Cache Optimizations

Hardware Approaches for reducing cache misses
Cache Optimization Space

Total cache size: Determines chip area and number of transistors

Performance factors:
- Miss rate, miss penalty, and hit time

Organization:
- Set Associativity and block size
- Multi-level organizations
- Auxiliary structures, e.g., to predict future accesses
- Main memory and memory interface design

Software Approaches
- Optimize memory access patterns
- Software prefetching

Execution time increase is the final measurement, and AMAT is a good approximation. Other metrics: miss per instruction, memory CPI
## Improving Cache Performance

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>1. Reducing miss rates</strong></td>
<td><strong>3. Reducing miss penalty or miss rates via parallelism</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Larger block size</td>
<td>Reduce miss penalty or miss rate by parallelism</td>
</tr>
<tr>
<td>larger cache size</td>
<td></td>
</tr>
<tr>
<td>higher associativity</td>
<td>Non-blocking caches</td>
</tr>
<tr>
<td>way prediction</td>
<td></td>
</tr>
<tr>
<td>Pseudoassociativity</td>
<td>Hardware prefetching</td>
</tr>
<tr>
<td>compiler optimization</td>
<td>Compiler prefetching</td>
</tr>
<tr>
<td><strong>2. Reducing miss penalty</strong></td>
<td><strong>4. Reducing cache hit time</strong></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Multilevel caches</td>
<td>Small and simple caches</td>
</tr>
<tr>
<td>critical word first</td>
<td>Avoiding address translation</td>
</tr>
<tr>
<td>read miss first</td>
<td></td>
</tr>
<tr>
<td>merging write buffers</td>
<td>Pipelined cache access</td>
</tr>
<tr>
<td>victim caches</td>
<td>Trace caches</td>
</tr>
</tbody>
</table>
Classifying cache misses

Classifying misses by causes (3Cs)

- **Compulsory**—To bring blocks into cache for the first time. Also called cold start misses or first reference misses. *(Misses in even an Infinite Cache)*

- **Capacity**—Cache is not large enough such that some blocks are discarded and later retrieved. *(Misses in Fully Associative Size X Cache)*

- **Conflict**—For set associative or direct mapped caches, blocks can be discarded and later retrieved if too many blocks map to its set. Also called collision misses or interference misses. *(Misses in N-way Associative, Size X Cache)*

More recent, 4th “C”:

- **Coherence**—Misses caused by cache coherence. To be discussed in multiprocessor
3Cs Absolute Miss Rate (SPEC92)

Compulsory vanishingly small
2:1 Cache Rule

miss rate 1-way associative cache size X = miss rate 2-way associative cache size X/2
3Cs Relative Miss Rate

Flaws: for fixed block size
Good: insight => invention

Conflict

Capacity

Cache Size (KB)

1-way
2-way
4-way
8-way

Compulsory
How To Reduce Misses?

3 Cs: Compulsory, Capacity, Conflict

In all cases, assume total cache size not changed:

What happens if:
1) Change Block Size:
   Which of 3Cs is obviously affected?

2) Change Associativity:
   Which of 3Cs is obviously affected?

3) Change Compiler:
   Which of 3Cs is obviously affected?
1. Reduce Misses via Larger Block Size

![Graph showing the relationship between block size and miss rate for different block sizes (1K, 4K, 16K, 64K, 256K). The graph indicates that increasing block size generally reduces the miss rate.](image-url)
2. Reduce Misses via Higher Associativity

建设用地 1 请求率:
- Miss Rate DM cache size N - Miss Rate 2-way cache size N/2

注意：执行时间只是最终衡量标准！
- Will Clock Cycle time increase?
- Hill [1988] suggested hit time for 2-way vs. 1-way external cache +10%,
  internal + 2%

Jouppi’s Cacti model: estimate cache access time by block number, block size, associativity, and technology
- Note cache access time also increases with cache size!
Example: Avg. Memory Access Time vs. Miss Rate

Example: assume $CCT = 1.10$ for 2-way, 1.12 for 4-way, 1.14 for 8-way vs. CCT direct mapped

<table>
<thead>
<tr>
<th>Cache Size (KB)</th>
<th>1-way</th>
<th>2-way</th>
<th>4-way</th>
<th>8-way</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.33</td>
<td>2.15</td>
<td>2.07</td>
<td>2.01</td>
</tr>
<tr>
<td>2</td>
<td>1.98</td>
<td>1.86</td>
<td>1.76</td>
<td>1.68</td>
</tr>
<tr>
<td>4</td>
<td>1.72</td>
<td>1.67</td>
<td>1.61</td>
<td>1.53</td>
</tr>
<tr>
<td>8</td>
<td>1.46</td>
<td>1.48</td>
<td>1.47</td>
<td>1.43</td>
</tr>
<tr>
<td>16</td>
<td>1.29</td>
<td>1.32</td>
<td>1.32</td>
<td>1.32</td>
</tr>
<tr>
<td>32</td>
<td>1.20</td>
<td>1.24</td>
<td>1.25</td>
<td>1.27</td>
</tr>
<tr>
<td>64</td>
<td>1.14</td>
<td>1.20</td>
<td>1.21</td>
<td>1.23</td>
</tr>
<tr>
<td>128</td>
<td>1.10</td>
<td>1.17</td>
<td>1.18</td>
<td>1.20</td>
</tr>
</tbody>
</table>

(Red means A.M.A.T. not improved by more associativity)
3. Reducing Misses via a “Victim Cache”

- How to combine fast hit time of direct mapped yet still avoid conflict misses?
- Add buffer to place data discarded from cache
- Jouppi [1990]: 4-entry victim cache removed 20% to 95% of conflicts for a 4 KB direct mapped data cache
- Used in Alpha, HP machines
4. Reducing Misses via “Pseudo-Associativity”

- How to combine fast hit time of Direct Mapped and have the lower conflict misses of 2-way SA cache?
- Divide cache: on a miss, check other half of cache to see if there, if so have a *pseudo-hit* (slow hit)

<table>
<thead>
<tr>
<th>Hit Time</th>
<th>Miss Penalty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pseudo Hit Time</td>
<td></td>
</tr>
</tbody>
</table>

- Drawback: CPU pipeline is hard if hit takes 1 or 2 cycles
  - Better for caches not tied directly to processor (L2)
  - Used in MIPS R10000 L2 cache, similar in UltraSPARC
5. Reducing Misses by **Hardware**

Prefetching of Instructions & Datals

- **E.g., Instruction Prefetching**
  - Alpha 21064 fetches 2 blocks on a miss
  - Extra block placed in “stream buffer”
  - On miss check stream buffer

- **Works with data blocks too:**
  - Jouppi [1990] 1 data stream buffer got 25% misses from 4KB cache; 4 streams got 43%
  - Palacharla & Kessler [1994] for scientific programs for 8 streams got 50% to 70% of misses from 2 64KB, 4-way set associative caches

- **Prefetching relies on having extra memory bandwidth that can be used without penalty**
6. Reducing Misses by **Software** Prefetching Data

**Data Prefetch**
- Load data into register (HP PA-RISC loads)
- Cache Prefetch: load into cache (MIPS IV, PowerPC, SPARC v. 9)
- Special prefetching instructions cannot cause faults; a form of speculative execution

**Prefetching comes in two flavors:**
- Binding prefetch: Requests load directly into register.
  - Must be correct address and register!
- Non-Binding prefetch: Load into cache.
  - Can be incorrect. Frees HW/SW to guess!

**Issuing Prefetch Instructions takes time**
- Is cost of prefetch issues < savings in reduced misses?
- Higher superscalar reduces difficulty of issue bandwidth