Special Topics on Networking and Distributed Systems

CSCI-2950u :: Data-Intensive Scalable Computing

Rodrigo Fonseca (rfonseca)

http://www.cs.brown.edu/courses/csci2950-u

brown-csci2950-u-f11@googlegroups.com

Based partly on lecture notes by David Mazières, Phil Levis, John Jannotti, Peterson & Davie
Welcome!

• **Course:** TuTh, 10:30-11:50, 506 CIT
• **Me:** Rodrigo Fonseca
  – Have been here for 2 years
  – Interested in distributed systems, operating systems, networking
  – You can find me at rfonseca@cs.brown.edu or in 329 CIT (with an appointment)
Data-Intensive Scalable Computing

• What does this mean?
  – Sometimes know as DISC
  – Data-Intensive: data deluge!
  – Scalable: computing should keep up as the data grows
  – Our focus:
    • Systems: how to build these scalable systems
    • Applications: what applications/algorithms are suitable for these systems
The Deluge, John Martin, 1834. Image from wikimedia commons.
How much data?

http://www.economist.com/node/15579717
How much data?

Google: 20 PB/day (2008)

Facebook: 36PB of user data
90TB/day (2010)

LSST: 3TB/day of image data

DNA Sequencing Throughput:
Increasing 5x per year

LHC: 10-15PB/year

Walmart, Amazon, NYSE...

Economist: mankind produced
1,200 Exabytes (billion GB) in 2010
So much data

- Easy to get
  - Complex, automated sensors
  - Lots of people (transactions, Internet)
- Cheap to keep
  - 2TB disk costs $80 dollars today (4c/GB!)
- Potentially very useful
  - Both for science and business
- Large scale: doesn’t fit in memory
How do we process this data?

- **Current minute-sort world record**
  - External sort: 2 reads, 2 writes from/to disk (optimal)
  - TritonSort (UCSD), ~1TB/minute
- **One disk: say 2TB, 100MB/s (sequential)**
  - how long to read it once?
  - ~ 2,000,000MB/100MB/s ~ 20,000s ~ 5.5h!
  - TritonSort: 52 nodes X 16 disks
- **Latency: say processing 1 item takes 1ms**
  - CPUs aren’t getting much faster lately
  - Only 1000 items per second!

- **Have to use many machines**
Challenges

• Traditional parallel supercomputers are not the right fit for many problems (given their cost)
  – Optimized for fine-grained parallelism with a lot of communication
  – Cost does not scale linearly with capacity

• => Clusters of commodity computers
  – Even more accessible with pay-as-you-go cloud computing
Parallel computing is hard!

Fundamental issues
- scheduling, data distribution, synchronization, inter-process communication, robustness, fault tolerance,

Different programming models
- Message Passing
- Shared Memory

Architectural issues
- Flynn’s taxonomy (SIMD, MIMD, etc.), network typology, bisection bandwidth
- UMA vs. NUMA, cache coherence

Different programming constructs
- mutexes, conditional variables, barriers,
- masters/slaves, producers/consumers, work queues,

Common problems
- livelock, deadlock, data starvation, priority inversion...
- dining philosophers, sleeping barbers, cigarette smokers,

Architectural issues
- Flynn’s taxonomy (SIMD, MIMD, etc.), network typology, bisection bandwidth
- UMA vs. NUMA, cache coherence

Different programming constructs
- mutexes, conditional variables, barriers,
- masters/slaves, producers/consumers, work queues,

The reality: programmer shoulders the burden of managing concurrency...

Slide from Jimmy Lin, University of Maryland
DISC Main Ideas

• Scale “out”, not “up”  
  – Se Barroso and Hölze, Chapter 3
• Assume failures are common  
  – Probability of “no machine down” decreases rapidly with scale…
• Move processing to the data  
  – Bandwidth is scarce
• Process data sequentially  
  – Seeks are *very* expensive
• Hide system-level details from the application developer
Examples

• **Google’s MapReduce**
  – Many implementations: most adopted is Hadoop
  – Used in many applications

• **Many extensions**
  – Dryad, CIEL, iterative versions

• **Other frameworks**
  – Graphlab, Piccolo, TritonSort
This course

• **We will study current DISC solutions**
  – Systems challenges
  – Programming models
  – Dealing with failures

• **We will look at several applications**
  – Information retrieval, data mining, graph mining,
    computational biology, finance, machine learning, …

• **Possibly**
  – Identify shortcomings, limitations
  – Address these!
Course Mechanics

• 3 Components
  – Reading papers (systems and applications)
    • Reviews of assigned papers (everyone)
    • Lead discussion in class (one or two people per paper)
  – A few programming assignments
    • For example, PageRank on a collection of documents
    • Department cluster, possibly on Amazon Cloud
    • Get your hands dirty
  – A mini research project
    • Systems and/or applications
    • Can be related to your research
    • Preferable in small groups (2 ideal)
Web and Group

- Site will be up later today
- Google group will be where you submit your reviews
  - Up to midnight, day before class
Next class

- Skim Barroso and Hölze, Chapters 1 and 3
- Skim Lin and Dyer, Chapter 1
- Read the MapReduce paper
- No review necessary
- Homework 0
  - [www.cs.brown.edu/courses/csci2950-u/f11/homework0.html](http://www.cs.brown.edu/courses/csci2950-u/f11/homework0.html)