Just Enough is More: Achieving Sustainable Performance in Thermally-Constrained Mobile Devices

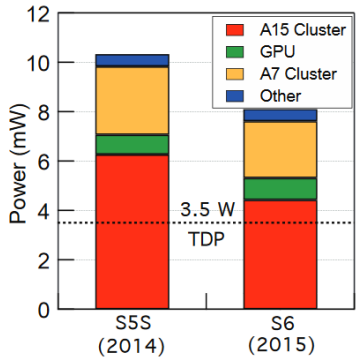
Ayse K. Coskun

Department of Electrical and Computer Engineering
Performance and Energy Aware Computing Laboratory
www.bu.edu/peaclab

BOSTON UNIVERSITY

Mobile Devices: Trends & Thermal Challenges

- SoC power densities have grown significantly
  - Mobile CPUs adopt aggressive µArch design, clock speeds.[*]
  - Single-thread CPU performance have improved by 3x-11x over generations *[Halpern et al. HPCA'16].*
  - Integrated GPUs further elevate power densities.
- Modern smartphones are thermally constrained
  - Chip and skin temperatures elevate to critical levels.
  - On/off-chip thermal couplings *[Xie et al., ICCAD'13][Prakash et al., DAC'15]*
  - Power and form-factor restrictions limit cooling capabilities.
- Thermal throttling
- Unsustainable user experience over extended application use.[**]



Power and TDP of S5/S6
[Halpern et al. HPCA'16]

[*] http://www.arm.com/files/pdf/AT-Exploring_the_Design_of_the_Cortex-A15.pdf
[**] http://arstechnica.com/gadgets/2013/11/when-benchmarks-arent-enough-cpu-performance-in-the-nexus-5

In our work …

- ✓ We aim at providing **sustainable performance** and propose **thermally-efficient QoS management**.

- ✓ We demonstrate and analyze the performance impacts of **multiple thermal constraints** of modern mobile devices.
- ✓ We present **runtime techniques** for improving thermal efficiency:
  - ✓ Fine-grained DVFS scheduling
  - ✓ Criticality-driven scheduling
  - ✓ CPU-GPU thermal coupling aware runtime
- ✓ We evaluate these techniques on mobile platforms
  - ✓ with **homogeneous** and **heterogeneous** multi-core CPUs,
  - ✓ under **CPU and skin temperature** constraints,

  and achieve up to 8x longer durations of extended sustainable performance.

Outline

- **Experimental Setup & Methodology**
- Thermal Constraints in Smartphones
- Techniques for Sustainable Performance
- Evaluation Results
- Summary

Experimental Setup & Methodology

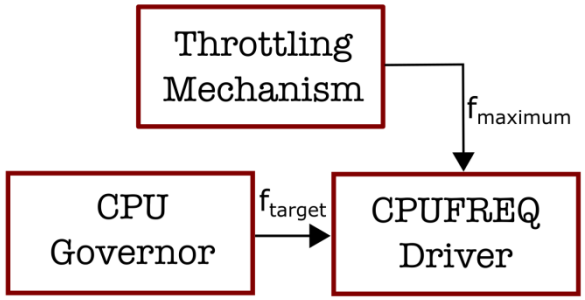
Platforms:

Device	Nexus 5 and Qualcomm MDP8974	Odroid-XU3
SoC	Qualcomm Snapdragon 800	Samsung Exynos 5422
CPU	Krait 400	ARM A15 + A7
Cores	4	4 + 4
CPU Freq.	2.2 GHz	2.1 GHz + 1.5 GHz
GPU	Adreno 330	Mali-T628
GPU Freq.	450 MHz	543 MHz

Applications:

- *Scimark* (FFT, SOR);
- *SPEC CPU 2006* (H264);
- *PARSEC* (bodytrack)
- *Gaming* (Edge of Tomorrow, Real Racing);
- *WebGL* (Aquarium, Pearl Boy, Rain);
- *Video player apps* (Mx Player, Rock Player);

- CPU & skin temperature control policies.
  - 90°C and 40°C thermal limits, respectively.
- Policy implementations:
  - cpufreq interface for frequency scaling
  - sched_setaffinity for thread-to-core mapping

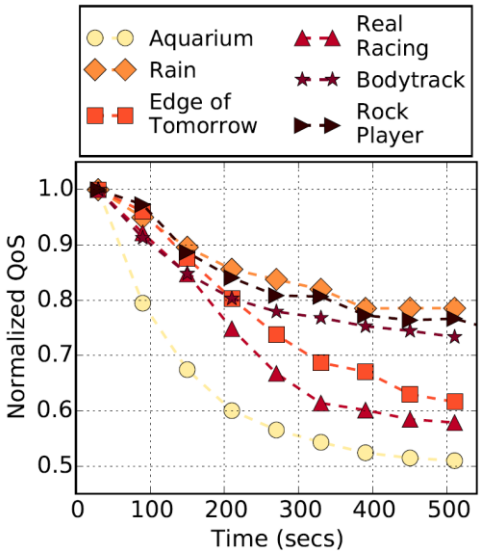


Outline

- Experimental Setup & Methodology
- **Thermal Constraints in Smartphones**
- Techniques for Sustainable Performance
- Evaluation Results
- Summary

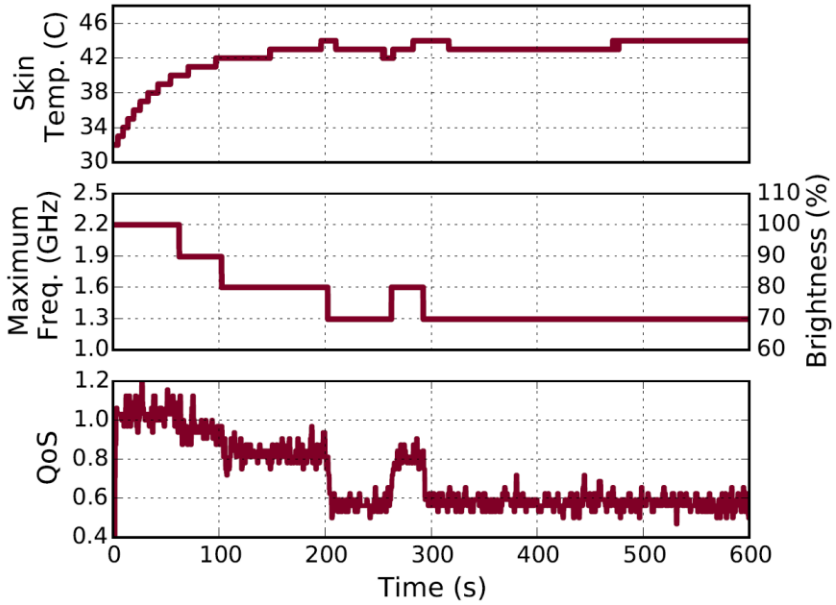
Thermal Constraints in Modern Smartphones

- CPU temperature induced throttling largely degrades QoS.



QoS degradation over time on Odroid-XU3 due to CPU thermal limits *[Sahin et al., ICCAD'16]*

- Modern smartphones are also constrained by skin temperature levels (e.g., 34°C-43°C *[Egilmez et al., DATE'15]*).

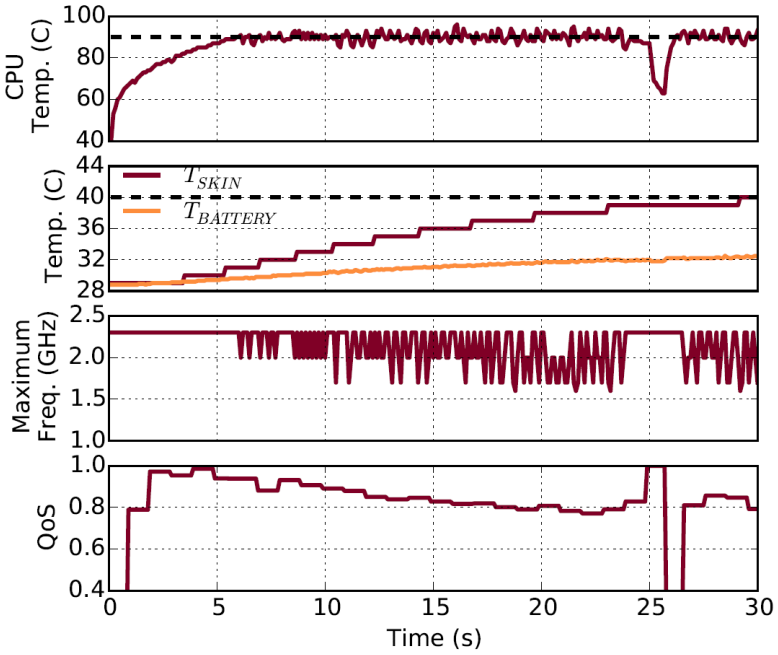


QoS degradation over time on Nexus 5 due to skin temperature violation while running EoT.

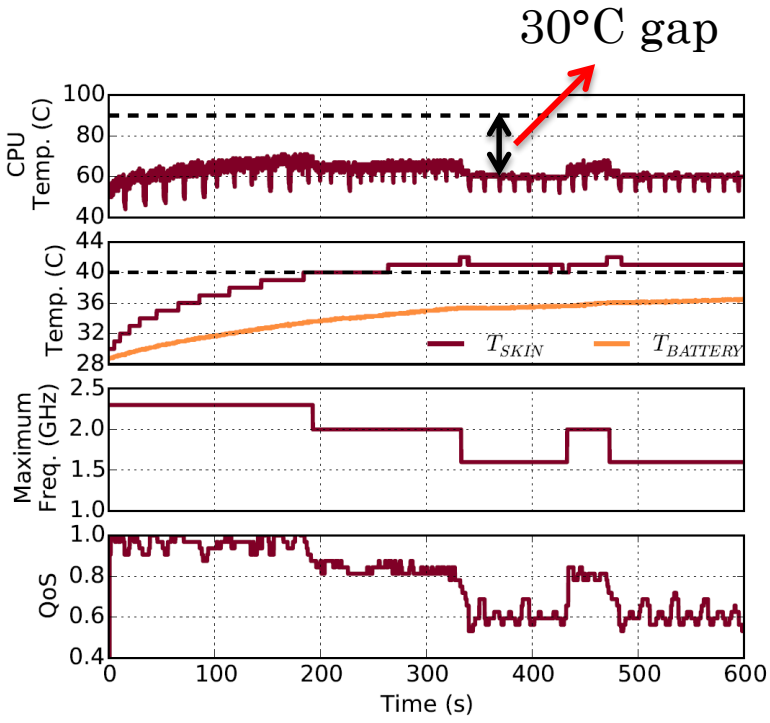
Users will experience significant performance loss when the device is used for extended durations (e.g., gaming, streaming etc.)

Thermal Constraints in Modern Smartphones

- CPU vs. skin level thermal constraints.



(a) FFT application

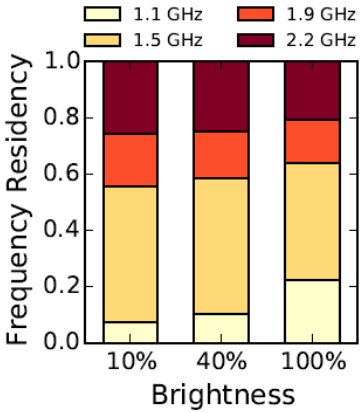


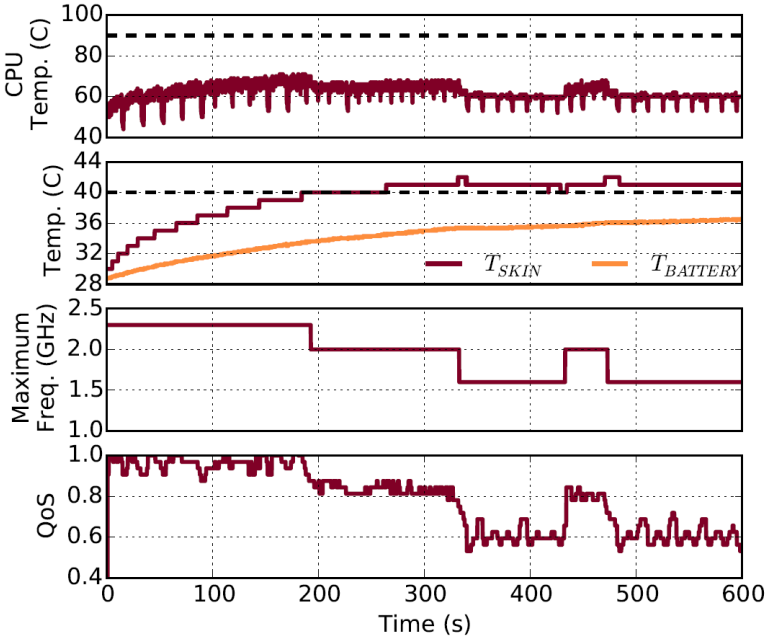
(b) MX Player application

- Throttling CPU to lower skin temperature can lead to large waste in CPU thermal headroom.

Thermal Constraints in Modern Smartphones

- Platform-level thermal management using additional knobs (e.g., display).
- CPU can better utilize its headroom.





(b) MX Player application

- Current scope of our work focuses on CPU level control knobs.

We propose QoS-centric thermal management to achieve sustained performance and provide novel observations to improve thermal efficiency.

Outline

- Experimental Setup & Methodology
- Thermal Constraints in Smartphones
- **Techniques for Sustainable Performance**
- Evaluation Results
- Summary

Providing QoS-Temperature Tradeoff

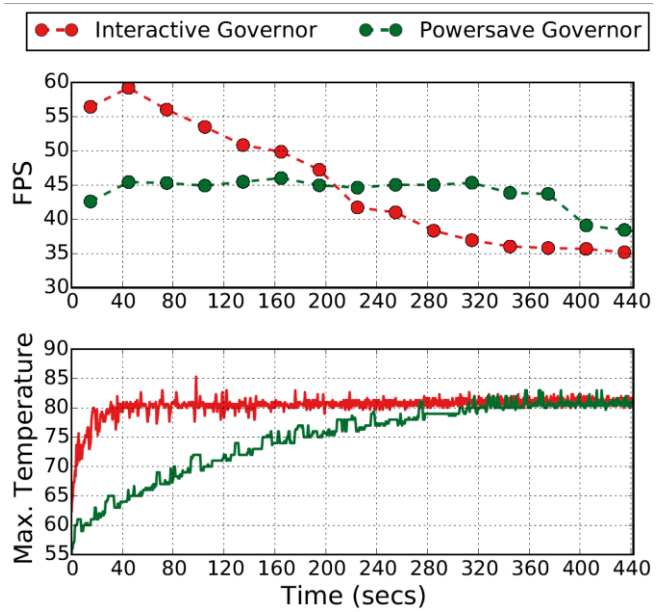
Trading off short term performance for sustainable performance.

Traditional approach:

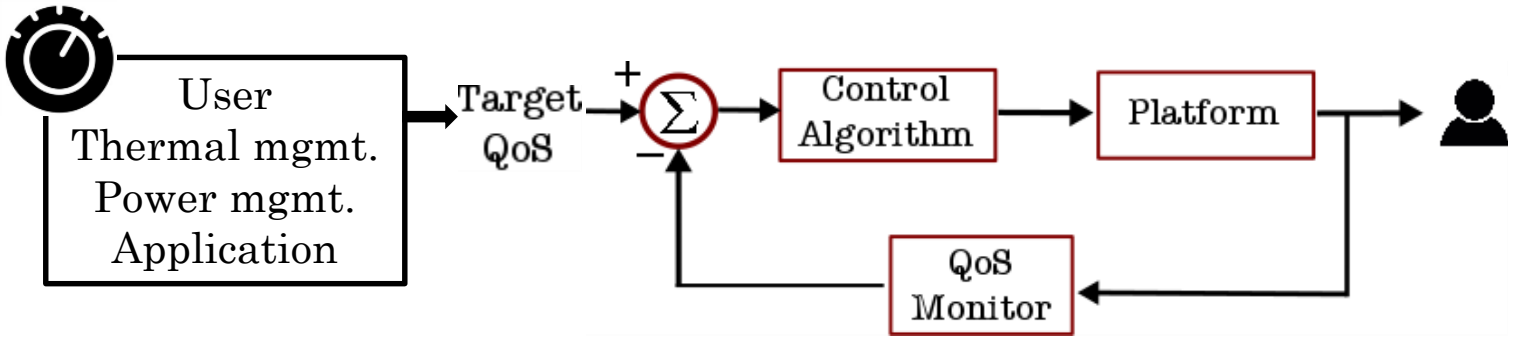
- Maximizing performance under thermal limits.
- Unsustainable performance if apps run longer.

Our approach:

- Limit short-term performance to a **"just enough"** level.
- Extend sustainability of acceptable QoS levels



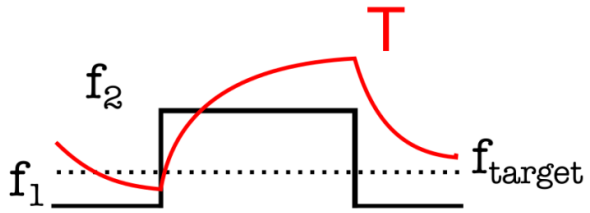
FPS and temperature traces for the Real Racing game on Odroid-XU3 *[Sahin et al., ICCAD'16]*



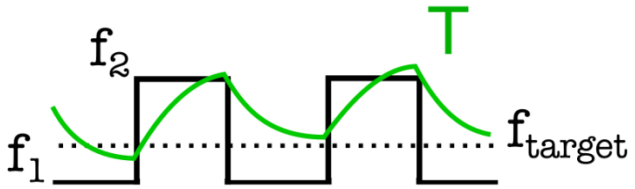
Fine-grained DVFS State Scheduling

Fine-grained, thermally-efficient scheduling of discrete DVFS states.

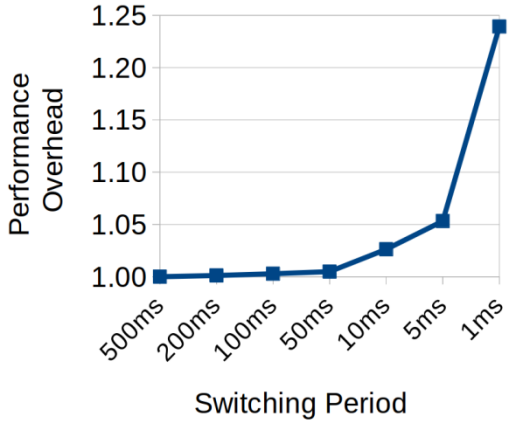
Key idea: Divide high frequency interval into fine-grained bins and distribute temporally while achieving same average frequency.



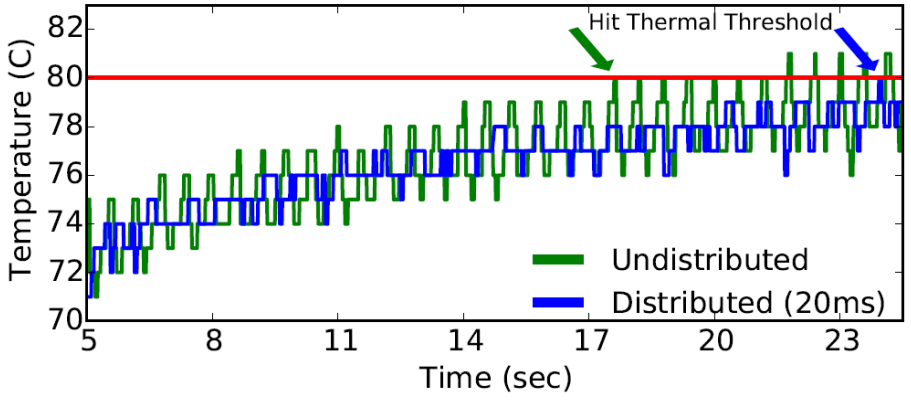
vs.



Limitations due to DVFS overhead:



Real-life demonstration:



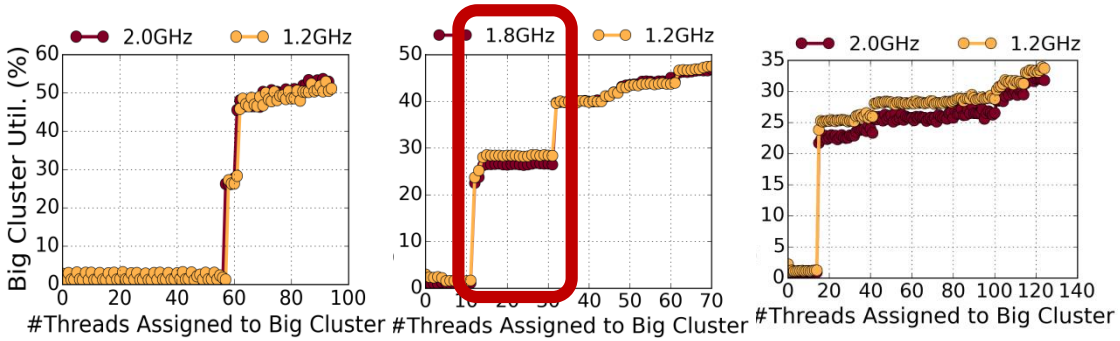
Temperature traces for two DVFS schedules

[Sahin et al., ICCAD'15]

Exploiting HW/SW Heterogeneity for Thermal Efficiency

Identifying and leveraging per-thread criticality in scheduling.

QoS scaling for RP (left) EoT (middle) Rain (right)

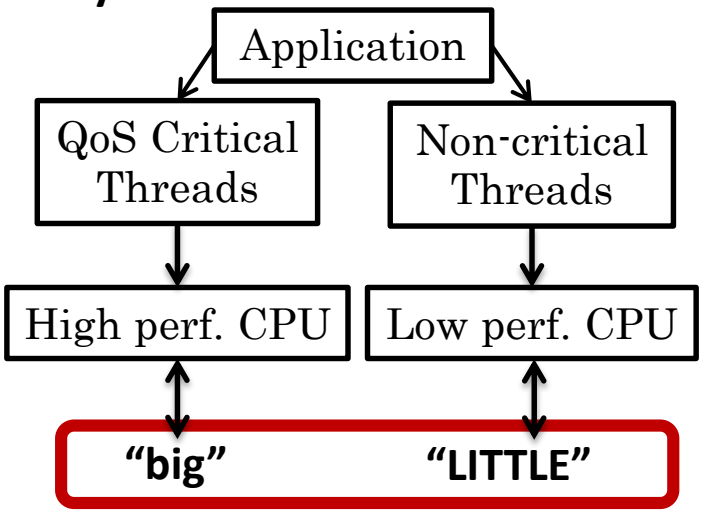


CPU utilization increase for RP (L) Rain (M) RR (R)

- Experiments on Odroid-XU3 (left) *[Sahin, ICCAD'16]*
- Non-critical threads increase CPU util.

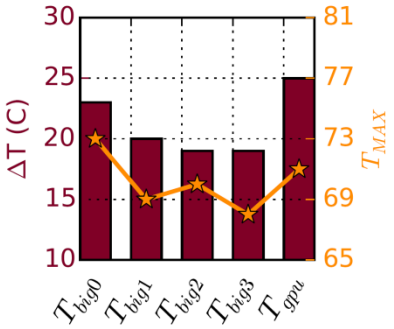
  - Accelerates heating.

Key idea:



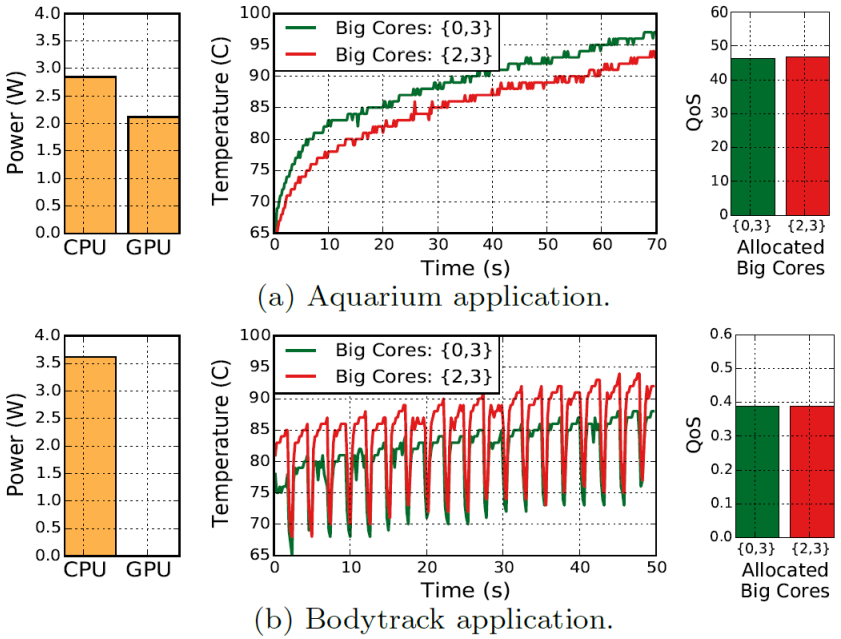
CPU-GPU Thermal Coupling Aware Runtime Management

Identifying thermally-efficient cores based on CPU-GPU thermal couplings.



CPU-GPU thermal coupling on Exynos 5 *[Sahin et al., ICCAD'16]*

- Most thermally-efficient CPU cores depend on GPU usage.
  - Application dependent!



Power (L), temperature (M) and QoS (right) *[Sahin et al., ICCAD'16]*

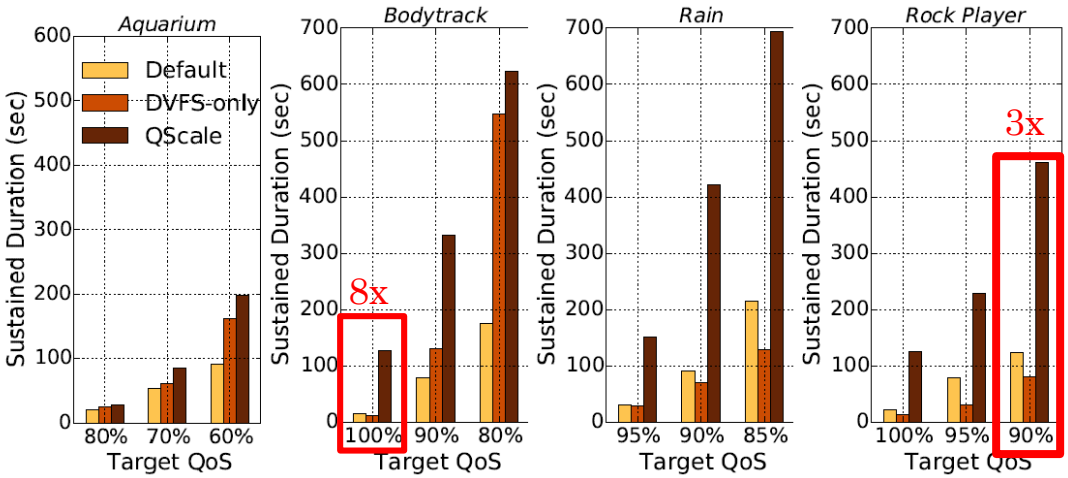
- Offline characterization of thermal coupling via microbenchmarks.
- Varying levels of GPU power -> Record the ordering of cores from the lowest to highest maximum temperature.

Outline

- Experimental Setup & Methodology
- Thermal Constraints in Smartphones
- Techniques for Sustainable Performance
- **Evaluation Results**
- Summary

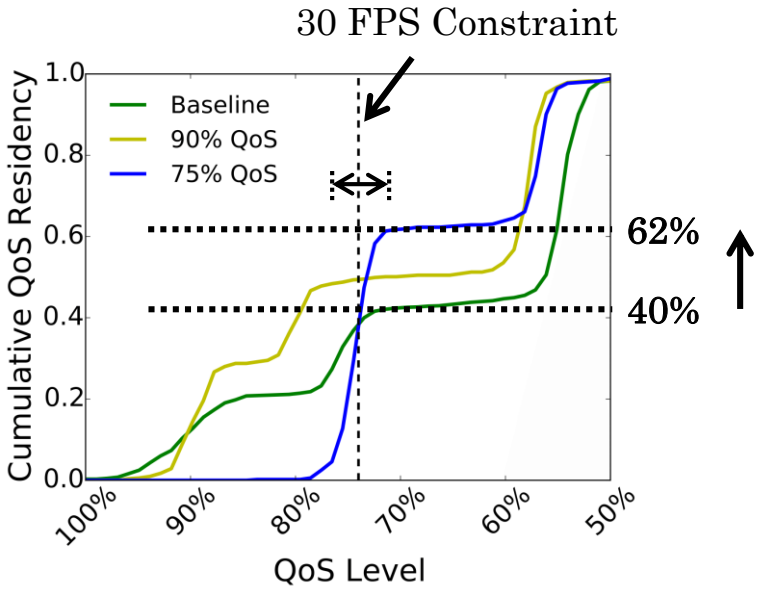
Evaluation: Extended Sustained Duration

Under CPU temperature constraints on Odroid-XU3:



- QScale provides the longest durations of sustainable performance.
- Larger improvements (e.g., up to 8x for *bodytrack*) in *Rain*, *bodytrack*, *Rock Player* due to criticality awareness in QScale.

Under skin temperature constraints on MSM8974:

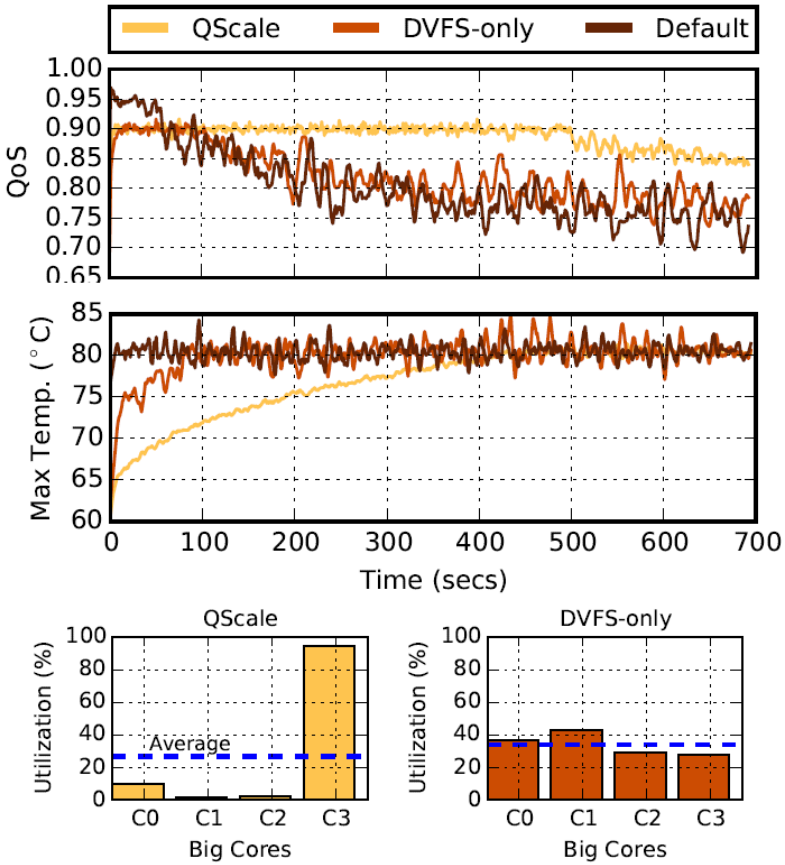


Aquarium

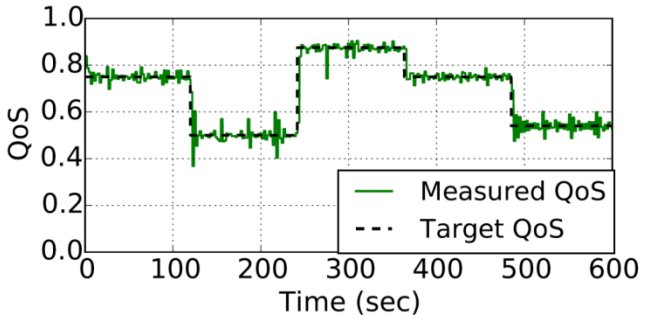
- 55% longer time with acceptable FPS

Evaluation: Runtime Behavior

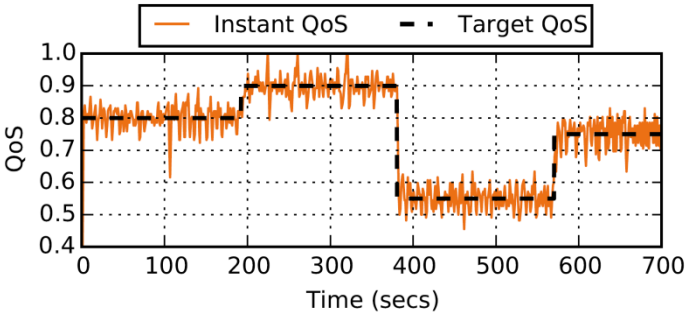
QoS and temperature traces for *Rain* application under 90% target:



Adapting to Dynamic QoS targets:



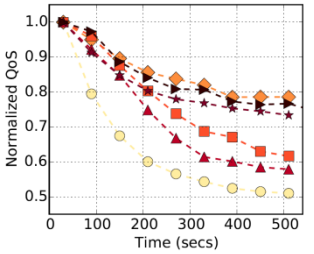
On MSM8974 running *Aquarium*
[Sahin et al., ICCAD'15]

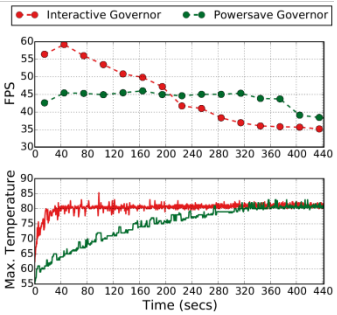


On Odroid-XU3 running *Edge of Tomorrow* *[Sahin et al., ICCAD'16]*

Summary & Takeaways

- Modern mobile devices are constrained by both skin and chip level thermal constraints.
- Throttling leads to unsustainable performance
  - Users expect consistent performance
- Thermally-efficient runtime techniques
  - Reduce temperature
  - Strictly adhere user performance requirements
- Up to 8x longer sustainable performance





BOSTON UNIVERSITY

PEACLAB
Performance and Energy Aware Computing Lab

References:

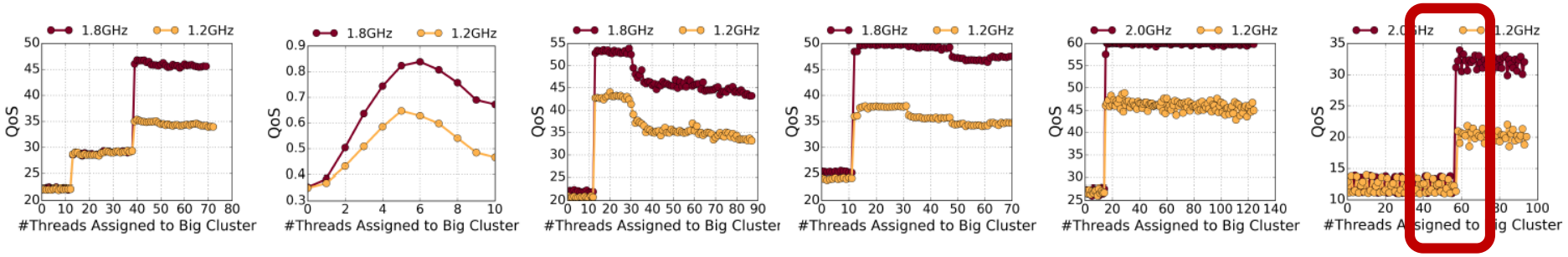
[1] O. Sahin and A.K. Coskun. "**On the Impacts of Greedy Thermal Management in Mobile Devices**.", IEEE *Embedded System Letters*, 2015

[2] O. Sahin, P.T. Varghese and Ayse K. Coskun. "**Just Enough is More: Achieving Sustainable Performance in Mobile Devices under Thermal Limitations**.", *In ICCAD*, 2015.

[3] O. Sahin and Ayse K. Coskun. "**QScale: Thermally-Efficient QoS Management on Heterogeneous Mobile Platforms**", *In ICCAD*, 2016.

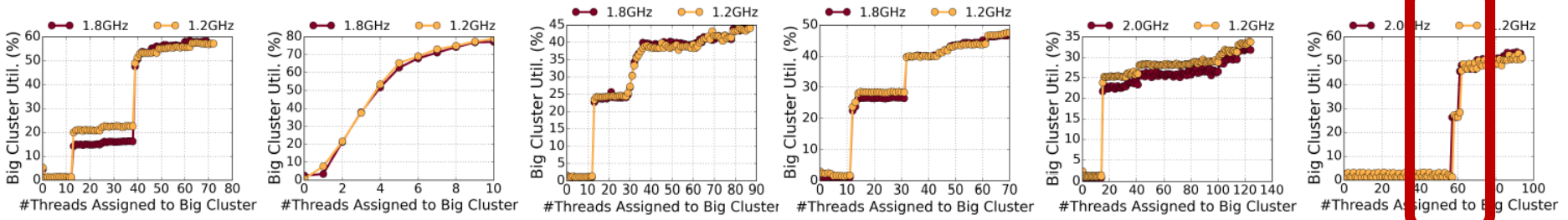
Backup Slides

Thread Criticality in Mobile Applications



(a) Aquarium (b) Bodytrack (c) EoT (d) Rain (e) Racing (f) RP

- QoS reaches to the maximum when only a few critical threads are executed on big cores.

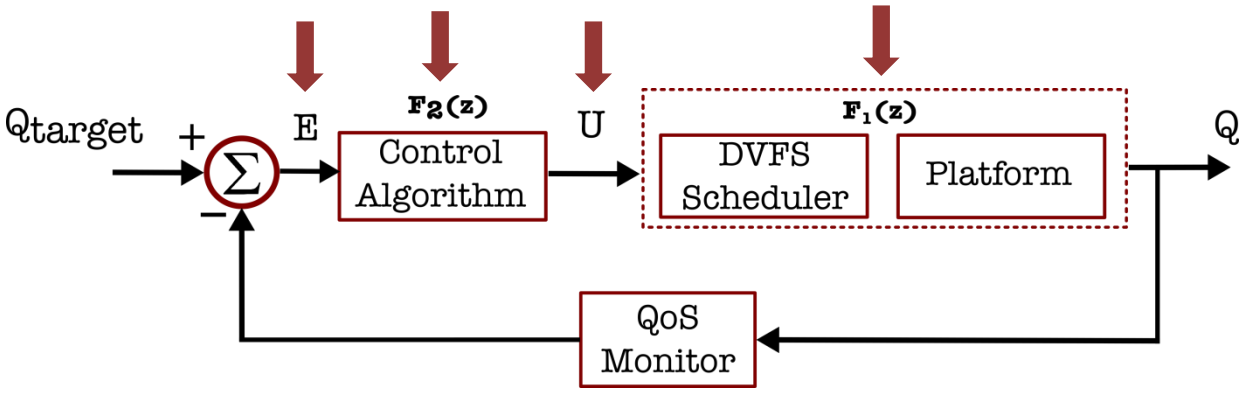


(a) Aquarium (b) Bodytrack (c) EoT (d) Rain (e) Racing (f) RP

- Big cluster utilization can increase when non-critical threads are assigned to big cores.
  - Accelerates heating.

We identify the critical threads of an application offline for runtime mapping of application threads among big and little cores.

Closed-loop QoS Controller Design Details



$$e[k] = Q_{target} - Q[k]$$

$$Q[k+1] = Q_{max} u[k]$$

$$F_1(z) = \frac{Q(z)}{U(z)} = \frac{Q_{max}}{z}$$

$$G(z) = \frac{F_1(z)F_2(z)}{1 + F_1(z)F_2(z)}$$

$$F_2(z) = \frac{U(z)}{E(z)} = \frac{z}{Q_{max}(z-1)}$$

$$u[k+1] = u[k] + e[k]/Q_{max}$$

- ✓ Ensures stable control around Q_{target}
- ✓ Convergences to Q_{target}

Summary of Throttling Results on Nexus 5

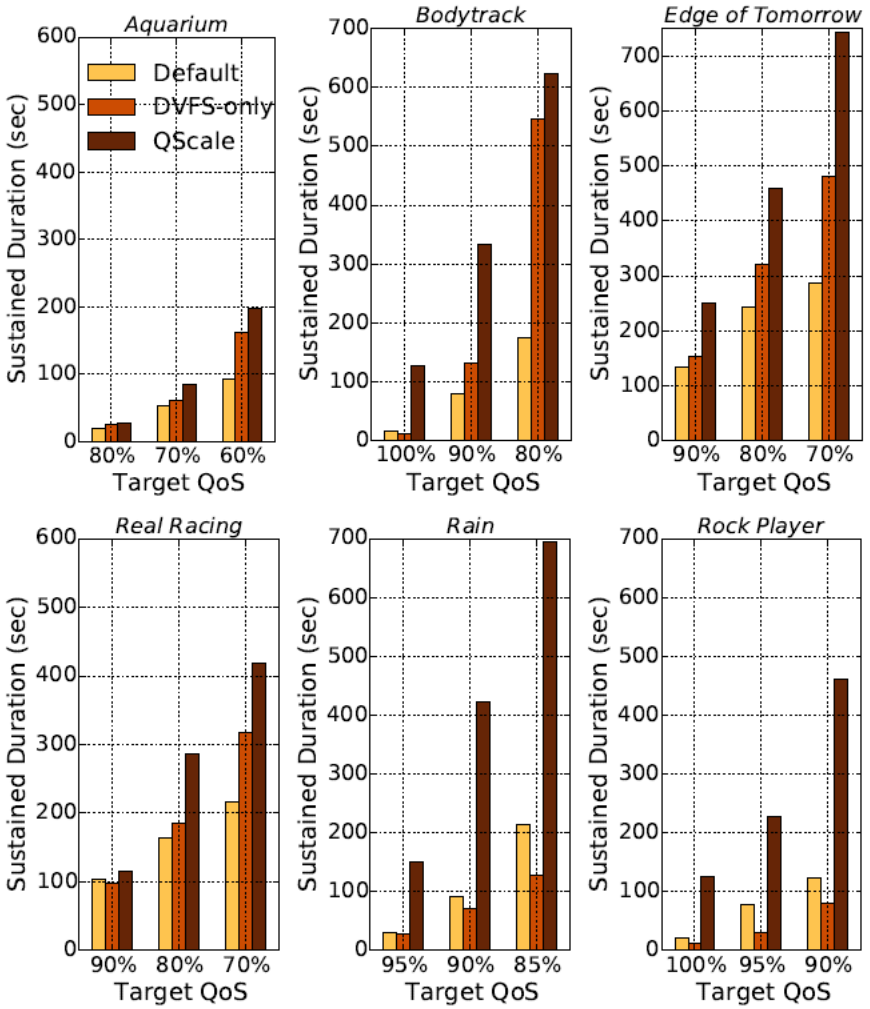
Table 2: Summary of results on Nexus 5.

App.	T_{max} (CPU)	T_{max} (Skin)	T_{max} (Battery)	Time to $T_{CPU,lim}$	Time to $T_{SKIN,lim}$	QoS Loss
FFT	96 °C	48 °C	40.2 °C	5.9 sec	29.2 sec	48.1%
SOR	98 °C	49 °C	40.8 °C	4.5 sec	29.4 sec	49.0%
H264	88 °C	44 °C	36.1 °C	-	37.3 sec	48.1%
Bodytrack	85 °C	44 °C	37.7 °C	-	53.9 sec	44.9%
Aquarium	67 °C	44 °C	37.6 °C	-	138.6 sec	44.4%
Edge of T.	67 °C	44 °C	37.7 °C	-	122.1 sec	44.1%
Real Racing	65 °C	44 °C	37.4 °C	-	160.1 sec	21.6%
MX Player	71 °C	42 °C	36.5 °C	-	184.1 sec	38.7%

Evaluation of QScale

- Policies in comparison:
  - **Default**: Android's Interactive DVFS govenor + HMP scheduler.
  - **DVFS-only:** Closed-loop DVFS controller + HMP scheduler.
  - **QScale**: Closed-loop DVFS controller + criticality-aware thread mapping.

- Figures show the sustained durations for each application under different Qos targets.
- QScale provides the longest durations of sustainable performance.
- Larger improvements (e.g., up to 8x for bodytrack) Rain, Bodytrack, Rock Player due to criticality awareness in QScale.



Qscale Overview

We propose **QScale** for providing thermally-efficient QoS management: [ICCAD'16]

- Runtime monitoring of CPU-GPU thermal coupling for core allocation.
- Thread-criticality driven scheduling for big.LITTLE.
- Closed-loop runtime DVFS control to guarantee desired performance.