Yukta: Multilayer Resource Controllers to Maximize Efficiency

Raghavendra Pradyumna Pothukuchi
Sweta Yamini Pothukuchi
Petros Voulgaris
Josep Torrellas

http://iacoma.cs.uiuc.edu/

International Symposium on Computer Architecture (ISCA) 2018
Resource Management in Computers

Configurable parameters

Limited resources
Energy
Storage

Many objectives
Performance
Thermals
Fairness

Resource Management System
Designing Computer Resource Management Systems

Different resources, parameters and design teams!

Modular

Coordinated

Handle uncertainty
Contributions of This Work

▪ Propose Robust Control Theory for computer resource management
  – Describe why robust control theory is essential for computers

▪ Develop Yukta, a modular approach for coordinated multilayer control
  – Present how robust control theory is applied to multilayer resource management

▪ Prototype Yukta on an ARM big.LITTLE octacore running Ubuntu
  – 37% better performance and 20% lower energy than state-of-the-art
Why Robust Control Theory is Suitable for Computers? - I

- Robust control theory: Branch of control theory for uncertain environments

- Designers set a guardband for the amount of uncertainty to tolerate
  - 60% guardband ⇒ controller’s guarantees hold even if model is off by 60%

- Computer control faces many forms of uncertainty
  - Partial system view, controller interference, program behavior, limited modeling…

Robustness to uncertainty enables optimality under modularity

System

Uncertainty

Model
Why Robust Control Theory is Suitable for Computers? - II

- Robust controllers have input channels for communication

We use Externals signals for controller communication
Why **Robust Control Theory** is Suitable for Computers? - III

- Can work with discretized inputs as common in computers
  - E.g., core frequency, \( f \in \{2 \text{ GHz, } 2.2 \text{ GHz } ... 3 \text{ GHz}\} \)

- Guarantee precise bounds on meeting targets
  - E.g., core power can be kept within \( \pm 0.05 \text{ W} \) of the power target

**Robust control theory is highly desirable for computer systems**
Yukta: Multilayer Robust Controllers

Modularly designed and **guaranteed** optimal behavior

**First** to propose robust control for modular multilayer management
Automated Synthesis of the Robust Controller

- Designer provides a model, and:
  - Output deviation bounds ($B$): e.g., ±0.05W for power deviations
  - Input weights ($W$): Relative overheads
  - Discrete input values ($Δ↓in$)
  - Uncertainty guardbands ($Δ↓u$): Degree of desired robustness
Automated Synthesis of the Robust Controller

- Designer provides a model, and:
  - Output deviation bounds ($B$): e.g., $\pm 0.05W$ for power deviations
  - Input weights ($W$): Relative overheads
  - Discrete input values ($\Delta \text{in}$)
  - Uncertainty guardbands ($\Delta \text{in}$): Degree of desired robustness

**Structured Singular Value $\text{SSV}(N\Delta)$**

Can the controller guarantee the bounds optimally even under uncertainty as big as the guardband?

**Synthesis is automated!**
Designing Multilayer Controllers

1. Select inputs, outputs
2. Decide external signals
3. Obtain model
4. Set controller parameters
5. Design controller

Layer 1, e.g. OS team
Layer 2, e.g. Hardware team

Interface

Modular and practical design
Prototype Yukta on a Challenging System

Minimize Energy×Delay under thermal and power constraints
Prototyped Yukta Architecture
Designing **Robust Controllers** for the Prototype

- Select inputs, outputs
- Decide External signals
- Obtain model
- Set controller parameters
- Design controller

Data driven modeling (System Identification):
2 benchmarks each from SPECint, SPECfp and PARSEC

- Output deviation bounds
- Input weights
- Input discretization
- Uncertainty guardband

OS Robust Controller

- \#tasks_{big}
- Avg.\#tasks per nonidle core_{little}
- Avg.\#tasks per nonidle core_{big}

HW Robust Controller

- frequency_{little}
- frequency_{big}
- \#cores_{little}
- \#cores_{big}

Application tasks

<table>
<thead>
<tr>
<th>Little</th>
<th>Big</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>B</td>
</tr>
<tr>
<td>L</td>
<td>B</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
</tr>
</tbody>
</table>

Ubuntu 15.04

Performance_{little}

Power_{little}

Temperature

Performance_{big}

Power_{big}
**Evaluation**

Industry based
- OS heuristic
- HW heuristic

[ISCA’16]
- Monolithic non-robust
- OS Robust
- HW Robust

Coordinated heuristics

Monolithic non-robust

Yukta: Multilayer robust

Average across 8 PARSEC and 6 SPEC (multiprogrammed) workloads

Yukta makes the system 37% faster with 20% lower energy ⇒ 50% better ED

![Normalized execution time](chart1)

![Normalized Energy x Delay](chart2)

15
Summary

- Dynamic resource management must meet many goals simultaneously
- Resource controllers should be modular and coordinated
- We propose using Robust Control Theory for computer management
  – Optimized for uncertainty
- We develop Yukta for formal multilayer resource management
- Prototype demonstrates significant advancement
  – 50% reduced Energy×Delay on average
- Yukta is essential for resource efficient computing
Concluding Remarks

- More details in the paper
  - Mathematical background of the Robust Controller
  - Yukta design details
  - Prototype design choices
  - Comparison with other designs and heterogeneous workloads
  - Sensitivity analysis

- Future work
  - Heterogeneous components in a layer
  - Hierarchical organization of controllers
Yukta: Energy×Delay minimization
Yukta: Energy $\times$ Delay minimization
Case Study with *blackscholes*

(a) Power of little cluster  
(b) Power of big cluster  
(c) Temperature  
(d) Application performance
Yukta: blackscholes

(a) Coordinated heuristic
(b) Decoupled heuristic
(c) Yukta: HW SSV+OS heuristic
(d) Yukta: HW SSV+OS SSV
Yukta: Comparison with LQG control
Yukta: Heterogeneous Workloads
Yukta: Sensitivity to Uncertainty Guardbands

(a) Normalized output bounds.  
(b) E×D.
Yukta: Sensitivity to Output Bounds

(a) Tracking 5.5BIPS of performance.  
(b) Minimizing E×D.
Yukta: Sensitivity to Input Weights