CS2410: Computer Architecture

Principles of computer architecture & design

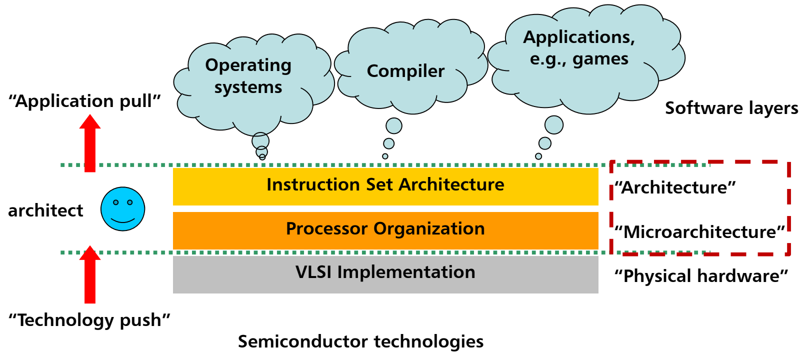
Sangyeun Cho

Computer Science Department
University of Pittsburgh

Dynamicity in computer architecture

- New technologies are developed and deployed continuously
  - Examples: cheaper & faster transistors (Moore's Law), storage class memory (best of DRAM and NAND flash), LCD and OLED displays, …
  - They expand the capabilities of a computer at a lower price
  - "Technology push"
- Need for new, more exciting applications call for higher computer performance and more capabilities
  - Examples: realistic 3D graphics based games, intelligent mining & management of large volumes of data (e.g., movies), …
  - "Application pull"
- Nonetheless, there are certain characteristics in computer design that appear to remain

Dynamicity in computer architecture



Operating systems

Compiler

Applications, e.g., games

Software layers

"Application pull"

architect

Instruction Set Architecture

Processor Organization

VLSI Implementation

"Architecture"

"Microarchitecture"

"Physical hardware"

"Technology push"

Semiconductor technologies

Asymmetry

- In a sense, computer architecture is all about how to overcome and/or exploit "asymmetry" present in computer hardware resources and software artifacts
- Can you give an example?

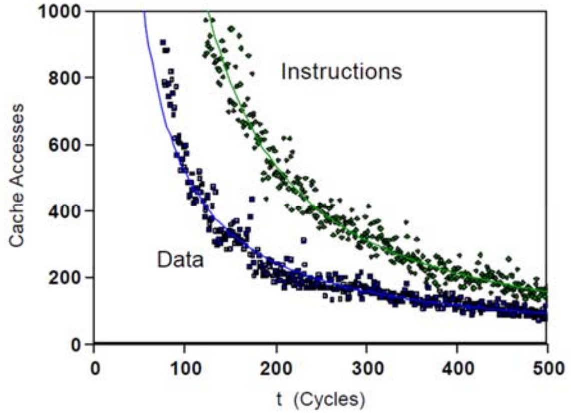
Asymmetry in hardware resources

- CPU operation latencies
  - ADD vs. DIV
- CPU speed vs. memory speed improvement rates
- Memory technology
  - SRAM latency vs. DRAM latency
- Main memory organization
  - Non-Uniform Memory Architecture (NUMA)
- On-chip shared L2 cache organization
  - Non-Uniform Cache Architecture (NUCA)
- Storage
  - Hard disk access (which block is read now and which block is next?)
  - SLC (single-level cell) vs. MLC (multi-level cell) NAND flash

Asymmetry in software behavior

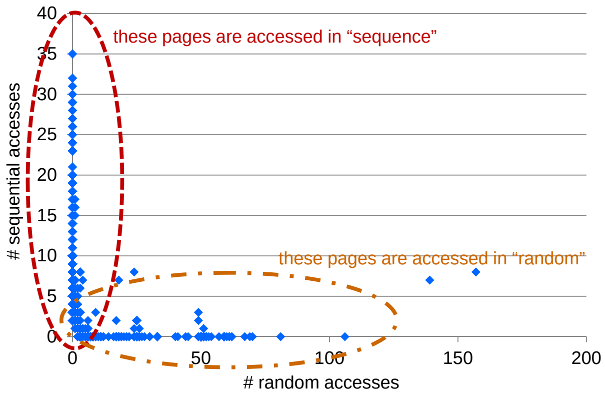
- 90/10 rule
  - Code repetition
- Memory access locality
  - Temporal
  - Spatial
- Working set & miss rate
- Storage access locality

Temporal locality example



Hartstein et al. JILP 2008

Sequential vs. random writes



"FINANCIAL1 TRACE"

Hashing

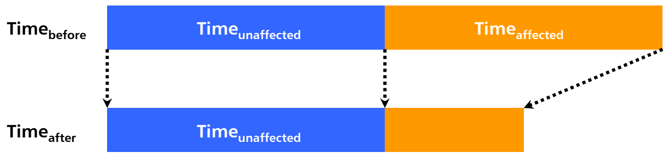
- Oftentimes, we need to collect & accumulate program execution information so that we can make informed decisions on resource usages
  - Branch prediction
  - Cache memory
- Information storage & retrieval mechanisms should be simple to enable efficient implementation
- What is hashing? ;-)
- Let’s take a look at how “memory” is organized at this point

Parallelism

- The key enabler for performance improvement in computer architecture is to uncover and exploit parallelism
  - Draw of graph (V,E) where nodes represent a task and edges show dependences
- Examples
  - Pipelining
  - Multicore designs
  - Memory-level parallelism (superscalar processor)
  - Interleaved memory
  - Interleaved cache
  - RAID
  - Multiple channels in SSDs
  - …

Amdahl’s law

- Optimization or parallelization usually applies to a portion
  - Places “limitation” of the scope of an optimization
  - Leads us to focus on “common cases”
  - “Make common case fast and rare case accurate”


$\text{Time}_{\text{before}}$
$\text{Time}_{\text{unaffected}}$
$\text{Time}_{\text{affected}}$
$\text{Time}_{\text{after}}$
$\text{Time}_{\text{unaffected}}$