Recitation 2.4

April 14, 2025

Some student optimizations

- Replacing pow() with multiplication
- Attempting vectorization
- Avoid repetition of distance computation
- A tighter bounding box for render

Announcement: Bonus tier cut-off is kept internal.

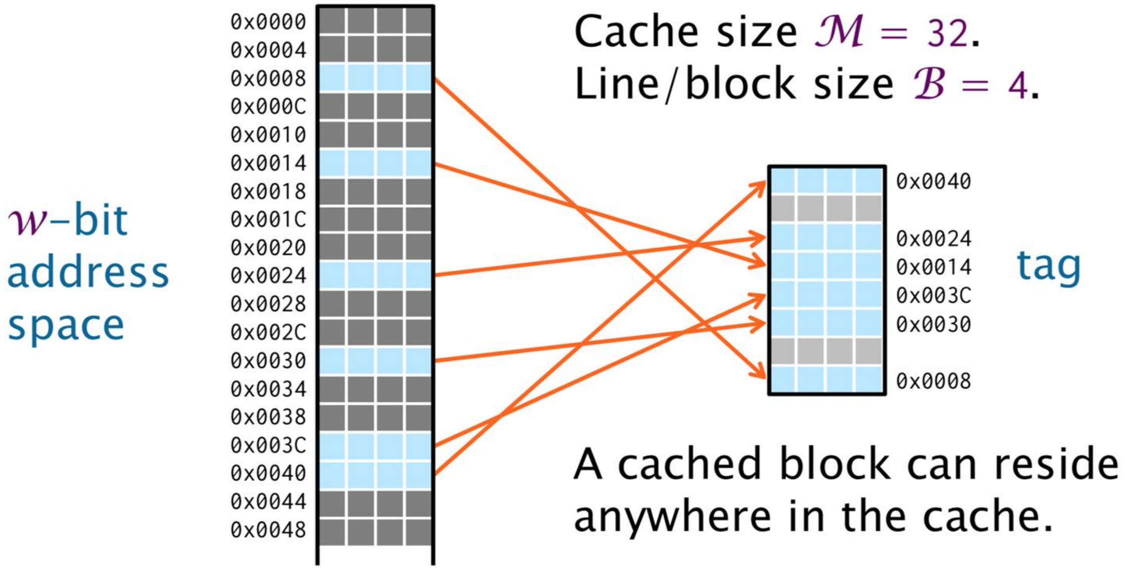
Cache analysis

Overview

- Fully Associative Cache
- Ideal cache model
- Tall cache assumption
- Submatrix caching lemma
- Example cache analysis

Fully associative

- Cached blocks can reside anywhere in cache.
- To find a block, search through the entire cache.
- Which block you evict depends on the evicting strategy. (e.g. LRU)



How Reasonable Are Ideal Caches?

"LRU" Lemma [ST85]. Suppose that an algorithm incurs Q cache misses on an ideal cache of size $\mathcal{M}$. Then on a fully associative cache of size $2\mathcal{M}$ that uses the least-recently used (LRU) replacement policy, it incurs at most $2Q$ cache misses. ■

Implication

For asymptotic analyses, one can assume optimal or LRU replacement, as convenient.

Software Engineering

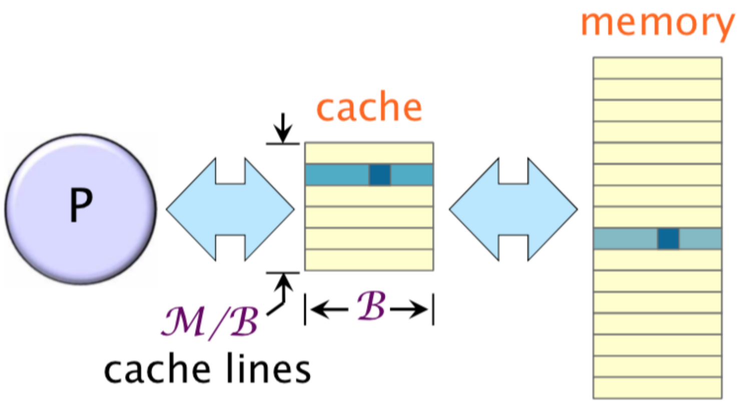
- Design a theoretically good algorithm.
- Engineer for detailed performance.
  - Real caches are not fully associative.
  - Loads and stores have different costs with respect to bandwidth and latency.

Ideal Cache Model

Ideal-Cache Model

Parameters

- Two-level hierarchy.
- Cache size of $\mathcal{M}$ bytes.
- Cache-line length of $\mathcal{B}$ bytes.
- Fully associative.
- Optimal, omniscient replacement.

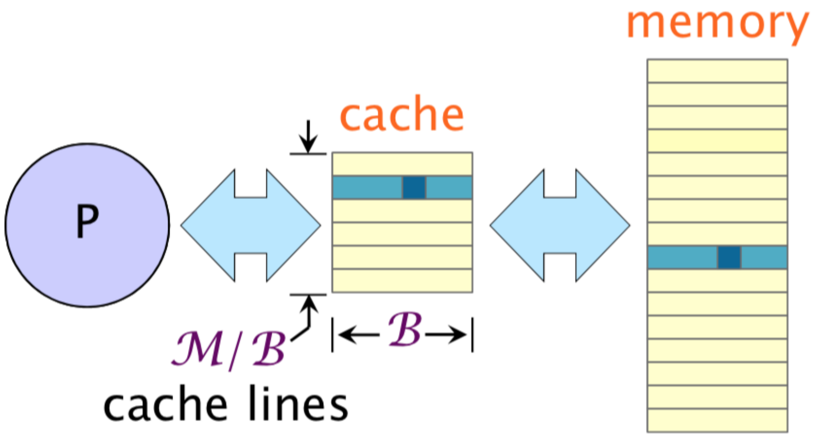


Performance Measures

- work W (ordinary running time)
- cache misses Q

Tall Cache Assumption

Tall Caches



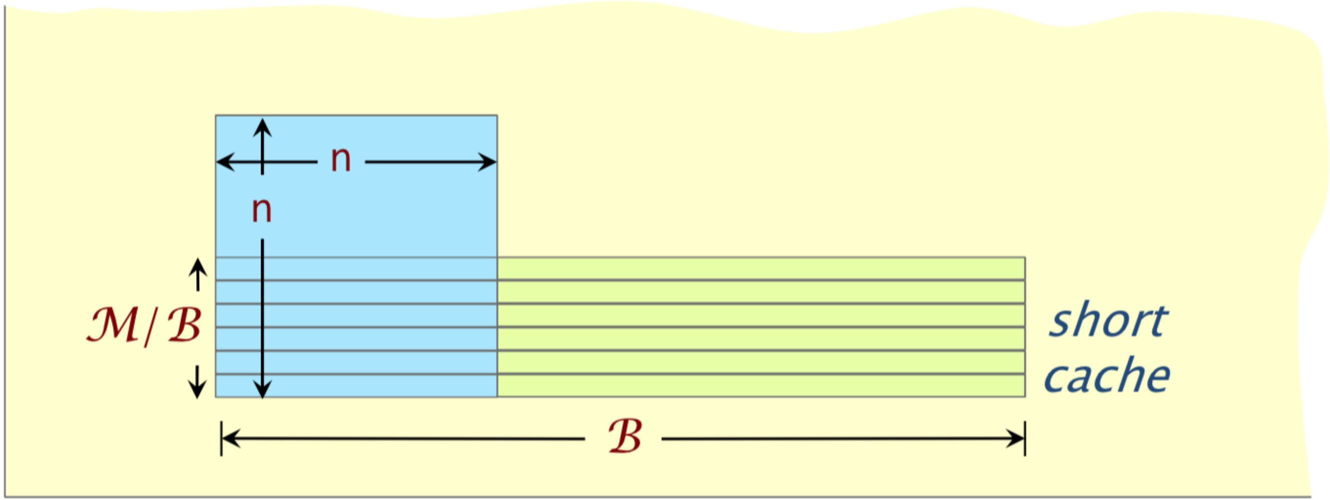
Tall-cache assumption

$\mathcal{B}^2 < c\mathcal{M}$ for some sufficiently small constant $c \leq 1$.

Example: Intel Xeon E5-2666 v3

- Cache-line length = 64 bytes.
- L1-cache size = 32 Kbytes.

What's Wrong with Short Caches?



Tall-cache assumption
$\mathcal{B}^2 < c\mathcal{M}$ for some sufficiently small constant $c \leq 1$.

An $n \times n$ submatrix stored in row-major order may not fit in a short cache even if $n^2 < c\mathcal{M}$!

Submatrix Caching Lemma

Cache-Miss Lemma

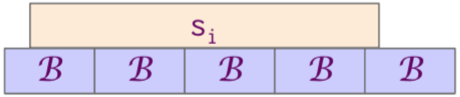
Lemma. Suppose that a program reads a set of r data segments, where the ith segment consists of s_i bytes, and suppose that

$$\sum_{i=1}^{r} s_i = N < \mathcal{M}/3 \text{ and } N/r \geq \mathcal{B} .$$

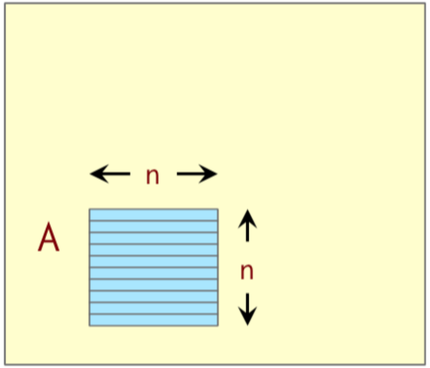
Then all the segments fit into cache, and the number of misses to read them all is at most $3N/\mathcal{B}$.

Proof. A single segment s_i incurs at most $s_i/\mathcal{B} + 2$ misses, and hence we have

$$\begin{aligned} \sum_{i=1}^{r} (s_i/\mathcal{B} + 2) &= N/\mathcal{B} + 2r \\ &= N/\mathcal{B} + (2r\mathcal{B})/\mathcal{B} \\ &\leq N/\mathcal{B} + 2N/\mathcal{B} \\ &= 3N/\mathcal{B} . \ \blacksquare \end{aligned}$$



Submatrix Caching Lemma



Lemma. Suppose that an $n \times n$ submatrix A is read into a tall cache satisfying $\mathcal{B}^2 < c\mathcal{M}$, where $c \leq 1$ is constant, and suppose that $c\mathcal{M} \leq n^2 < \mathcal{M}/3$. Then A fits into the cache, and the number of misses to read all of A's elements is at most $3n^2/\mathcal{B}$.

Proof. We have $r = n$, $s_i = n$, $N = n^2$. Since $\mathcal{B}^2 < c\mathcal{M} \leq n^2$, we have $\mathcal{B} \leq n = N/r$. Also, $N < \mathcal{M}/3$. Thus, the Cache-Miss Lemma applies. ■

Solve $T(n) = aT(n/b) + f(n)$, where $a \geq 1$ and $b > 1$.

CASE 1	$f(n) = O(n^{\log_b a - \epsilon})$ constant $\epsilon > 0$	⇒	$T(n) = \Theta(n^{\log_b a})$
CASE 2	$f(n) = \Theta(n^{\log_b a} \lg^k n)$ constant $k \geq 0$	⇒	$T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$
CASE 3	$f(n) = \Omega(n^{\log_b a + \epsilon})$ constant $\epsilon > 0$ (and regularity)	⇒	$T(n) = \Theta(f(n))$