Just Enough is More:
Achieving Sustainable Performance in Mobile Devices under Thermal Limitations*

Onur Sahin, Paul T. Varghese, Ayse K. Coskun

Department of Electrical and Computer Engineering

BOSTON UNIVERSITY

January 2016

* *Published in International Conference on Computer-Aided Design (ICCAD) 2015*

Mobile Devices: Trends & Thermal Challenges

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps
- Elevated chip/skin temperatures

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps
- Elevated chip/skin temperatures
  - Tightly integrated on/off-chip components

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps
- Elevated chip/skin temperatures
  - Tightly integrated on/off-chip components
  - Efficient active cooling?

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑cores + ↑frequency + performance demanding apps
- Elevated chip/skin temperatures
  - Tightly integrated on/off-chip components
  - Efficient active cooling?
  - Power and size restrictions apply

Mobile Devices: Trends & Thermal Challenges

- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps
- Elevated chip/skin temperatures
  - Tightly integrated on/off-chip components
  - Efficient active cooling?
  - Power and size restrictions apply
- Thermal throttling

Mobile Devices: Trends & Thermal Challenges

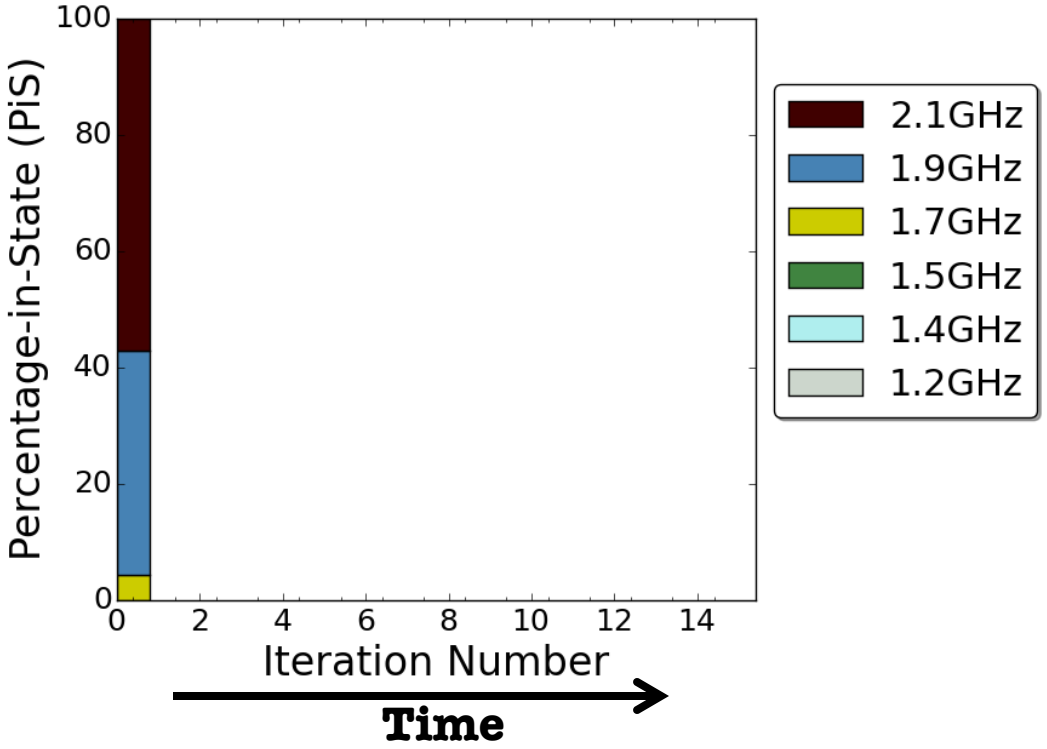
- Growing power density
  - ↑ cores + ↑ frequency + performance demanding apps
- Elevated chip/skin temperatures
  - Tightly integrated on/off-chip components
  - Efficient active cooling?
  - Power and size restrictions apply
- Thermal throttling
  - User experience?

Outline

- **Problem Identification**
- Proposed Solution
- Evaluation Methodology
- Evaluation Results

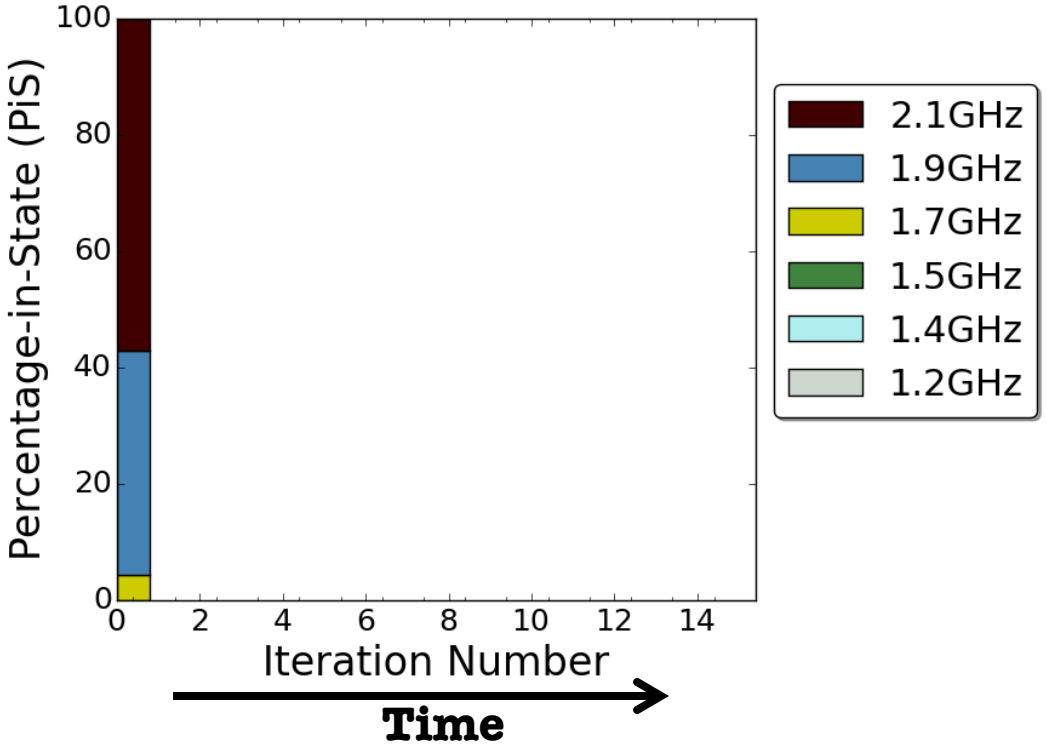
Problem Motivation: Throttling over extended durations

Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

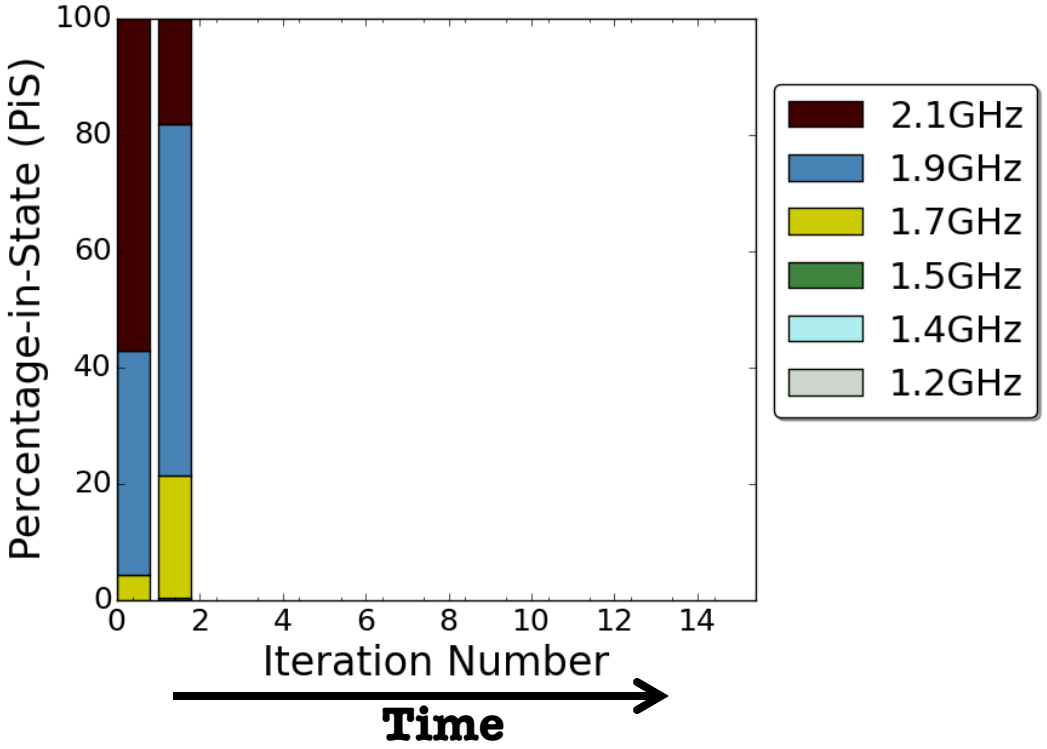
Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

- ✓ Works well for short-term performance

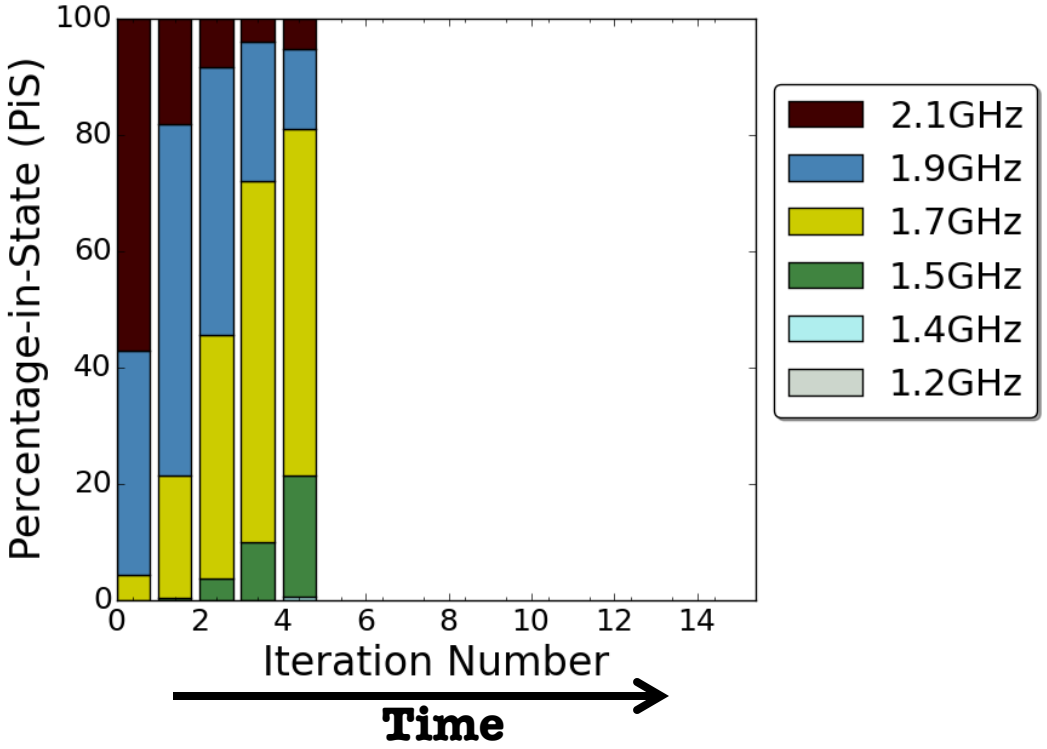
Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

- ✓ Works well for short-term performance
- Longer device use?

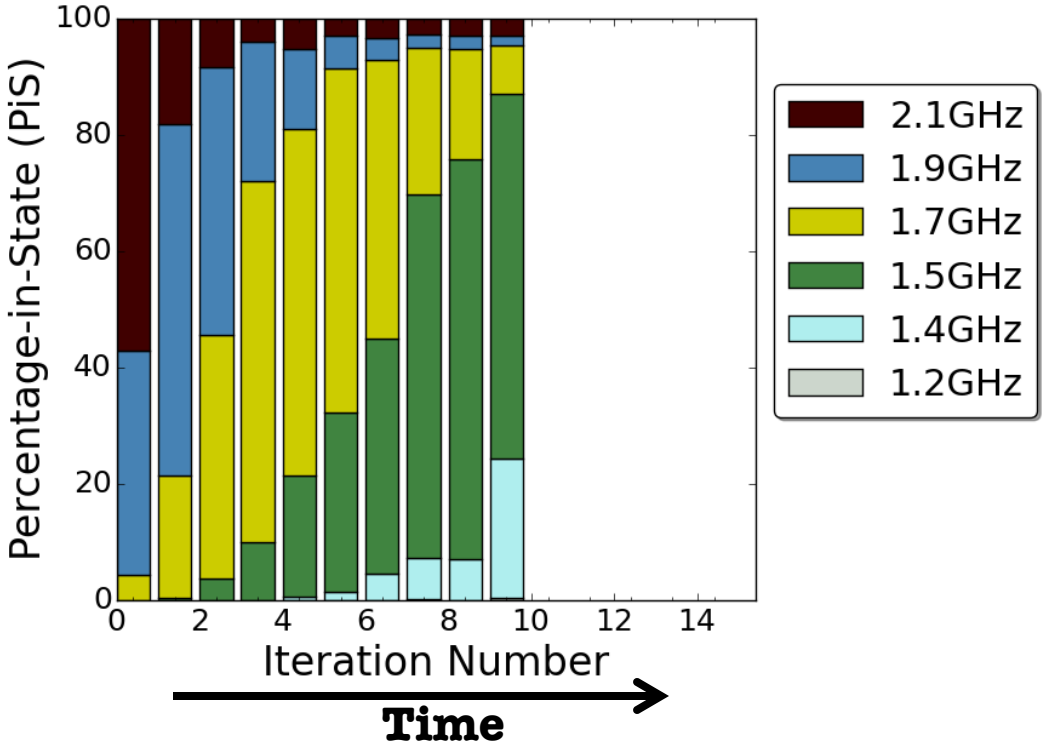
Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

- ✓ Works well for short-term performance
- Longer device use?

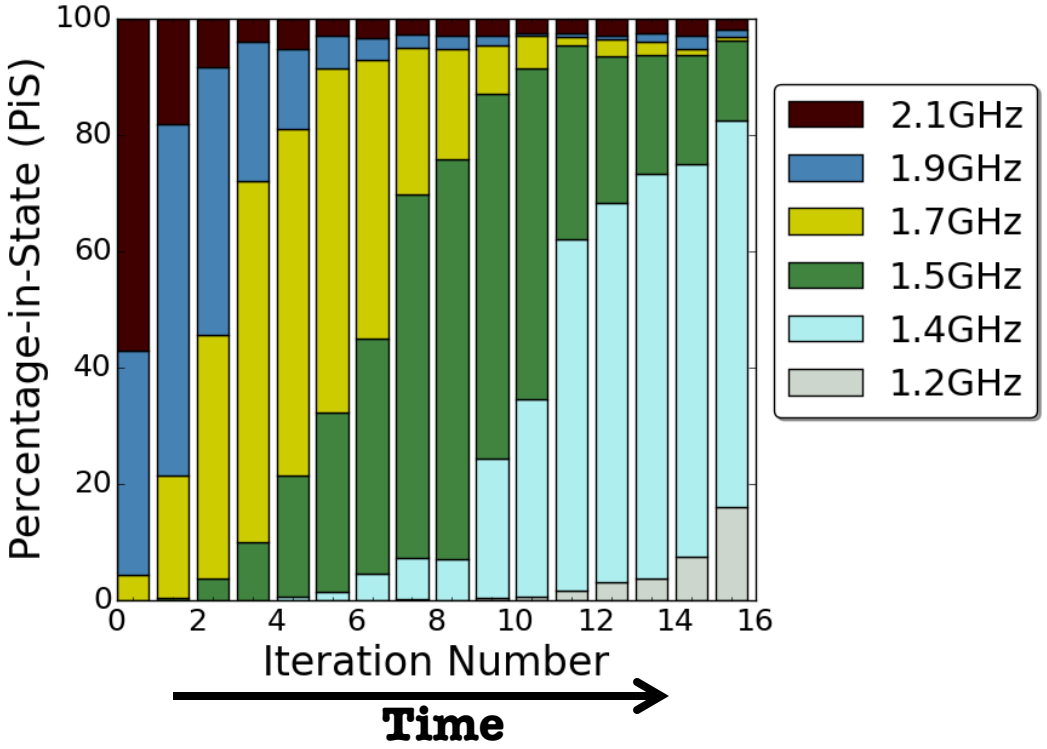
Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

- ✓ Works well for short-term performance
- Longer device use?

Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

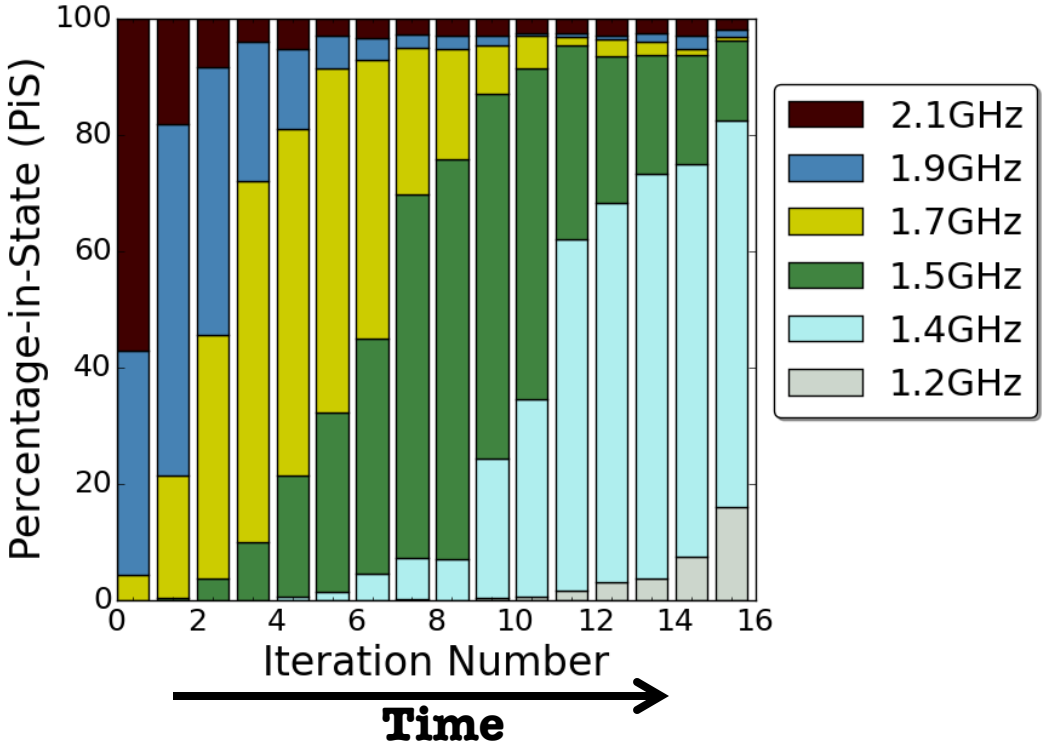
✓ Works well for short-term performance

- Longer device use?

✗ Performance becomes unsustainable

*http://arstechnica.com/gadgets/2013/11/when-benchmarks-arent-enough-cpu-performance-in-the-nexus-5.

Problem Motivation: Throttling over extended durations



Maximizing performance under thermal constraints

✓ Works well for short-term performance

• Longer device use?

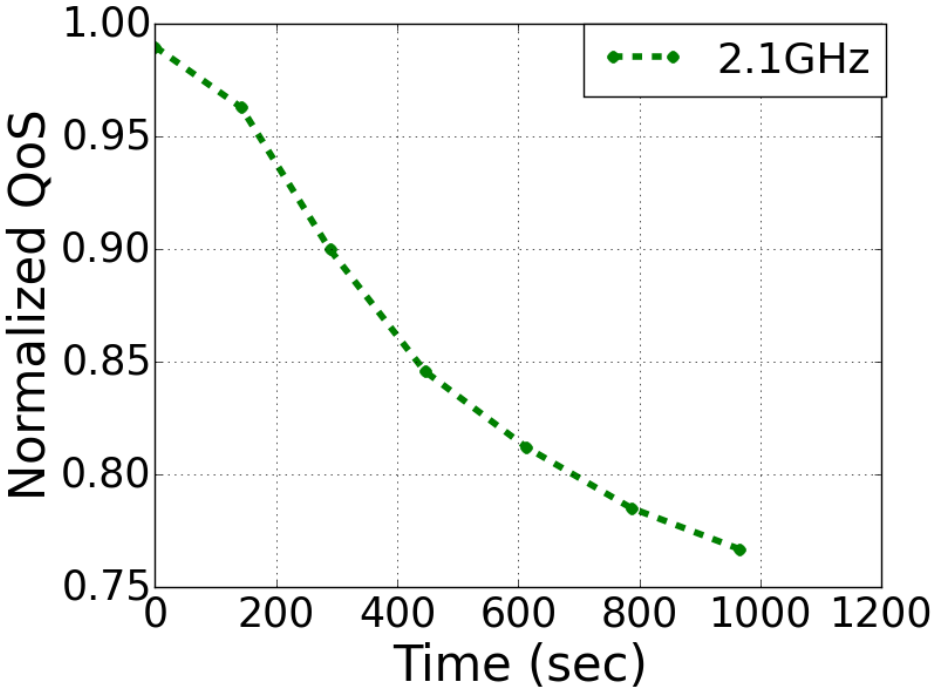
✗ Performance becomes unsustainable

Greedily improving immediate performance hurts sustained performance

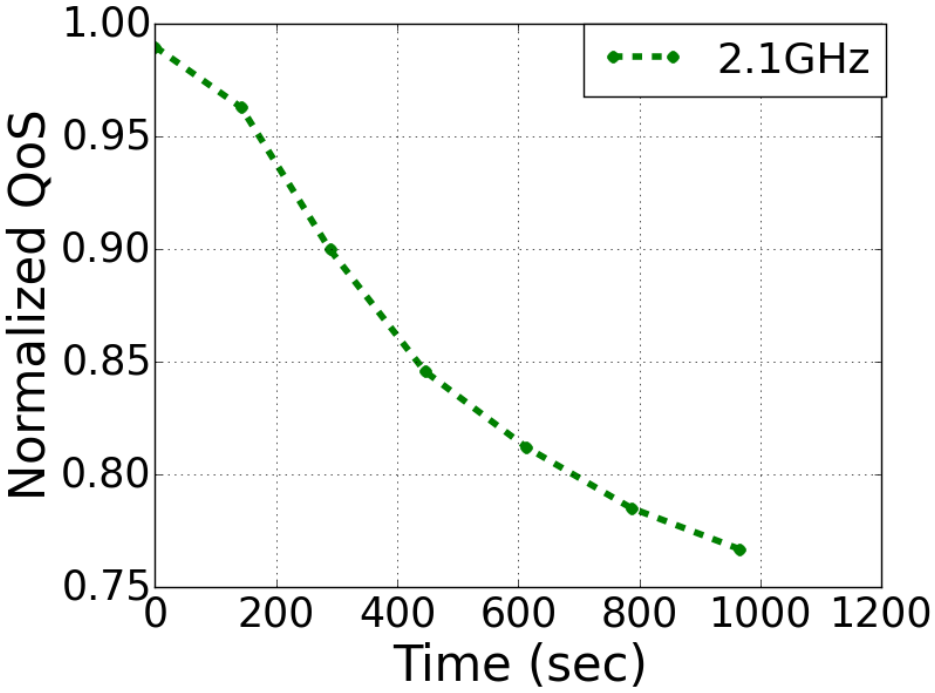
*http://arstechnica.com/gadgets/2013/11/when-benchmarks-arent-enough-cpu-performance-in-the-nexus-5.

QoS-Temperature Tradeoff for Sustainability

QoS-Temperature Tradeoff for Sustainability

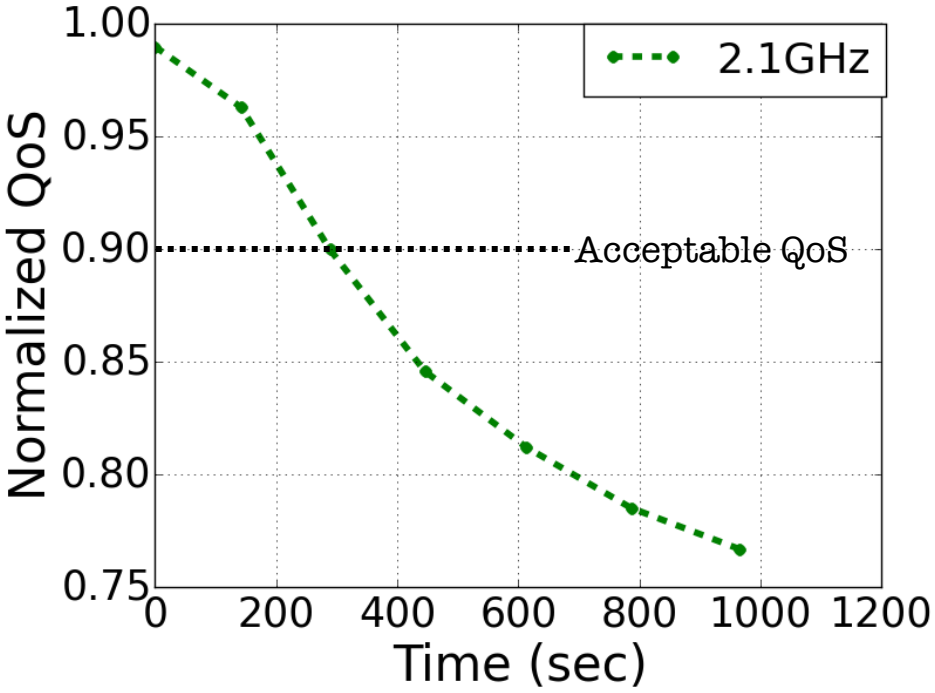


QoS-Temperature Tradeoff for Sustainability



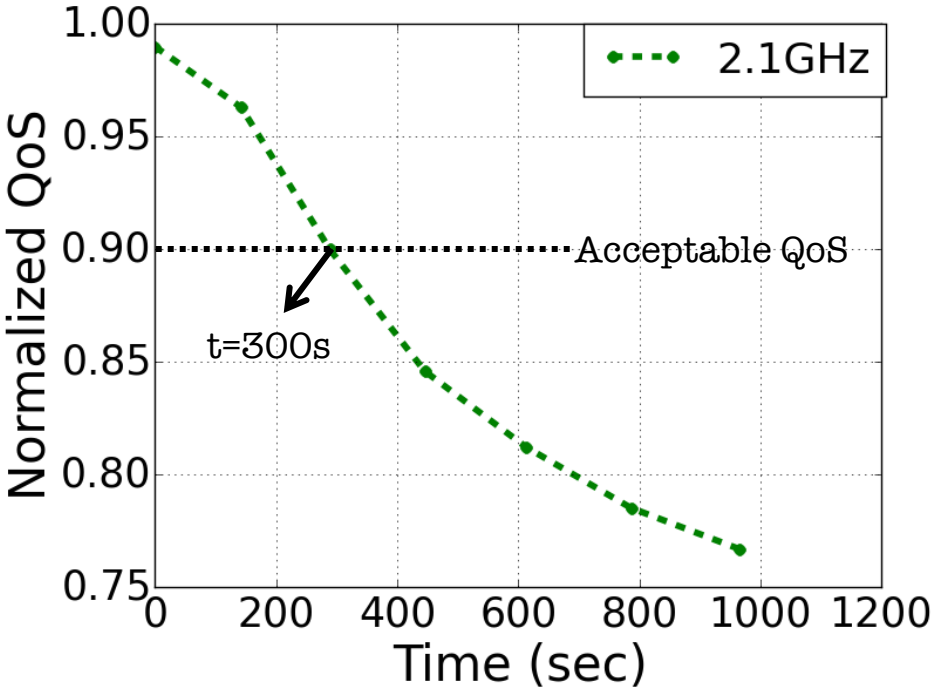
- QoS degrades over time
- QoS metric is app-specific

QoS-Temperature Tradeoff for Sustainability



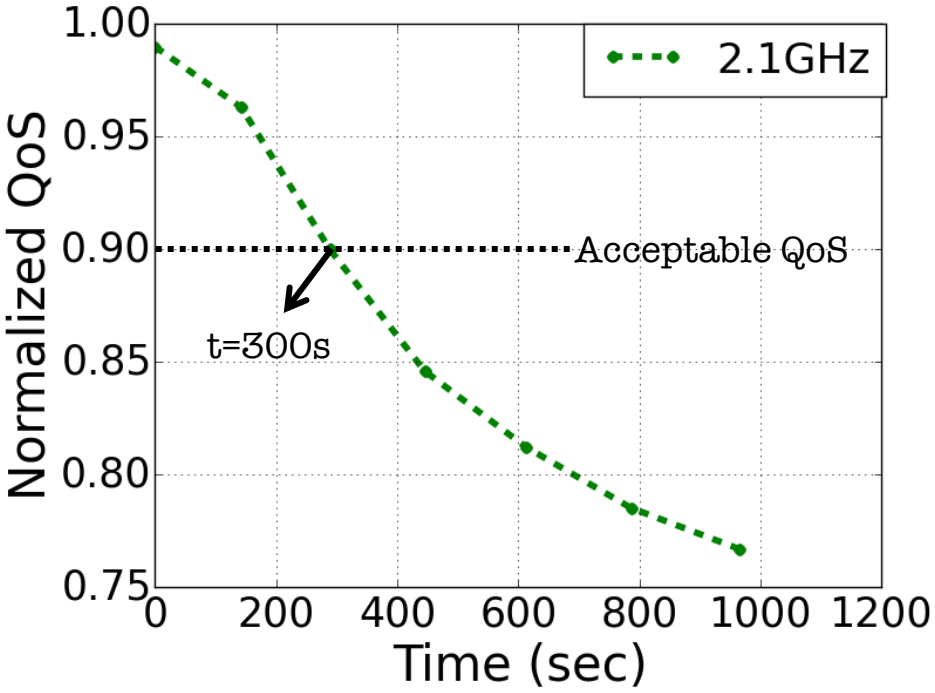
- QoS degrades over time
- QoS metric is app-specific

QoS-Temperature Tradeoff for Sustainability



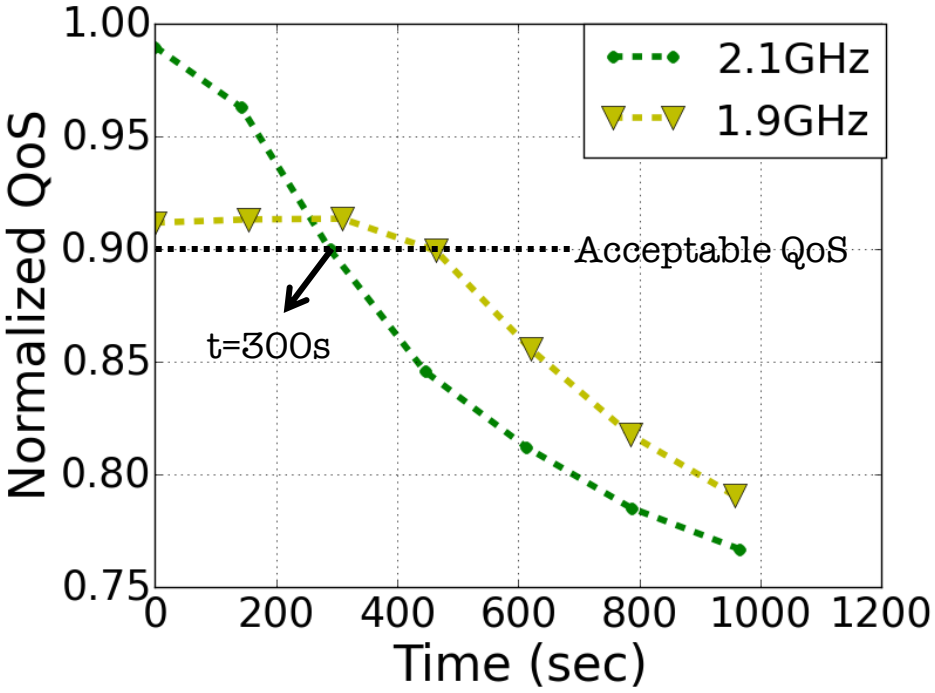
- QoS degrades over time
- QoS metric is app-specific

QoS-Temperature Tradeoff for Sustainability



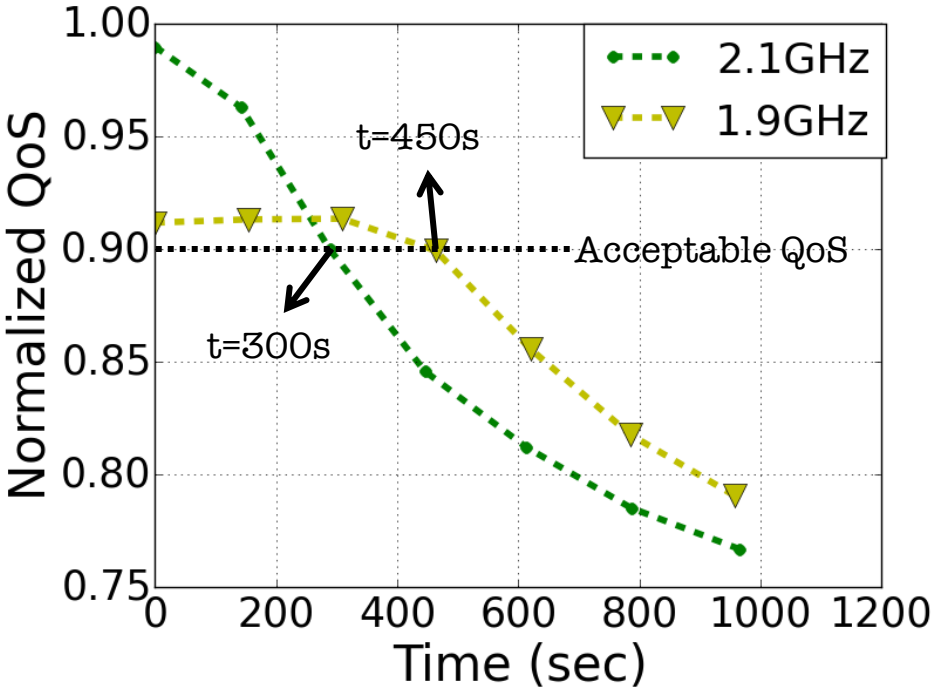
- QoS degrades over time
- QoS metric is app-specific
- Limit short-term performance to **"just enough"** level

QoS-Temperature Tradeoff for Sustainability



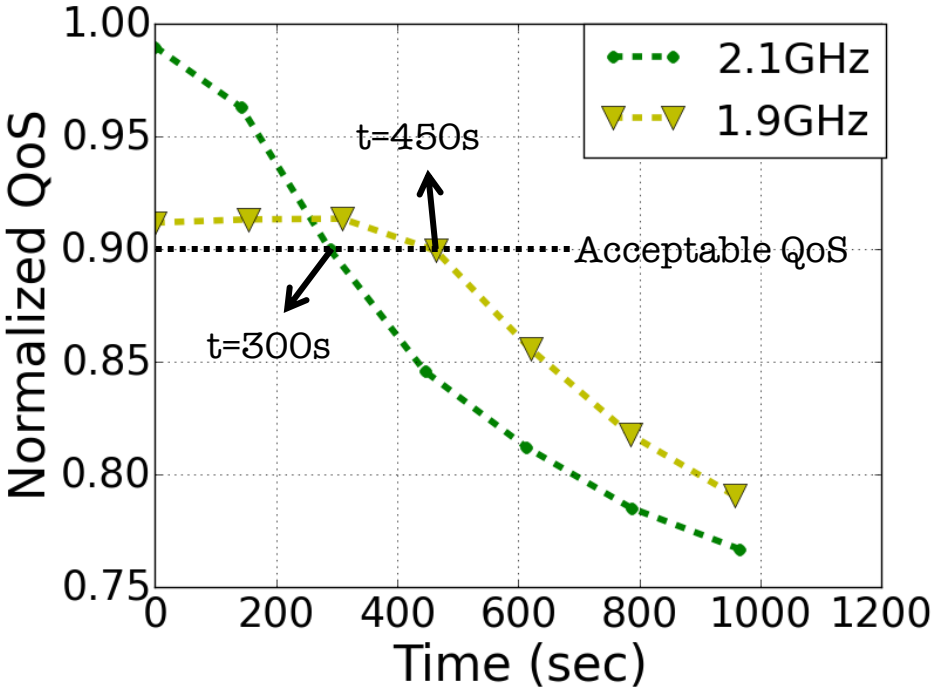
- QoS degrades over time
- QoS metric is app-specific
- Limit short-term performance to **"just enough"** level

QoS-Temperature Tradeoff for Sustainability



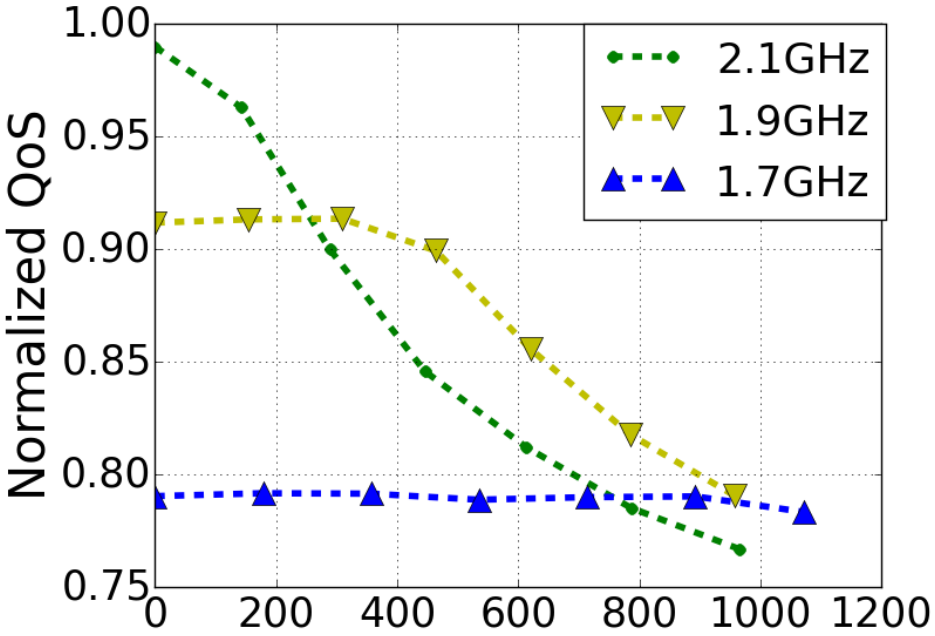
- QoS degrades over time
- QoS metric is app-specific
- Limit short-term performance to **"just enough"** level

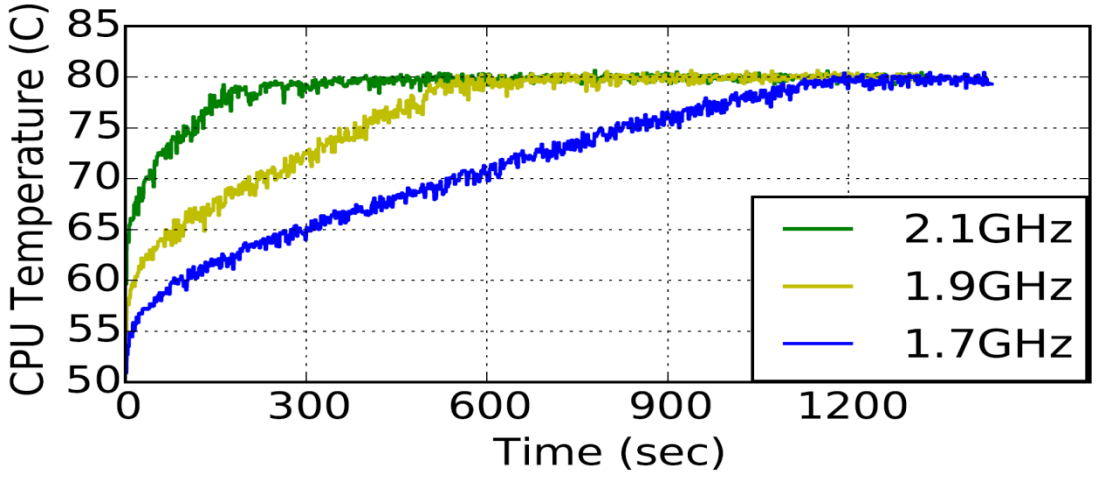
QoS-Temperature Tradeoff for Sustainability



- QoS degrades over time
- QoS metric is app-specific
- Limit short-term performance to **"just enough"** level
- Extends sustainability of acceptable QoS levels

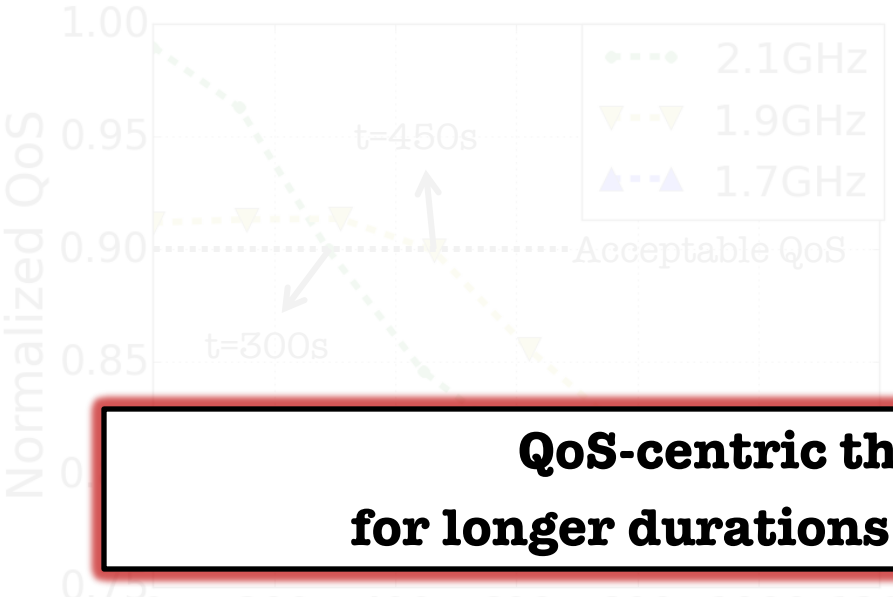
QoS-Temperature Tradeoff for Sustainability





- QoS degrades over time
- QoS metric is app-specific
- Limit short-term performance to **"just enough"** level
- Extends sustainability of acceptable QoS levels

QoS-Temperature Tradeoff for Sustainability



- QoS degrades over time
- Limit short-term performance to **"just enough"** level
- Extends sustainability of acceptable QoS levels

QoS-centric thermal management
for longer durations of sustained performance

Outline

- Background and Problem Identification
- **Proposed Solution**
- Evaluation Methodology
- Evaluation Results

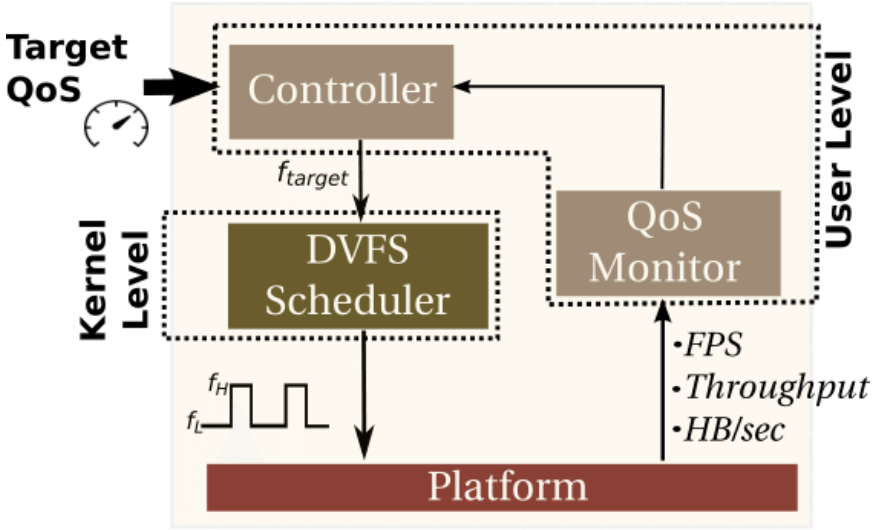
QoS Tuning for Sustained Performance

QoS Tuning for Sustained Performance

- Efficient thermal management while keeping QoS at “just enough”

QoS Tuning for Sustained Performance

- Efficient thermal management while keeping QoS at "just enough"



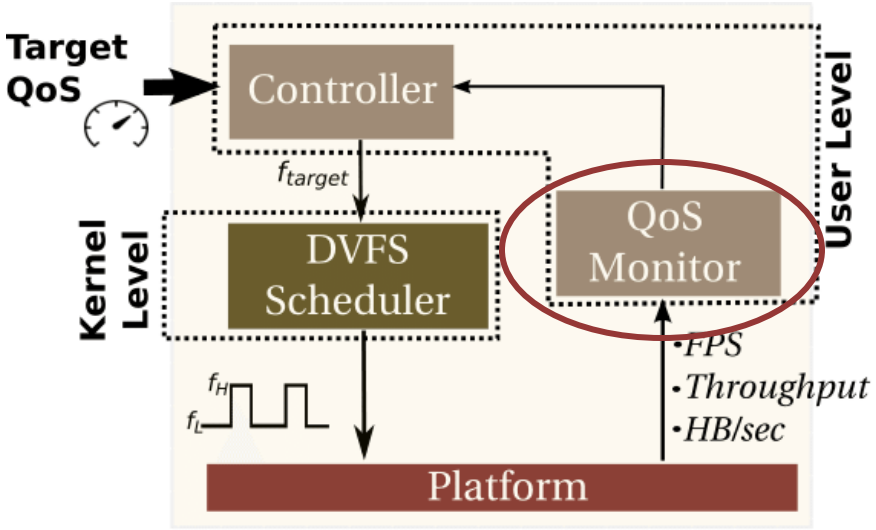
Monitoring application specific QoS information

Closed-loop control policy for adjusting $\mathbf{f_{target}}$

Fine-grained thermally-aware DVFS state scheduling

QoS Tuning for Sustained Performance

- Efficient thermal management while keeping QoS at "just enough"



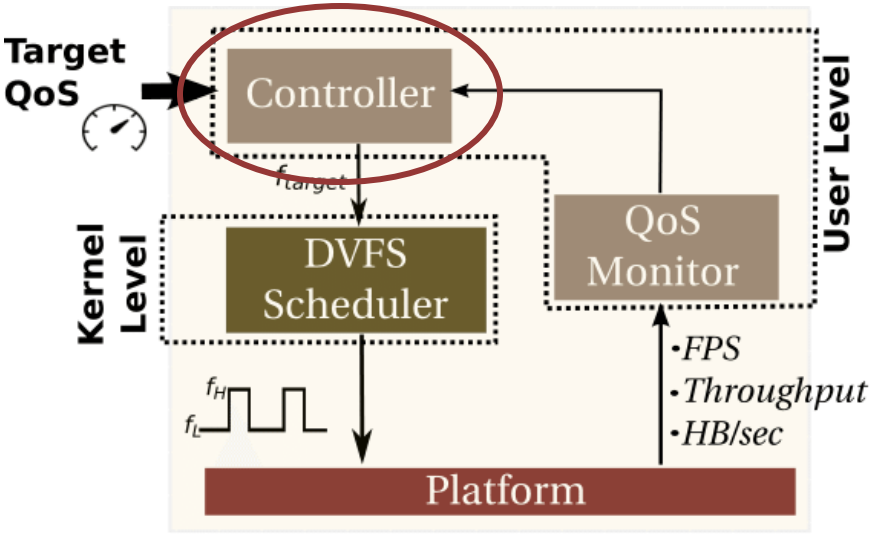
➡ Monitoring application specific QoS information

Closed-loop control policy for adjusting $\mathbf{f_{target}}$

Fine-grained thermally-aware DVFS state scheduling

QoS Tuning for Sustained Performance

- Efficient thermal management while keeping QoS at "just enough"



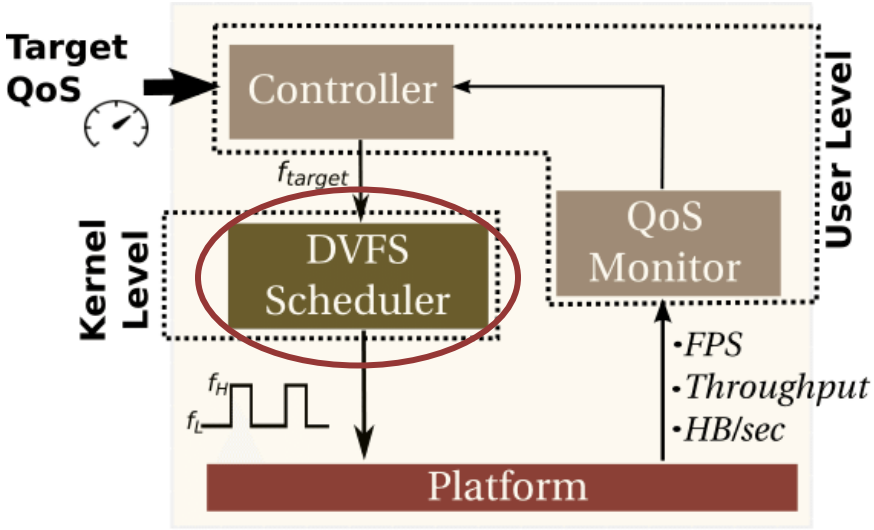
Monitoring application specific QoS information

→ Closed-loop control policy for adjusting $\mathbf{f_{target}}$

Fine-grained thermally-aware DVFS state scheduling

QoS Tuning for Sustained Performance

- Efficient thermal management while keeping QoS at "just enough"



Monitoring application specific QoS information

Closed-loop control policy for adjusting $\mathbf{f_{target}}$

Fine-grained thermally-aware DVFS state scheduling

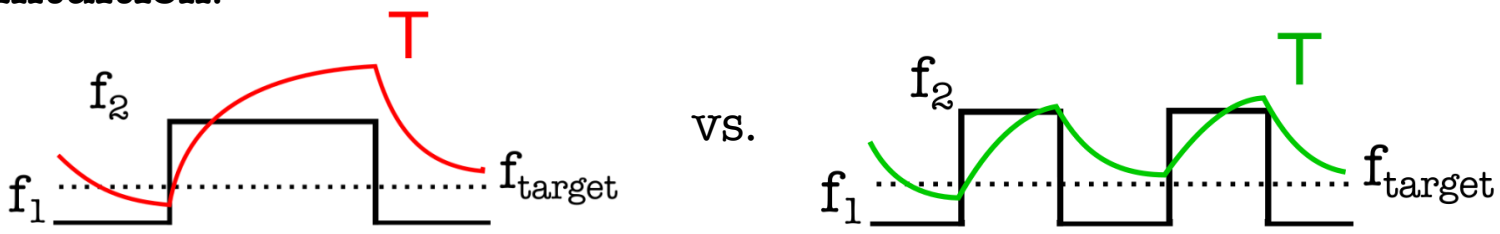
Thermally-Efficient DVFS State Scheduling

- Thermally-efficient time scheduling of discrete DVFS states

Thermally-Efficient DVFS State Scheduling

- Thermally-efficient time scheduling of discrete DVFS states

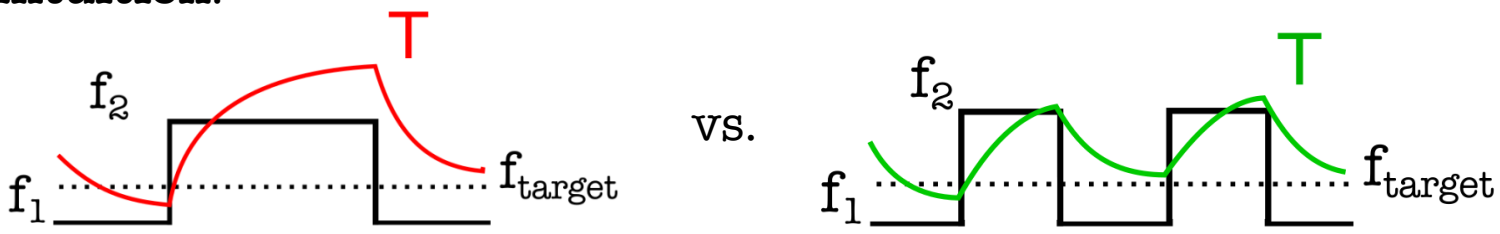
Intuition:



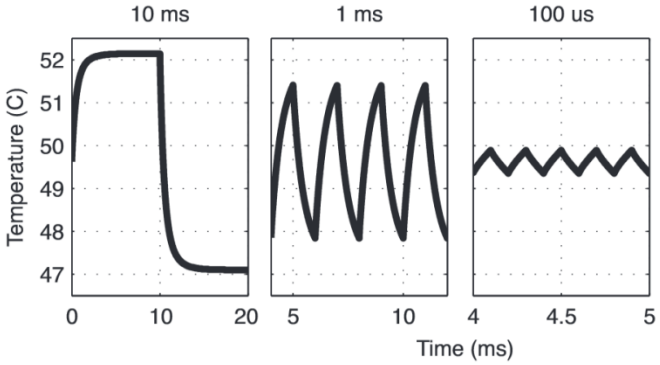
Thermally-Efficient DVFS State Scheduling

- Thermally-efficient time scheduling of discrete DVFS states

Intuition:



Fast DVFS:

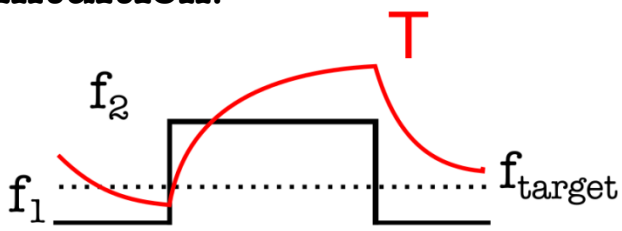


DVFS Granularity
Impact on temperature
[Khan, IGCC'11]

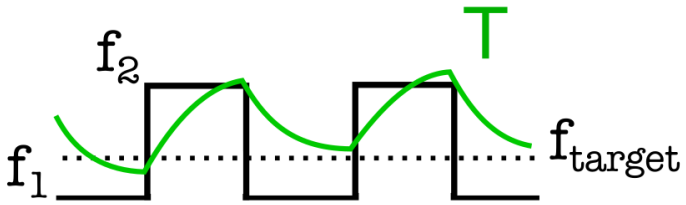
Thermally-Efficient DVFS State Scheduling

- Thermally-efficient time scheduling of discrete DVFS states

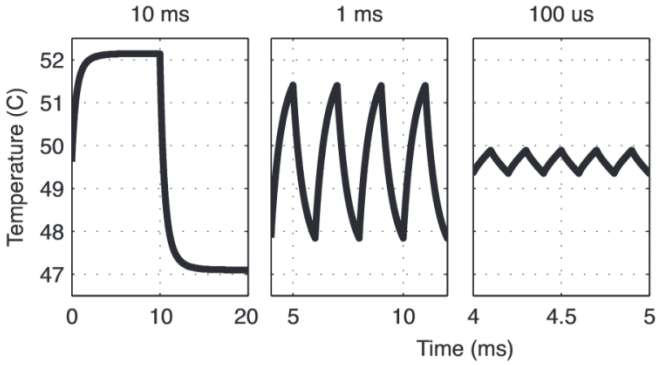
Intuition:



vs.

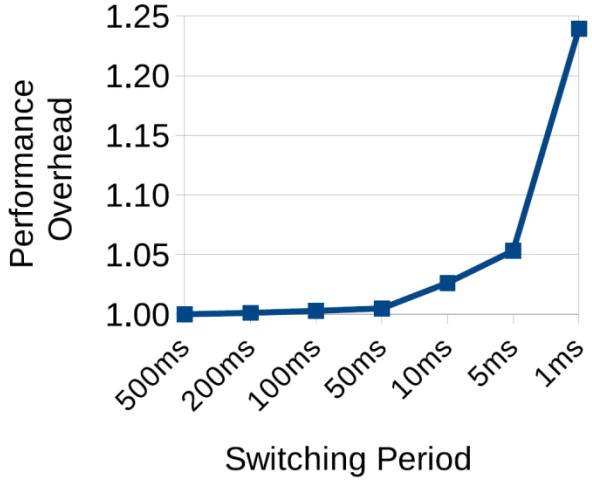


Fast DVFS:



DVFS Granularity
Impact on temperature
[Khan, IGCC'11]

DVFS overhead matters too!



Outline

- Background and Problem Identification
- Proposed Solution
- **Experimental Methodology**
- Evaluation Results

Experimental Methodology

Hardware Platform:

- Qualcomm Snapdragon 800 MDP
  - Quad-core Krait 400 CPU: 300MHz to 2.1GHz

Experimental Methodology

Hardware Platform:

- Qualcomm Snapdragon 800 MDP
  - Quad-core Krait 400 CPU: 300MHz to 2.1GHz

Application Set:

Application	Category	QoS Metric
Sjeng	Artificial Intelligence	Throughput
H.264	Media Processing	Throughput
LU	Math	Throughput
Pearl Boy	Graphics/WebGL	Frames per second
Aquarium	Graphics/WebGL	Frames per second
Bodytrack	Computer Vision	Heartbeats/sec

Experimental Methodology

Hardware Platform:

- Qualcomm Snapdragon 800 MDP
  - Quad-core Krait 400 CPU: 300MHz to 2.1GHz

Application Set:

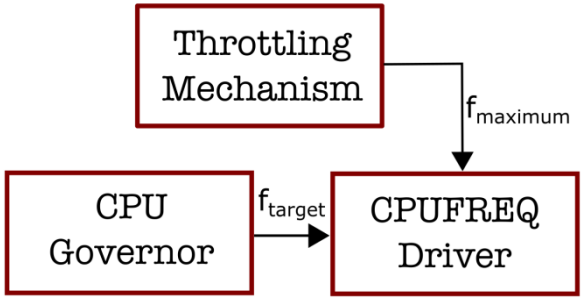
Application	Category	QoS Metric
Sjeng	Artificial Intelligence	Throughput
H.264	Media Processing	Throughput
LU	Math	Throughput
Pearl Boy	Graphics/WebGL	Frames per second
Aquarium	Graphics/WebGL	Frames per second
Bodytrack	Computer Vision	Heartbeats/sec

Power management:

- Linux Ondemand governor

Thermal Management:

- T_{CPU} : DVFS using a PID Controller
- T_{SKIN}: Threshold based reactive DVFS

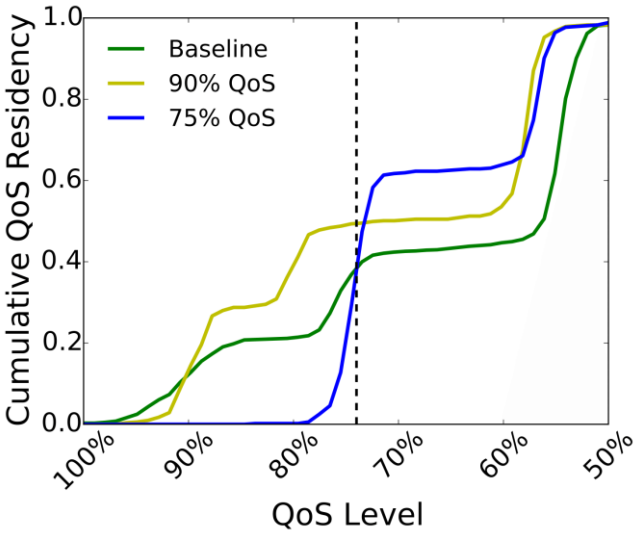


Outline

- Background and Problem Identification
- Proposed Solution
- Evaluation Methodology
- **Evaluation Results**

Evaluation: Extending the Sustained QoS Durations

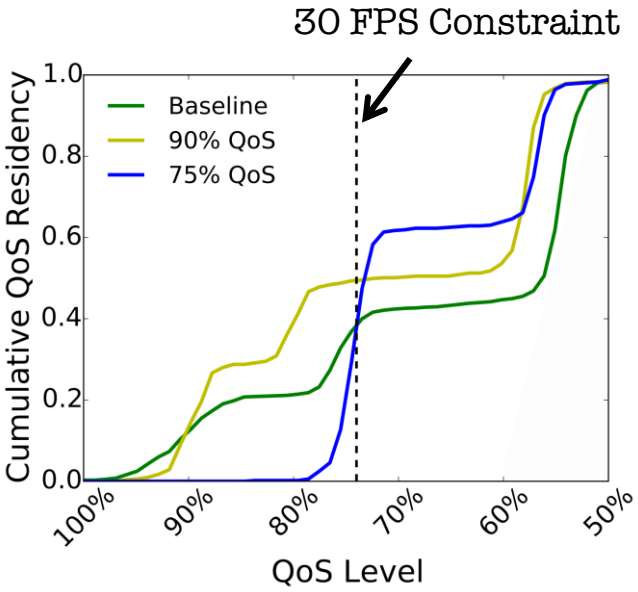
Time spent above a QoS level:



Aquarium

Evaluation: Extending the Sustained QoS Durations

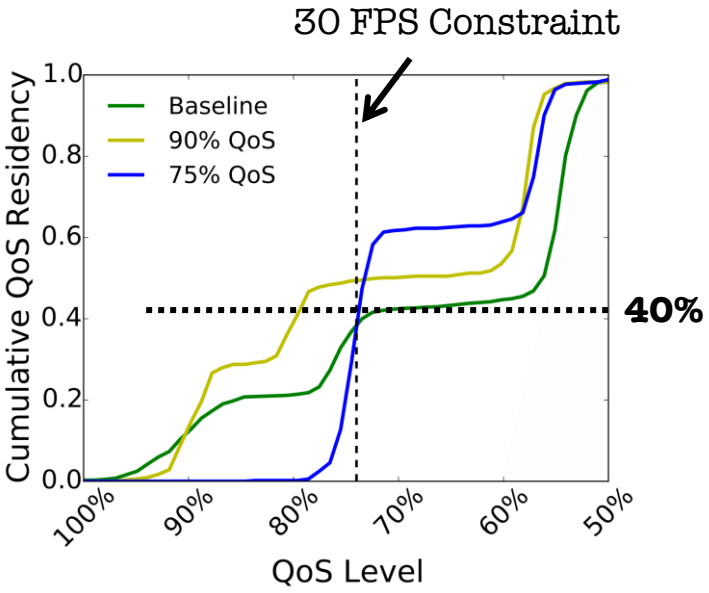
Time spent above a QoS level:



Aquarium

Evaluation: Extending the Sustained QoS Durations

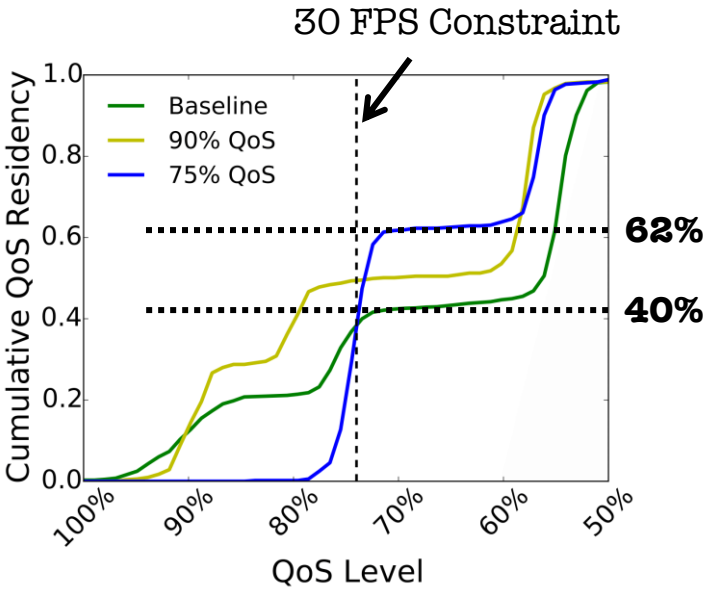
Time spent above a QoS level:



Aquarium

Evaluation: Extending the Sustained QoS Durations

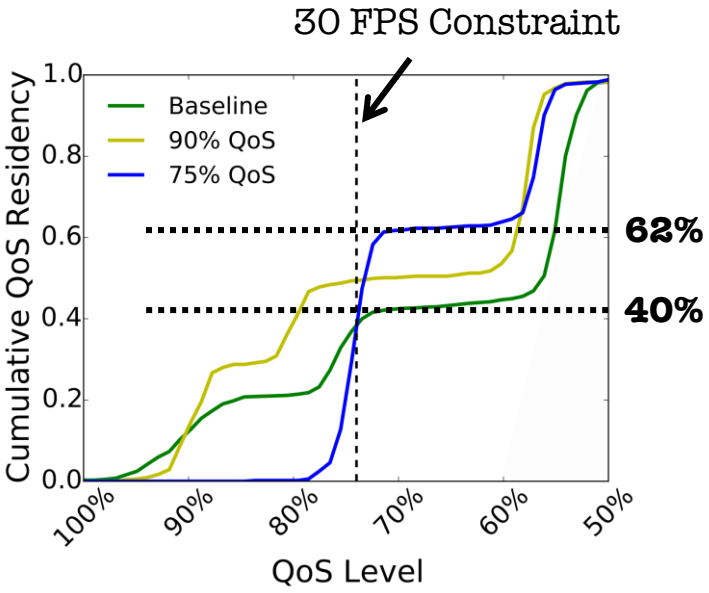
Time spent above a QoS level:



Aquarium

Evaluation: Extending the Sustained QoS Durations

Time spent above a QoS level:

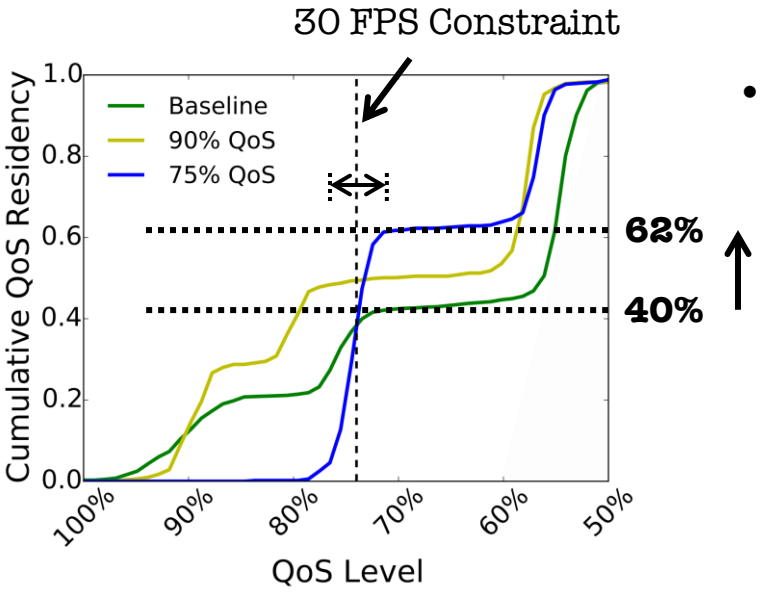


Aquarium

- Sustained duration improves from 40% to 62% of the execution time

Evaluation: Extending the Sustained QoS Durations

Time spent above a QoS level:

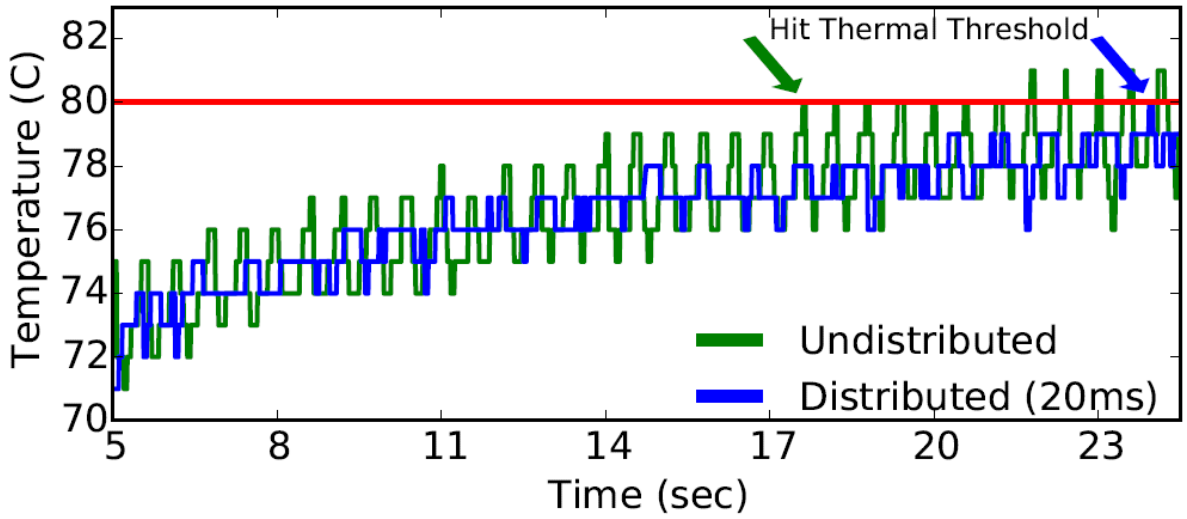


Aquarium

- Sustained duration improves from 40% to 62% of the execution time

55% longer time with acceptable FPS

Evaluation: DVFS State Scheduler in Action

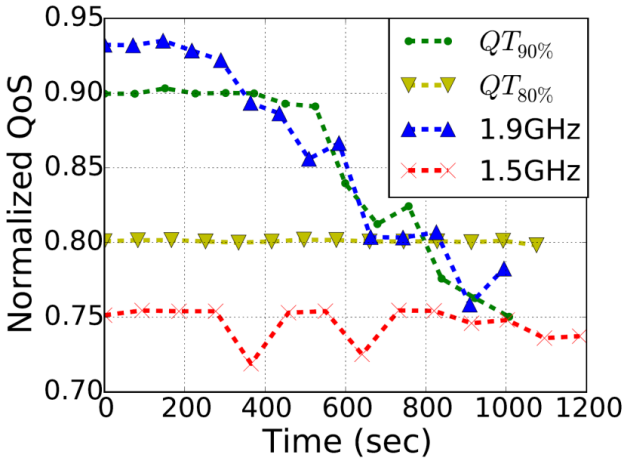


Fine-grained distributed vs. Coarse-grained undistributed DVFS

Evaluation: Precise QoS Control

Average QoS per iteration (normalized to QoS_{MAX}):

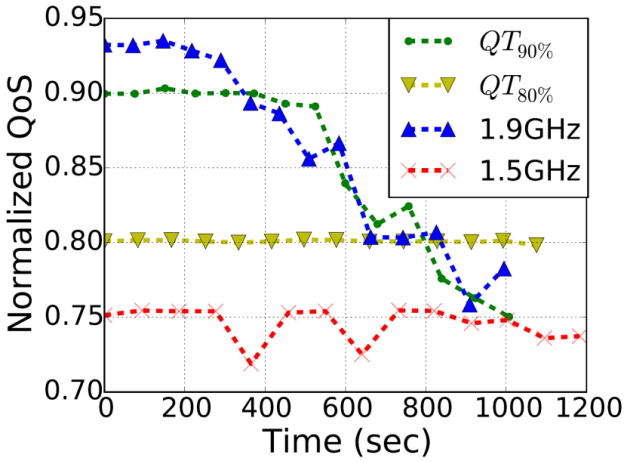
Comparison to performance-aware static frequency selection policy



Evaluation: Precise QoS Control

Average QoS per iteration (normalized to QoS_{MAX}):

Comparison to performance-aware static frequency selection policy

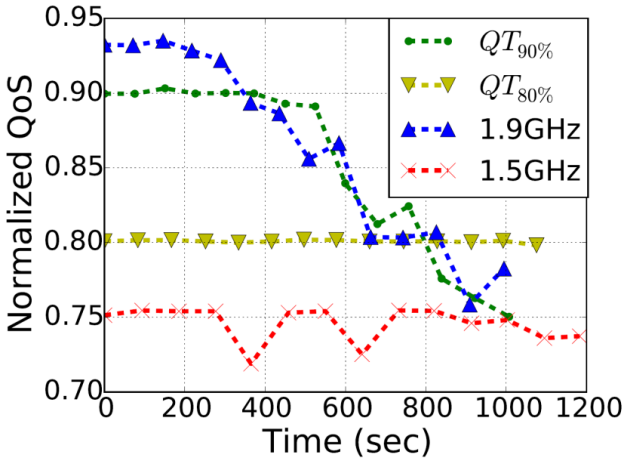


- Static frequencies may result in over/under-provisioning

Evaluation: Precise QoS Control

Average QoS per iteration (normalized to QoS_{MAX}):

Comparison to performance-aware static frequency selection policy

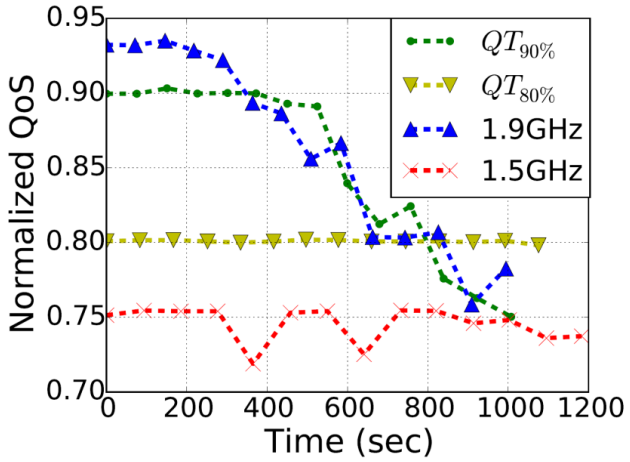


- Static frequencies may result in over/under-provisioning
- Over-provisioning can lead to earlier throttling

Evaluation: Precise QoS Control

Average QoS per iteration (normalized to QoS_{MAX}):

Comparison to performance-aware static frequency selection policy

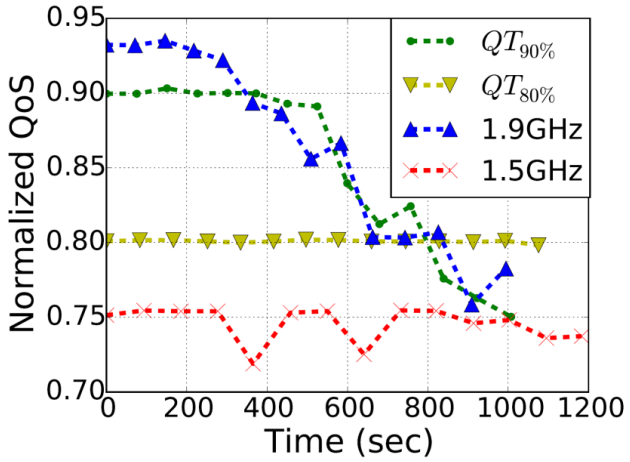


- Static frequencies may result in over/under-provisioning
- Over-provisioning can lead to earlier throttling
- Prosed dynamic controller precisely meets target QoS levels

Evaluation: Precise QoS Control

Average QoS per iteration (normalized to QoS_{MAX}):

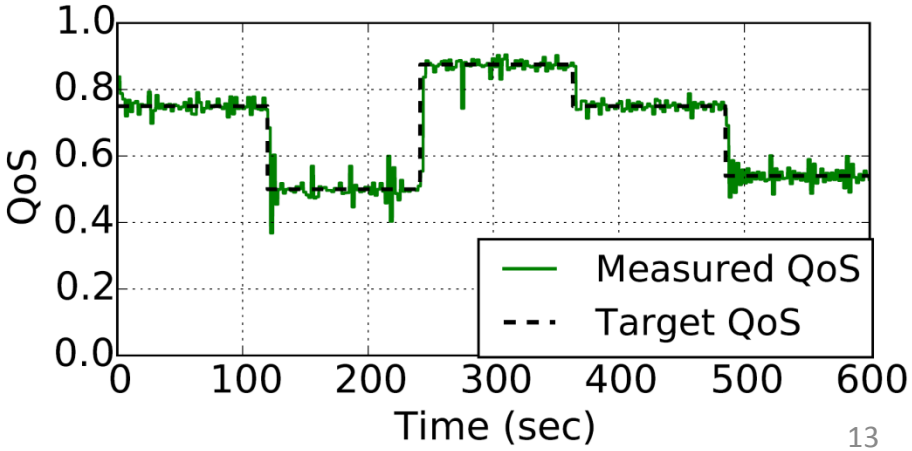
Comparison to performance-aware static frequency selection policy



- Static frequencies may result in over/under-provisioning
- Over-provisioning can lead to earlier throttling
- Prosed dynamic controller precisely meets target QoS levels

Adapting to Dynamic QoS Requirements:

- QoS requirements may change dynamically:
  - Upon user demand, battery level etc.



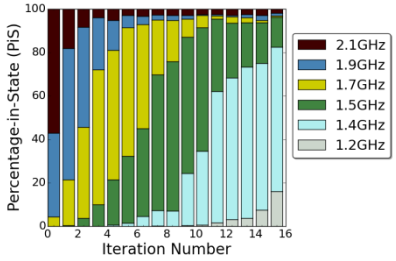
Summary & Takeaways

Summary & Takeaways

- Mobile users require consistent performance

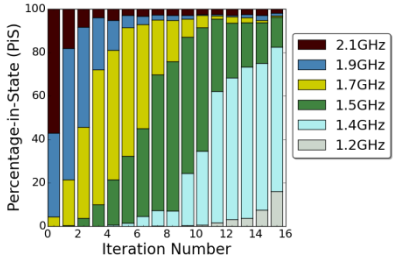
Summary & Takeaways

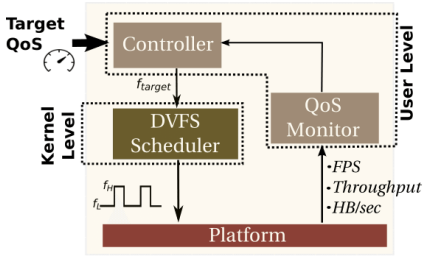
- Mobile users require consistent performance
- Greedy vs. QoS-aware thermal management
- Short-term vs. sustained performance



Summary & Takeaways

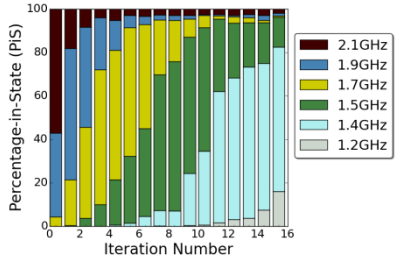
- Mobile users require consistent performance
- Greedy vs. QoS-aware thermal management
- Short-term vs. sustained performance
- QoS-aware thermal management for sustainable performance

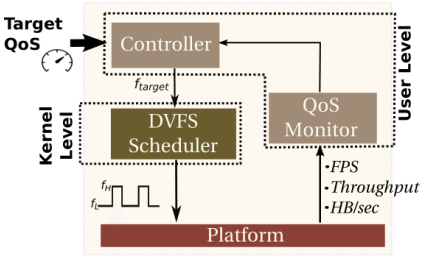




Summary & Takeaways

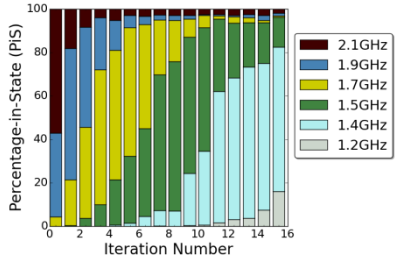
- Mobile users require consistent performance
- Greedy vs. QoS-aware thermal management
- Short-term vs. sustained performance
- QoS-aware thermal management for sustainable performance
- Longer durations of acceptable QoS *(up to 74%)*

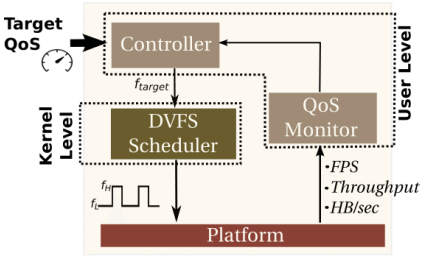




Summary & Takeaways

- Mobile users require consistent performance
- Greedy vs. QoS-aware thermal management
- Short-term vs. sustained performance
- QoS-aware thermal management for sustainable performance
- Longer durations of acceptable QoS *(up to 74%)*





Just Enough is More: Achieving Sustainable Performance in Mobile Devices under Thermal Limitations

Onur Sahin, Ayse K. Coskun
{**sahin**, acoskun}**@bu.edu**

BOSTON UNIVERSITY

PEACLAB
Performance and Energy Aware Computing Lab

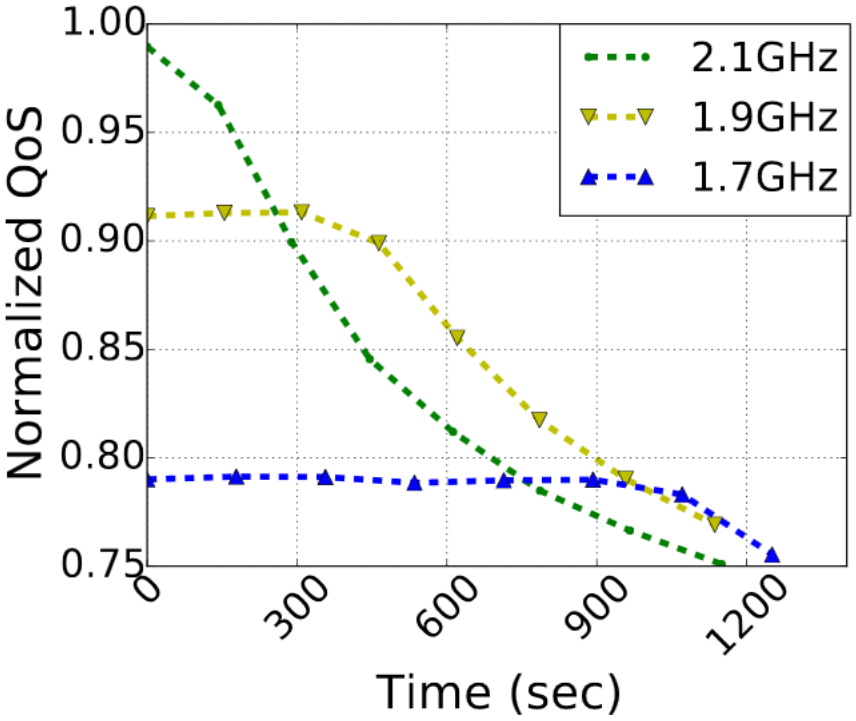
References:

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

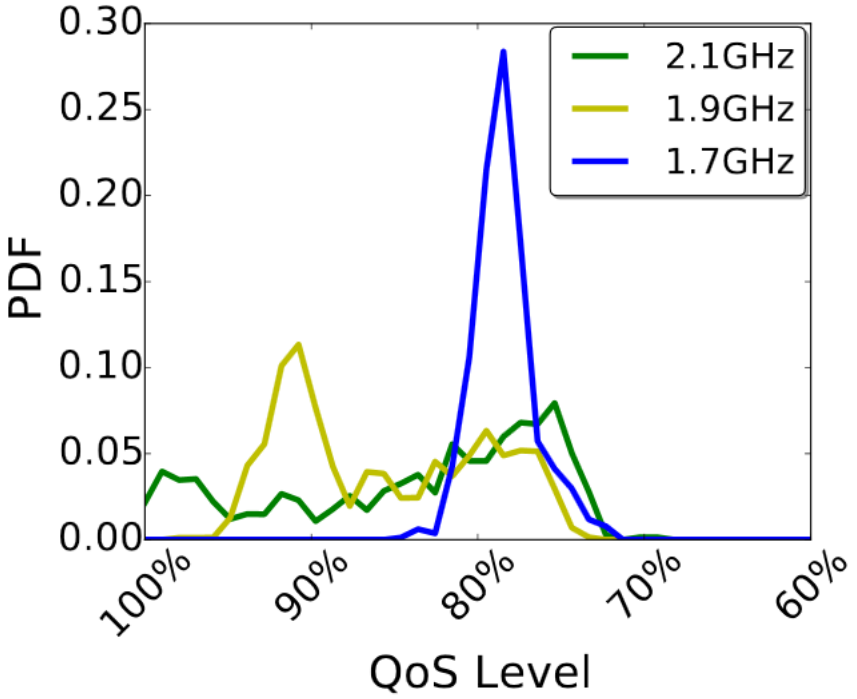
[2] O. Sahin, P.T. Varghese and Ayse K. Coskun. "**Just Enough is More: Achieving Sustainable Performance in Mobile Devices under Thermal Limitations**.", *In International Conference on Computer-Aided Design (ICCAD)*, 2015.

Backup Slides

QoS Distribution for Different DVFS States

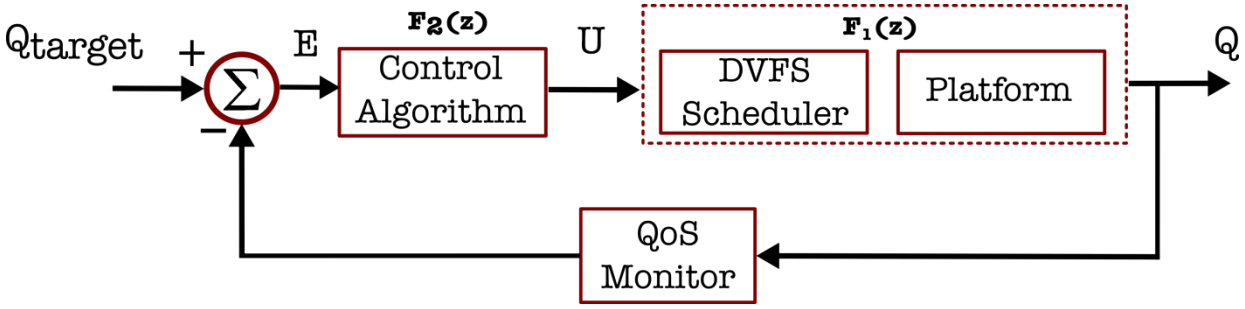


(a) Average QoS over Time

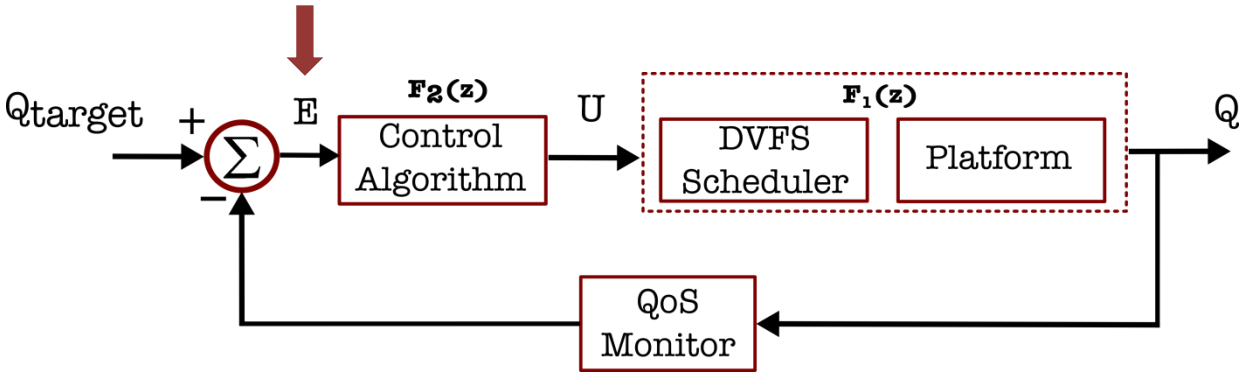


(b) QoS Distribution

Closed-loop QoS Controller Design Details

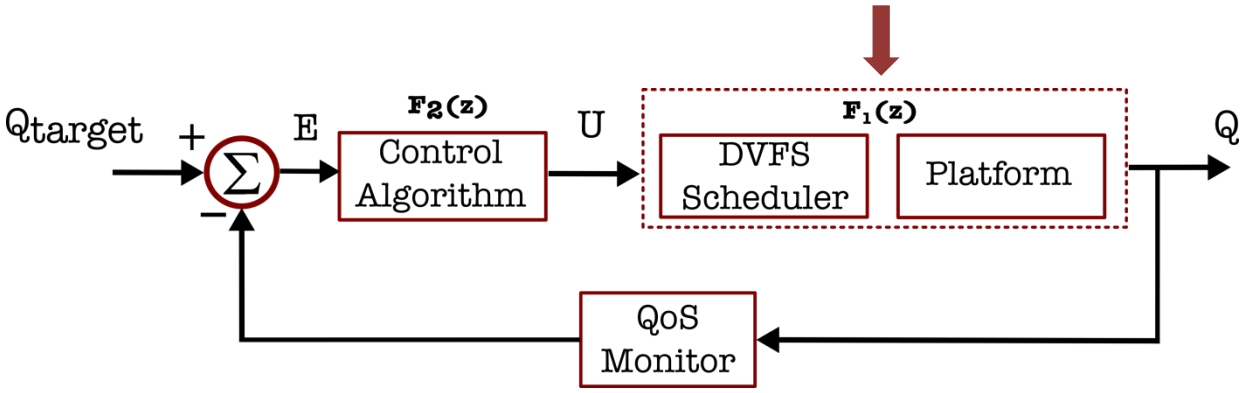


Closed-loop QoS Controller Design Details



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

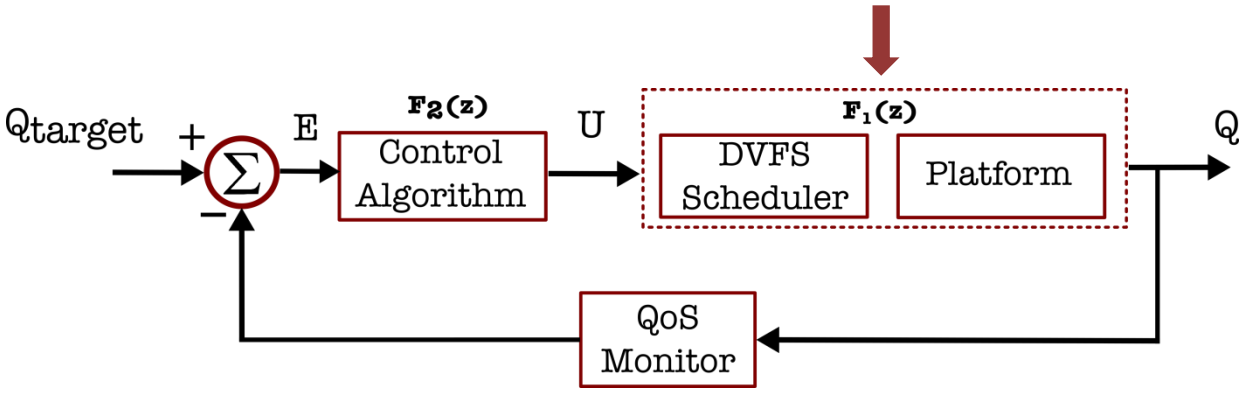
Closed-loop QoS Controller Design Details



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

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

Closed-loop QoS Controller Design Details

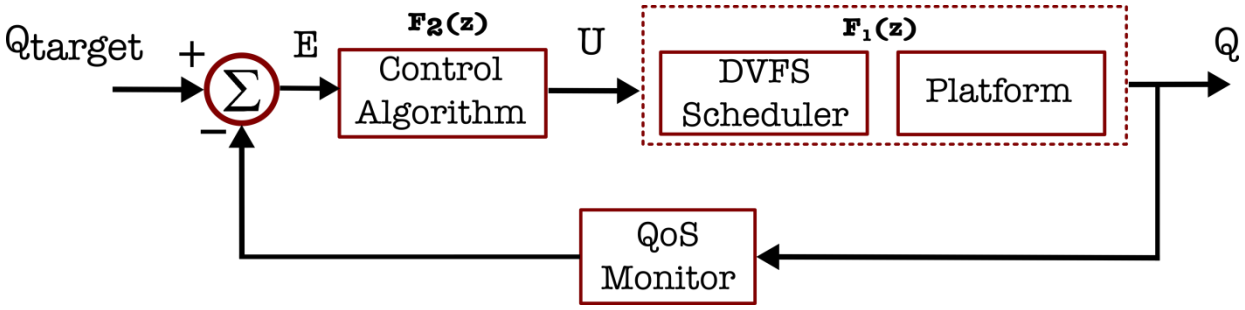


$$e[k] = Q_{target} - Q[k]$$

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

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

Closed-loop QoS Controller Design Details



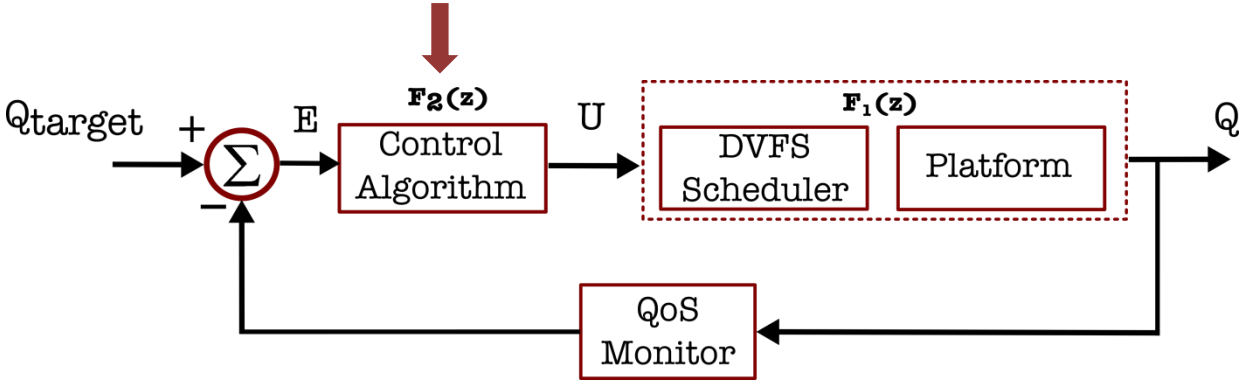
$$e[k] = Q_{target} - Q[k]$$

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

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

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

Closed-loop QoS Controller Design Details



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

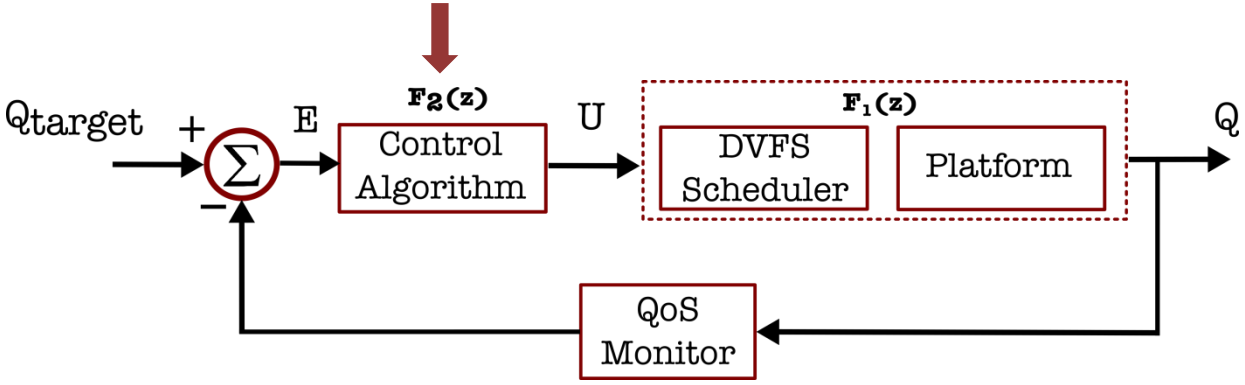
$$Q[k+1] = Q_{max} u[k]$$

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

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

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

Closed-loop QoS Controller Design Details



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

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

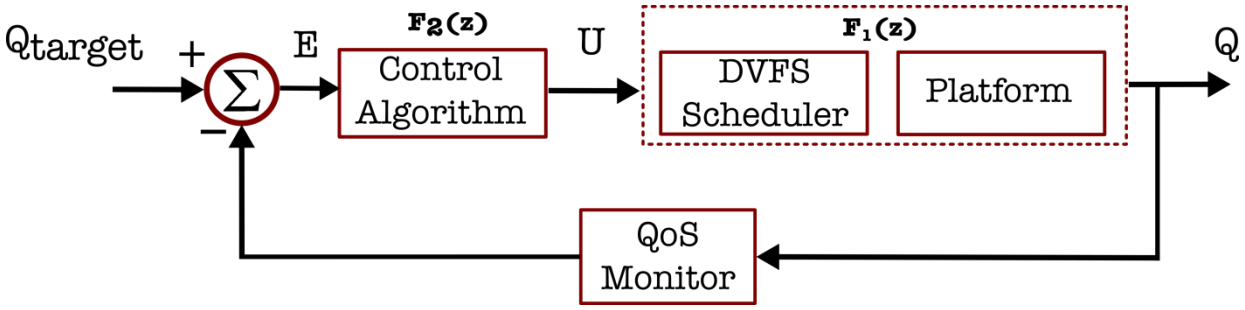
$$F_1(z) = \frac{Q(z)}{U(z)} = \frac{Q_{max}}{z}$$

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

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

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

Closed-loop QoS Controller Design Details



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

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

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

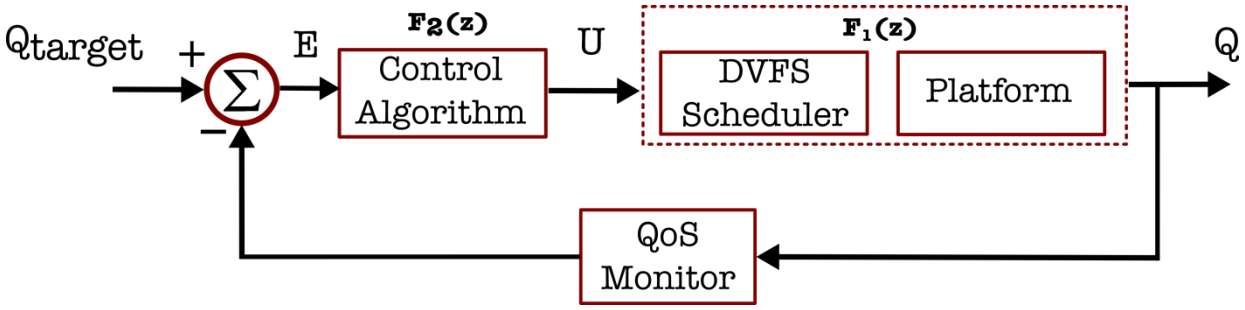
$$G(z) = \frac{F_1(z)F_2(z)}{1 + F_1(z)F_2(z)}$$

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

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

✓ Ensures stable control around Q_{target}

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}

Power, Throttling Improvement and QoS Precision

		H264			Bodytrack			Sjeng			LU			
		ondemand & DTM	QT90	QT80	ondemand & DTM	QT90	QT80	ondemand & DTM	QT90	QT80	ondemand & DTM	QT90	QT80	QT70
Before Throttling	Average QoS	0.99	0.89	0.80	0.96	0.902	0.804	1.01	0.896	0.805	0.995	0.903	0.806	0.707
	Standard Deviation of QoS	0.035	0.042	0.037	0.02	0.028	0.043	0.086	0.059	0.065	0.107	0.109	0.075	0.074
Overall Execution	Average QoS	0.85	0.85	0.79	0.871	0.872	0.803	0.85	0.84	0.79	0.723	0.756	0.732	0.695
	QoS Degradation	27.2%	16.6%	**0.3%**	18.3%	11.3%	**0.1%**	28%	18%	4%	35%	30%	22%	7.4%
	Time Spent in Throttling	85.4%	55.5%	**6.8%**	59.4%	48.3%	**0%**	86.9%	61.7%	26.1%	87.7%	86.6%	73.9%	**45.4%**
	Average Power	0.99	0.97	0.88	0.92	0.97	0.86	0.99	0.96	0.89	0.97	1.02	0.97	0.91
	Energy Consumption	1.01	0.98	0.94	0.91	0.97	0.93	1.01	0.98	0.96	1.02	1.01	0.99	0.97
	QoS/Watt	0.99	1.02	1.05	1.09	1.04	1.08	0.99	1.013	1.03	0.997	0.994	1.006	1.03