Architecture Topics

Topics Overview

- Classes of parallelism (data, control, shared/distributed memory, OoO, superscalar, pipelines, multithreading, heterogeneity)
- Underlying mechanisms (caching, atomicity, consistency, coherence, interrupts/events, handshaking, ID)
- Floating point representation (in support of HPC)
- Performance metrics (IPC, benchmarks, network/memory bandwidth, peak performance, sustained perfomance)
- Power (power/energy, larger scale, embedded, density, static/dynamic, DVFS)
- Scaling (Big data, HPC, fault tolerance, data bound computation, volume, velocity, scale out, cost of data movement)

Suggested Additions/Notes

SMP, Cluster, Supercomputer arch	SMP, Cluster in curriculum, SC advanced
FPGA, microcontroller	Include for a CE curriculum?
Hardware and system simulators	Advanced architecture topic
Switch CPI to IPC	Easy to change
Dependences, functional unit throughput	Already in curriculum
Superscalar, OoO	Already in curriculum
Weak consistency	Already in curriculum
Performance experiment design	Grad level topic for research methods
Performance modeling	Upper level elective topic
Disk, flash, NVRAM, RAID	Already in most architecture texts
Replace Amdahl's law w/ Brent's Theorem	Cover Brent's in Theory, Amdahl in Arch?

Programming Topics

Programming Topics - Overview

- Main goal is to introduce parallel programming topics into intro programming, data structures, and systems classes
  - Secondary goal is to target upper-level classes
- High-level themes include:
  - **Paradigms and notations** – SIMD, shared memory, message passing, client/server, big data stack, threads, tasks, data parallel, etc.
  - **Semantics and correctness** – synchronization, concurrency defects, …
  - **Memory models** – sequential consistency, weak consistency, …
  - **Performance and energy** – computation and data decomposition, scheduling/mapping, data layout and locality, tools and metrics
- Most topics at a shallow level (Bloom level C or K) for intro courses, but at a deeper level for upper-level courses (or deferred to upper-level completely, so at N Bloom level for intro courses)

Programming Topics - Updates

- Incorporated new programming topics related to distributed computing (e.g. client/server), big data (e.g., MapReduce), and power/energy
- Eliminated some topics from original guidelines completely, since no longer relevant
- Added small number of other topics missed in original guidelines, or newer ideas (e.g., accelerator programming)

Programming Topics – Input from Reviewers

- Several suggestions to do a better job on cross-cutting ideas
  - Across programming areas, algorithms, architecture – e.g., scalability
- Better definitions of terms and acronyms
- More limited energy/power topics – keep them high level
- Some suggestions for updating topics, new topics, eliminating topics no longer relevant
  - Under discussion for revisions
  - We are looking for volunteers to help!

Algorithms Topics

PDC in Algorithms

- Builds on traditional sequential coverage
- Introduces PDC “dots” early on that can be connected downstream
- PDC coverage along three broad streams
  - Concepts and Issues Concurrency, communication, differences between resources (time, space, energy), scheduling, load-balancing
  - Techniques: broadcast, reduction, prefix computations, data decomposition, MapReduce
  - Problems: Sorting, Matrix multiplication, prefix
- Inclusion of ideas motivated by Big Data, Energy considerations and Distributed computing

PDC In Algorithms

- Recent Feedback: Some PDC topics not suitable for CS1 in many settings

- Qualitative introduction to Concurrency, asynchrony and dependencies
- Model of choice
- Connecting across courses