Resource Efficient Computing for Warehouse-scale Datacenters

Christos Kozyrakis

Stanford University
http://csl.stanford.edu/~christos

DATE Conference – March 21st 2013
Computing is the Innovation Catalyst

Science
Government
Commerce
Healthcare
Education
Entertainment

Faster, cheaper, greener
The Datacenter as a Computer

- $500M+ investment
- 3000 construction related jobs
- 707,000 sq ft
- 60 MW Total Critical Power
- 1.5 million man-hours-of-labor
- 3400 tons of steel
- 190 miles of conduit
- 2400 tons of copper
- 7.5 miles of chilled water piping
- 26,000 cubic yards of concrete

[K. Vaid, Microsoft Global Foundation Services, 2010]
Advantages of Large-scale Datacenters

- Scalable capabilities for demanding services
  - Websearch, social nets, machine translation, cloud computing
  - Compute, storage, networking

- Cost effective
  - Low capital & operational expenses
  - Low total cost of ownership (TCO)
Datacenter Scaling

- **Cost reduction**
  - Switch to commodity servers *one time trick*
  - Improved power delivery & cooling *PUE < 1.15*

- **Capability scaling**
  - More datacenters *>$300M per DC*
  - More servers per datacenter *@60MW per DC*
  - Multicore servers *End of voltage scaling*
  - Scalable network fabrics
Datacenter Scaling through Resource Efficiency

- Are we using our current resources efficiently?
- Are we building the right systems to begin with?
Our Focus: Server Utilization

Servers dominate datacenter cost
- CapEx and OpEx

Server resources are poorly utilized
- CPUs cores, memory, storage

Total Cost of Ownership

- Servers: 61%
- Energy: 16%
- Cooling: 14%
- Networking: 6%
- Other: 3%

Server utilization


[U. Hoelzle and L. Barosso, 2009]
Low Utilization

Primary reasons
- Diurnal user traffic & unexpected spikes
- Planning for future traffic growth
- Difficulty of designing balanced servers

Higher utilization through workload co-scheduling
- Analytics run on front-end servers when traffic is low
- Spiking services overflow on servers for other services
- Servers with unused resources export them to other servers
  - E.g., storage, Flash, memory

So, why hasn’t co-scheduling solved the problem yet?
Interference \(\rightarrow\) Poor Performance & QoS

- Interference on shared resources
  - Cores, caches, memory, storage, network
  - Large performance losses
    - E.g. 40% for Google apps [Tang’11]

- QoS issue for latency-critical applications
  - Optimized for low 99th percentile latency in addition to throughput
  - Assume 1% chance of >1 sec server latency, 100 servers used per request
  - Then 63% chance of user request latency >1 sec

- Common cures lead to poor utilization
  - Limited resource sharing
  - Exaggerated reservations
Higher Resource Efficiency wo/ QoS Loss

- Research agenda
  - Workload analysis
    - Understand resource needs, impact of interference
  - Mechanisms for interference reduction
    - HW & SW isolation mechanisms (e.g., cache partitioning)
  - Interference-aware datacenter management
    - Scheduling for min interference and max resource use
  - Resource efficient hardware design
    - Energy efficient, optimized for sharing

- Potential for >5x improvement in TCO
Datacenter Scheduling

Two obstacles to good performance

- Interference: sharing resources with other apps
- Heterogeneity: running on suboptimal server configuration
Paragon: interference-aware Scheduling [ASPLOS’13]

- Quickly classify incoming apps
  - For heterogeneity and interference caused/tolerated
- Heterogeneity & interference aware scheduling
  - Send apps to best possible server configuration
  - Co-schedule apps that don’t interfere much
- Monitor & adapt
  - Deviation from expected behavior signals error or phase change
Fast & Accurate Classification

- Cannot afford to exhaustively analyze workloads
  - High churn rates of evolving and/or unknown apps
- Classification using collaborative filtering
  - Similar to recommendations for movies and other products
  - Leverage knowledge from previously scheduled apps
  - Within 1min of sparse profiling we can estimate
    - How much interference an app causes/tolerates on each resource
    - How well it will perform on each server type
Paragon Evaluation

- 5K apps on 1K EC2 instances (14 server types)
Paragon Evaluation

- Better performance with same resources
- Most workloads within 10% of ideal performance
Paragon Evaluation

- Better performance with same resources
  - Most workloads within 10% of ideal performance
  - Can serve additional apps without the need for more HW
High Utilization & Latency-critical Apps

- Example: scheduling work on underutilized memcached servers
  - Reporting QPS at cutoff of 500usec for 95th % latency

- High potential for utilization improvement
  - All the way to 100% CPU utilization impact QoS impact

- Several open issues
  - System configuration, OS scheduling, management of hardware resources
Datacenter Scaling through Resource Efficiency

- Are we using our current resources efficiently?

- Are we building the right systems to begin with?
Main Memory in Datacenters

- Server power main energy bottleneck in datacenters
  - PUE of ~1.1 → the rest of the system is energy efficient
- Significant main memory (DRAM) power
  - 25-40% of server power across all utilization points
  - Low dynamic range → no energy proportionality

[U. Hoelzle and L. Barosso, 2009]
**DDR3 Energy Characteristics**

- DDR3 optimized for high bandwidth (1.5V, 800MHz)
  - On chip DLLs & on-die-termination lead to high static power
  - 70pJ/bit @ 100% utilization, 260pJ/bit at low data rates

- LVDDR3 alternative (1.35V, 400MHz)
  - Lower Vdd → higher on-die-termination
  - Still disproportional at 190pJ/bit

- Need memory systems that consume lower energy and are proportional
  - What metric can we trade for efficiency?
Memory Use in Datacenters

Resource Utilization for Microsoft Services under Stress Testing [Micro’11]

<table>
<thead>
<tr>
<th></th>
<th>CPU Utilization</th>
<th>Memory BW Utilization</th>
<th>Disk BW Utilization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Large-scale analytics</td>
<td>88%</td>
<td>1.6%</td>
<td>8%</td>
</tr>
<tr>
<td>Search</td>
<td>97%</td>
<td>5.8%</td>
<td>36%</td>
</tr>
</tbody>
</table>

- Online apps rely on memory capacity, density, reliability
  - But not on memory bandwidth
  - Web-search and map-reduce
    - CPU or DRAM latency bound, <6% peak DRAM bandwidth used
  - Memory caching, DRAM-based storage, social media
    - Overall bandwidth by network (<10% of DRAM bandwidth)

- We can trade off bandwidth for energy efficiency
Mobile DRAMs for Datacenter Servers [ISCA’12]

- Same core, capacity, and latency as DDR3
- Interface optimized for lower power & lower bandwidth (\(1/2\))
  - No termination, lower frequency, faster powerdown modes
- Energy proportional & energy efficient
Mobile DRAMs for Datacenter Servers [ISCA’12]

- **LPDDR2 module**: die stacking + buffered module design
  - High capacity + good signal integrity
- **5x reduction in memory power, no performance loss**
  - Save power or increase capability in TCO neutral manner
- **Unintended consequences**
  - Energy efficient DRAM → L3 cache power now dominates
Summary

Resource efficiency
- A promising approach for scalability & cost efficiency
- Potential for large benefits in TCO

Key questions
- Are we using our current resources efficiently?
  - Research on understanding, reducing, and managing interference
  - Hardware & software
- Are we building the right systems to begin with?
  - Research on new compute, memory, and storage structures