Designing for the Cloud
An Architect’s Perspective

Christos Kozyrakis

Computer Systems Lab
Stanford University
http://csl.stanford.edu/~christos

WIOSCA Workshop – June 2009
Good News – Bad News

The opportunities
- Lots of parallelism (request-level)
- Mostly focused on throughput
  - Latency often limited by WAN
- Room for innovation in SW & HW
  - Assuming SaaS, VMs, data-center in a box, ...

The challenges
- Cost sensitive, consumer systems
  - Commodity components, energy efficiency, ...
- Large distributed systems
  - Large number of nodes, heterogeneity, ~O(1) scaling, ...
- Efficiency across range of evolving apps and scenarios
- Non-stop operation
  - Availability, durability, predictability, ...
So what’s an architect to do?
1. Design a System, not a Chip

- Top-down design of middleware & HW
  - Meaningful metrics and goals
  - Focus on true bottlenecks (e.g. HW vs language efficiency)
  - Use high-level properties & system-wide techniques

- Example: assuming system-wide data replication
  - Focus on per-node error detection instead of error correction
  - Forward requests to avoid latency spikes, thermal issues, ...
  - Energy proportionality by turning nodes on/off
    - E.g. results from energy-efficient Hadoop

---

**Graphs and Diagrams**

- Runtime
- Energy
- Power (kW)
- Cumulative Inactivity Distribution
2. From Compute- to Data-Centric

- Compute is now cheap and efficient
- Focus on memory, storage, and network
  - Throughput, latency, energy, power, ...
  - Rethink arch, HW/SW interface, support for distributed ops
  - Tradeoff compute for communication
3. Rethink Components & Balance

- What are the right components for the system?
  - Server, notebook, or embedded?
  - Best way to connect them and balance their features?

- Example: energy efficient sorting

![Diagram showing energy-aware systems and related scores.](Image)
4. Tools and Methodologies

- In great need of
  - Analysis of workloads and systems
  - Models for emerging apps & architectures
  - Benchmarks (workloads and metrics)
  - Convincing prototyping techniques
  - Understanding of performance, energy, and reliability tradeoffs