The Memory Hierarchy & Locality

Why do we need to know about the memory hierarchy/locality?

- These are topics related to computer architecture
  - CS 333: you might be taking this soon
  - So why introduce this here in CS216?
- Course focus: levels of representation, abstraction. So apply this to:
  - How we think about efficiency
  - How we think about data in memory

Efficiency, Order Classes

- Recall order classes (Big-Oh, Big-Theta…)
  - Differences are meaningful when...
    - Inputs get large
    - You find a difference in order-class
- A high-level comparison tool. Not so useful when:
  - Two solutions are in the same order-class
  - It’s important to fine-tune an algorithm

Efficiency, Order Classes

- Order classes based on counting statements
  - Is even this kind of counting a useful tool for more fine-grained analysis of an algorithm?
- One assumption is that all operations take the same amount of time.
  - Is that really true?

Definitions

Cycle – (for our purposes) the time it takes to execute a single simple instruction. (ex. Add 2 registers together)
Memory Latency – time it takes to access memory
What can be done?

- Goal: Attempt to reduce the number of accesses to the slower levels.
- How?

Locality

**Temporal Locality** (locality in time) – If an item is referenced, it will tend to be referenced again soon.

**Spatial Locality** (locality in space) – If an item is referenced, items whose addresses are close by will tend to be referenced soon.

Caches

- Each level is a sub-set of the level below.
- Example:
  - `mov eax, [i]`
  - `add ecx, [i]`

  **Cache Hit** – address requested is in cache
  **Cache Miss** – address requested is NOT in cache

  **Cache line size** (chunk size) – the number of contiguous bytes that are moved into the cache at one time

Some Numbers

- Here are some typical numbers for a 1 GHz processor
  - Cycle time: 1 ns
  - L1 cache: accessed in 2-3 ns
  - L2 cache: accessed in 20-50 ns
  - Main memory: accessed in 60-100 ns
  - Disk: accessed in 5-12 ms
- (Reminder: ns = 10^(-9); ms = 10^(-3))

More Example Numbers

- L1 cache may hold 32-256 Kb
- L2 cache may hold 1-32 Mb
- Main memory: 512Mb – 2 Gb
- Disk: 40-400 Gb
Trends

- In the 20 years between 1980 and 2000:
  - CPU speed: 600x
  - SRAM: capacity: 200x, latency: 100x
  - DRAM: capacity: 8,000x, latency: 6x
  - Disk: capacity: 50,000x, latency: 10x

- See any implications?

Who is working on this problem?

- Architects
- Compiler Writers
- Programmers
- Operating System

Locality and Data Structures

- Which has better spatial locality, arrays or linked lists?

Compiler Writers

```
loop:    mov eax, [i]
dec eax
mov [i], eax
cmp eax, 0
jg  loop
```