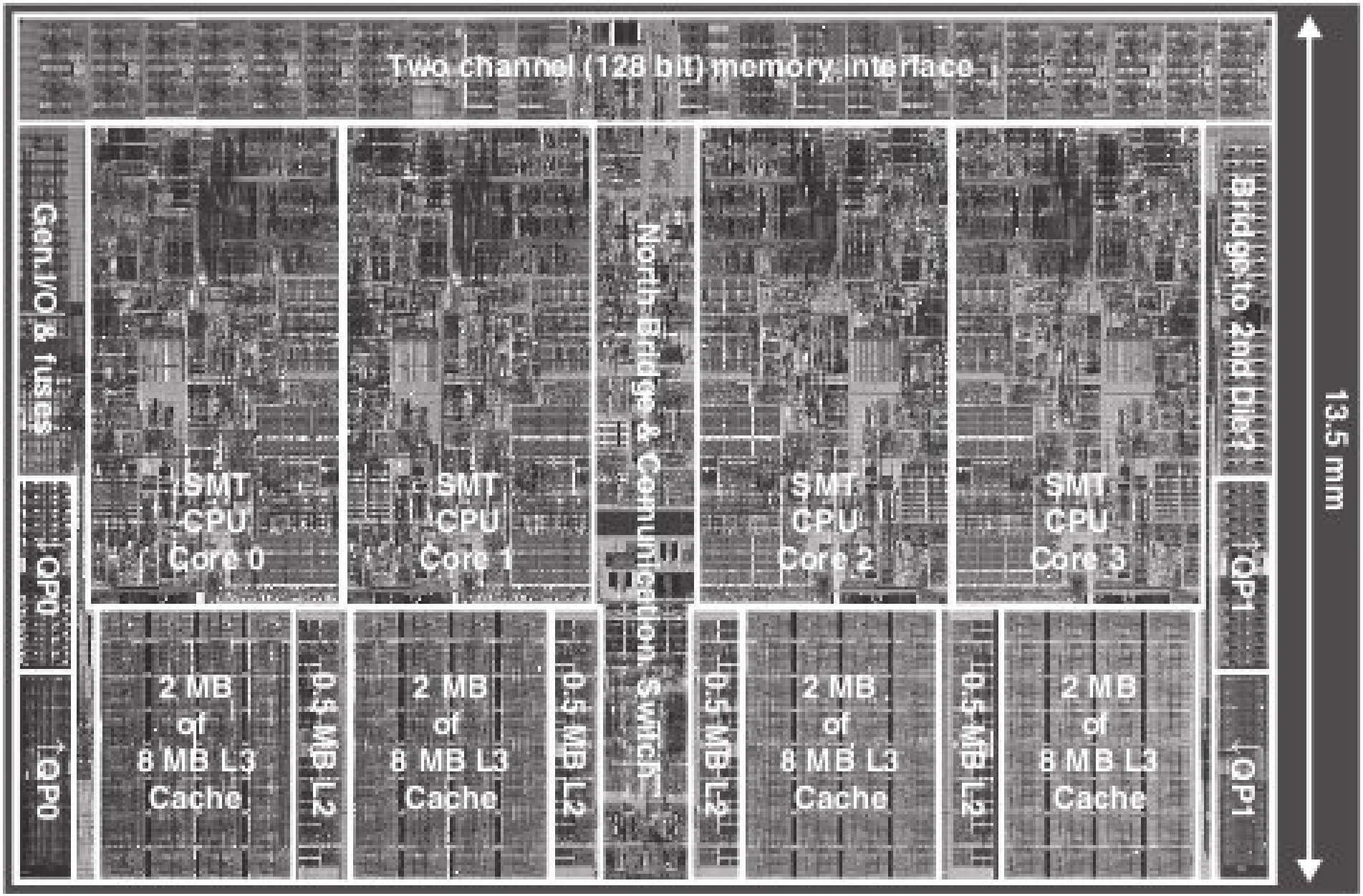
$$\textit{miss rate}_{\text{L2}} = \frac{\text{N}^{\text{o}}\ \textit{misses}_{\text{L2}}}{\text{N}^{\text{o}}\ \text{acessos à cache L2}}$$

$$\textit{miss rate}\ \text{global} = \textit{miss rate}_{\text{L1}} \times \textit{miss rate}_{\text{L2}}$$

AMD Opteron X4 (Barcelona)



Intel Nehalem



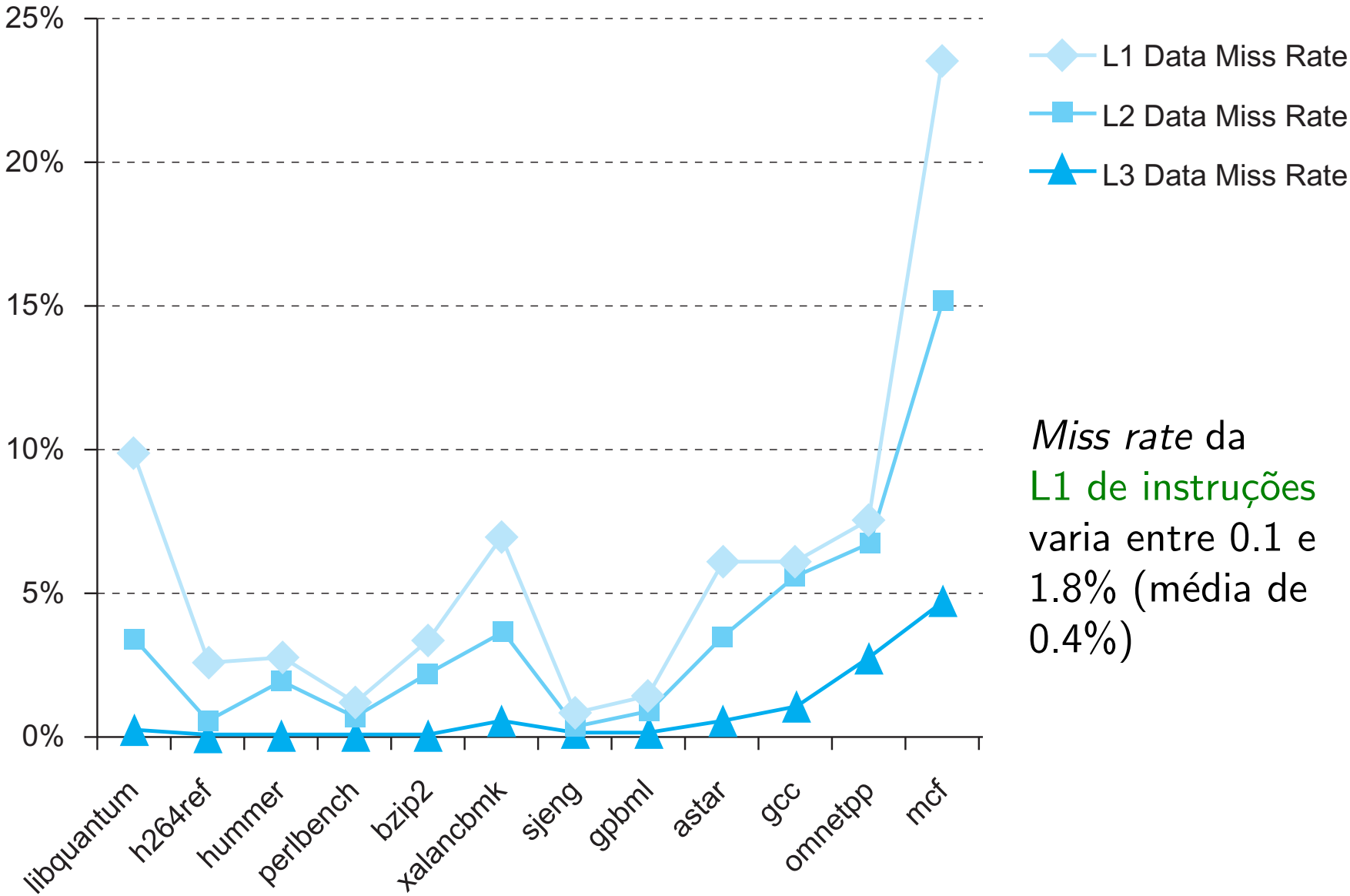
Caches: Intel Nehalem vs. AMD Opteron X4 (Barcelona)

Characteristic	Intel Nehalem	AMD Opteron X4 (Barcelona)
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	32 KB each for instructions/data per core	64 KB each for instructions/data per core
L1 cache associativity	4-way (I), 8-way (D) set associative	2-way set associative
L1 replacement	Approximated LRU replacement	LRU replacement
L1 block size	64 bytes	64 bytes
L1 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L1 hit time (load-use)	Not Available	3 clock cycles
L2 cache organization	Unified (instruction and data) per core	Unified (instruction and data) per core
L2 cache size	256 KB (0.25 MB)	512 KB (0.5 MB)
L2 cache associativity	8-way set associative	16-way set associative
L2 replacement	Approximated LRU replacement	Approximated LRU replacement
L2 block size	64 bytes	64 bytes
L2 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L2 hit time	Not Available	9 clock cycles
L3 cache organization	Unified (instruction and data)	Unified (instruction and data)
L3 cache size	8192 KB (8 MB), shared	2048 KB (2 MB), shared
L3 cache associativity	16-way set associative	32-way set associative
L3 replacement	Not Available	Evict block shared by fewest cores
L3 block size	64 bytes	64 bytes
L3 write policy	Write-back, Write-allocate	Write-back, Write-allocate
L3 hit time	Not Available	38 (?)clock cycles

Caches: ARM Cortex-A8 vs. Intel Core i7 (Nehalem)

Characteristic	ARM Cortex-A8	Intel Nehalem
L1 cache organization	Split instruction and data caches	Split instruction and data caches
L1 cache size	32 KiB each for instructions/data	32 KiB each for instructions/data per core
L1 cache associativity	4-way (I), 4-way (D) set associative	4-way (I), 8-way (D) set associative
L1 replacement	Random	Approximated LRU
L1 block size	64 bytes	64 bytes
L1 write policy	Write-back, Write-allocate(?)	Write-back, No-write-allocate
L1 hit time (load-use)	1 clock cycle	4 clock cycles, pipelined
L2 cache organization	Unified (instruction and data)	Unified (instruction and data) per core
L2 cache size	128 KiB to 1 MiB	256 KiB (0.25 MiB)
L2 cache associativity	8-way set associative	8-way set associative
L2 replacement	Random(?)	Approximated LRU
L2 block size	64 bytes	64 bytes
L2 write policy	Write-back, Write-allocate (?)	Write-back, Write-allocate
L2 hit time	11 clock cycles	10 clock cycles
L3 cache organization	–	Unified (instruction and data)
L3 cache size	–	8 MiB, shared
L3 cache associativity	–	16-way set associative
L3 replacement	–	Approximated LRU
L3 block size	–	64 bytes
L3 write policy	–	Write-back, Write-allocate
L3 hit time	–	35 clock cycles

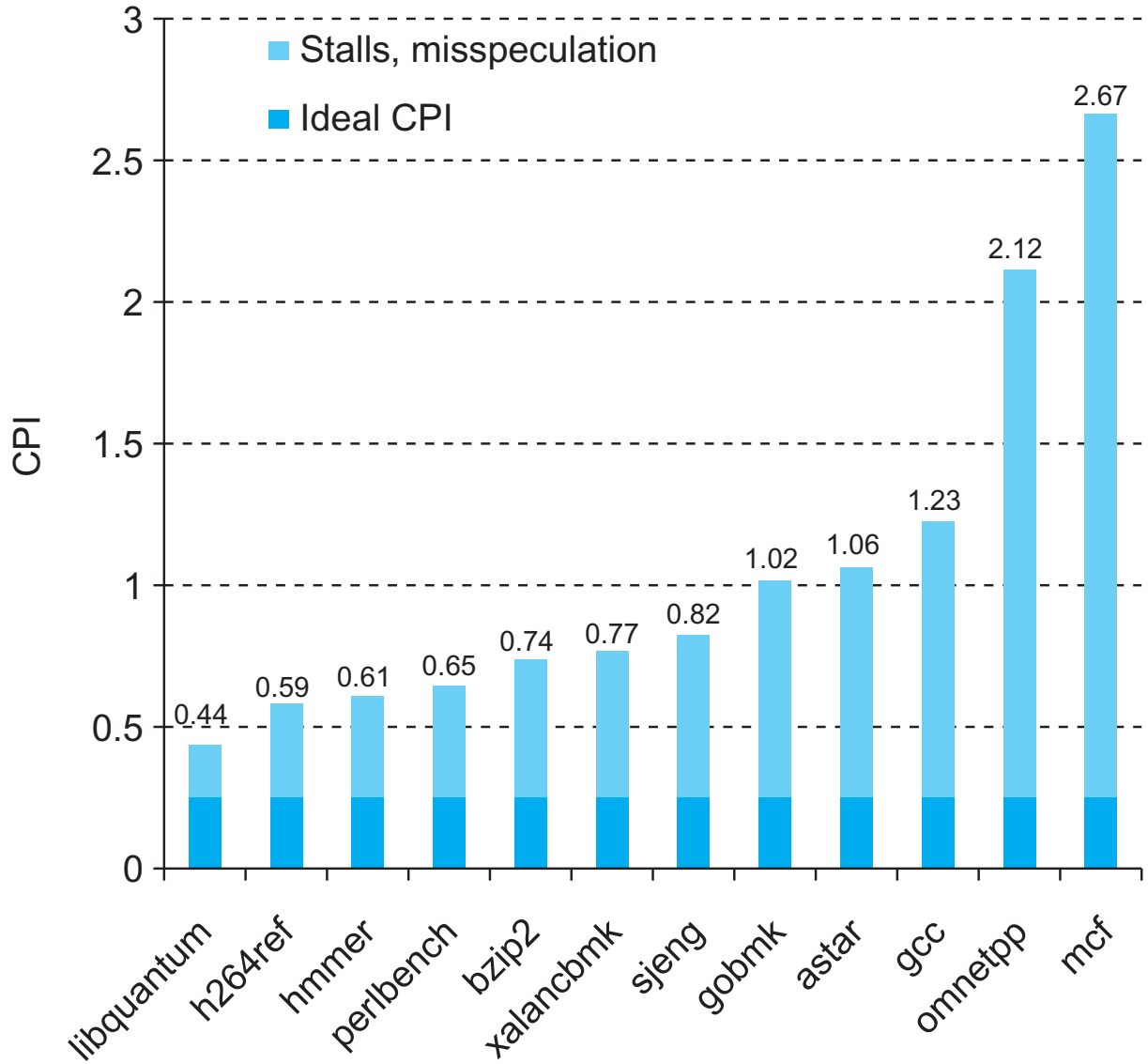
Comportamento da cache do Intel Core i7 920 (Nehalem)



Miss rate da L1 de instruções varia entre 0.1 e 1.8% (média de 0.4%)

Valores obtidos para SPECint2006

CPI no Intel Core i7 920 (Nehalem)



Valores obtidos para SPECint2006