SoftSKU: Optimizing Server Architectures for Microservice Diversity @Scale

Akshitha Sriraman
Abhishek Dhanotia
Thomas F. Wenisch

Rapid Increase in Modern Web Services

Client

Web

Feed

Ads

Cache

Stringent Service-Level Objectives + Moore’s law decline

μservices SLOs + Moore’s law decline -> need for customized hardware
Are Customized Platforms Always Needed?

Customized platforms -> expensive

Hardware resource fungibility

Procurement @scale

Testing overhead

Urgent need for limited SKUs that support a variety of microservices
Performance of Commodity Servers

Maintain commodity hardware for procurement efficiency & scalability

How performant is commodity hardware for these μservices?

Do μservices have common bottlenecks that can inspire future CPU designs?
Contributions

- Comprehensive characterization of Facebook’s microservices
  - System-level and architectural bottlenecks
  - Enormous bottleneck diversity across microservices

- Concept of “soft” server SKUs
  - Tuning coarse-grained OS & hardware configuration knobs

- μSKU
  - Automates soft-SKU search & configuration via production AB-tests
  - Deploys soft SKUs on production microservices

~7.2% perf. boost on production μservices + no extra hardware
Introduction: Commodity servers AND performance?

Characterization: System-level & architectural understanding of key FB μservices

Soft SKU: Achieving data center performance- & cost-efficiency

μSKU: Tool to automate Soft SKU creation

Evaluation: 7% better performance on production systems
Introduction: Commodity servers AND performance?

Characterization: System-level & architectural understanding of key FB μservices

Soft SKU: Achieving data center performance- & cost-efficiency

μSKU: Tool to automate Soft SKU creation

Evaluation: 7% better performance on production systems
### Facebook’s Production Microservices

<table>
<thead>
<tr>
<th>μService</th>
<th>Throughput (QPS)</th>
<th>Response Latency</th>
<th>Pathlength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web</td>
<td>$O(100)$</td>
<td>$O(ms)$</td>
<td>$O(10^6)$</td>
</tr>
<tr>
<td>Feed1</td>
<td>$O(1000)$</td>
<td>$O(ms)$</td>
<td>$O(10^9)$</td>
</tr>
<tr>
<td>Feed2</td>
<td>$O(10)$</td>
<td>$O(s)$</td>
<td>$O(10^9)$</td>
</tr>
<tr>
<td>Ads1</td>
<td>$O(10)$</td>
<td>$O(ms)$</td>
<td>$O(10^9)$</td>
</tr>
<tr>
<td>Ads2</td>
<td>$O(100)$</td>
<td>$O(ms)$</td>
<td>$O(10^9)$</td>
</tr>
<tr>
<td>Cache1</td>
<td>$O(100K)$</td>
<td>$O(\mu s)$</td>
<td>$O(10^3)$</td>
</tr>
<tr>
<td>Cache2</td>
<td>$O(100K)$</td>
<td>$O(\mu s)$</td>
<td>$O(10^3)$</td>
</tr>
</tbody>
</table>

- **Feed2**: 2 s, 100 µs, 2.000002 s, 120 µs, 20%!
- **Cache1**: 2 s, 100 µs, 20 µs, 20 µs, 20%

Diversity in service traits causes diversity in bottlenecks faced.
Context Switch Penalty

- Up to 23% context switch penalty

Cache can benefit from context switches/switching delay reductions
Several microservices face much higher front-end stalls
Web incurs an unusually high LLC code MPKI
More Characterizations

More details in the paper
Characterization Takeaways

Can we achieve perf. efficiency without building customized hardware?

Diversity in bottlenecks across microservices

Custom CPU SKUs

Prohibitively expensive
Introduction: Commodity servers AND performance?

Characterization: System-level & architectural understanding of key FB μservices

Soft SKU: Achieving data center performance- & cost-efficiency

μSKU: Tool to automate Soft SKU creation

Evaluation: 7% better performance on production systems
“Soft” SKUs: Best of Both Worlds

Tune coarse HW & OS knobs on commodity HW

Performance efficiency

Procurement efficiency & scalability

Core Frequency
Uncore Frequency
Core Count
CDP: LLC
Prefetcher
Transparent Huge Pages (THP)
Statically-Allocated Huge Pages (SHP)

Tuning configurable OS & HW knobs for each microservice may improve perf. efficiency & scalability
Tuning Configurable OS & CPU Knobs

Manual tuning: Not scalable

Complex design space
- Core freq.
- Uncore freq.
- Prefetchers
- Core count
- THP
- SHP
- CDP

Quick code evolution

Synthetic load tests don’t capture production load

Small knob tuning effects

Need for an automated design & deployment tool to create soft SKUs
Introduction: Commodity servers AND performance?

Characterization: System-level & architectural understanding of key FB μservices

Soft SKU: Achieving data center performance- & cost-efficiency

μSKU: Tool to automate Soft SKU creation

Evaluation: 7% better performance on production systems
µSKU: Tool to Design & Deploy Soft SKUs

Design Phase

µSKU picks a server running the µservice

Prepare server for the tuning experiment

! Prefetch
Web
Performs A/B testing

? Préfécher
Web
Prefetch

> Préfécher
Web
Prefetch

Varies a tunable knob

Web
μSKU: Tool to Design & Deploy Soft SKUs

**Deployment Phase**

<table>
<thead>
<tr>
<th>Knob</th>
<th>Best config</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core freq.</td>
<td>2.2 GHz</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Prefetcher</td>
<td>on</td>
</tr>
</tbody>
</table>

Selects performant knob configs.

Applies config. to production servers

Further A/B testing

A/B tester’s design space map

Soft SKU generator
Introduction: Commodity servers AND performance?

Characterization: System-level & architectural understanding of key FB µservices

Soft SKU: Achieving data center performance- & cost-efficiency

µSKU: Tool to automate Soft SKU creation

Evaluation: 7% better performance on production systems
Methodology

- \( \mu \text{SKU} \) evaluated on Web & Ads1
  - Web evaluated on two platforms: Skylake & Broadwell

- Soft SKUs compared against:
  - Stock and hand-tuned production server configurations
Web (Broadwell) improves when prefetching is off: Better BW util.
Soft SKU can achieve ~7.2% throughput improvement on production systems with no extra hardware requirement
Conclusion

Hardware resource fungibility + performance efficiency

Soft SKUs can achieve perf. improvements on existing hardware

FB’s microservices are significantly diverse

μSKU achieves ~7.2% throughput gains on production systems
SoftSKU: Optimizing Server Architectures for Microservice Diversity @Scale

Abstract

The variety and complexity of microservices in warehouse-scale data centers has given rise to a need for more flexible and scalable server architectures. SoftSKU is a new approach that allows for the dynamic reconfiguration of server architectures to support diverse microservice workloads. This enables organizations to optimize their infrastructure for varying workloads and business needs.

By Akshitha Sriraman, Abhishek Dhanotia, and Thomas F. Wenisch