Lecture 3

DRAM Access Patterns

Rule	Ideal Values	Description
Granularity	6 - 64 B	Amount of data transferred in a single request
		Reading smaller sizes is very inefficient
Locality	1- 4kB	If you use an address now, you will likely use it again soon (fetch from cache, not memory)
		If you use an address now, you will likely use one close to it soon (In same cache line, if realated objects are stored too far away from eachother, the cache line can be flushed causing a memory read (inefficient)
L1, L2 caching	64-256 kB	Set of bytes read/written repeatedly is stored here until replaced, subsequent reads and writes hit the cache not memory, much faster. Shared between 2 threads
L3 Caching	8-20MB	""__"". Shared between all cores

Hit Ratio

- The Hit Ratio is the proportion of request that hit in the cache
- It is a measure of effectiveness of the caching mechanism
- Depends on locality of application

Cache Flushing/ Re-filling

- When a cache becomes full and needs to make room for a new entry it must first evict an old one
- Needs a replacement policy
  - Random Works well in practice

Cache Topologies

Fully Associative Caches

- Any line can be anywhere in the cache
  - Advantage: Can replace any line
  - Disadvantage: Hard to find specific lines

Direct Mapped Cache

- Every address has exactly one slot

    - Advantage: easy to find a line
    - Disadvantage: must replace an entire fixed line

K-Way Set Associative Cache

- Each slot holds *k* lines
    - Advantage : relatively easy to find a line
    - Disadvantage: Some choice needed in replacing a line

Cache Coherence

- Given 2 cores/ threads **A** and **B**
- If **A** and **B** both cache address x
- **A** writes to x
    - updates cache
- How does **B** find out?
- There are many different protocols

MESI

- Modified
    - Any modified cache data must be written back to memory
- Exclusive
    - Not modified, *I* have the only copy
- Shared
    - Not modified, may be cached elsewhere
- Invalid
    - Cache contents is not meaningful
- **See slide deck for diagrammatic explanation**

Write-through Caches

- Immediately broadcast changes
    - Advantages:
        - Memory and caches always agree
        - May lead to more read hits
    - Disadvantages:
        - Bus traffic on all writes
        - Most writes to unshared data
            - e.g. loop indexes

Write-back Caches

- Accumulate changes in cache
- Write back when line evicted
    - Occurs when either:
        - Cache space is needed for something else
        - Another processor wants the data