Playing with power at runtime

Slightly slowed applications, major energy savings

Raphaël BLEUSE,[*] Sophie CERF,[**] Éric RUTTEN[*]

[*]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, France

[**]Univ. Lille, Inria, CNRS, Centrale Lille, CRIStAL, France

Journées calcul et données 2022

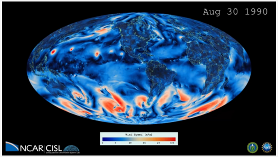
2022-10-12

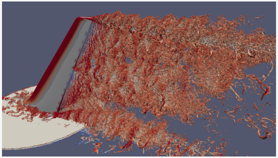
significant contributors:

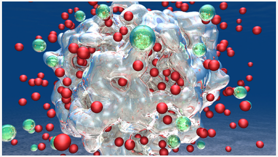
Ismail HAWILA, Swann PERARNAU, Valentin REIS

High-Performance Computing

Applications …







… running on Platforms

@ANL

Energy-aware HPC

@CBS

Performance/Energy Trade-off

Dynamic Management of HPC Systems

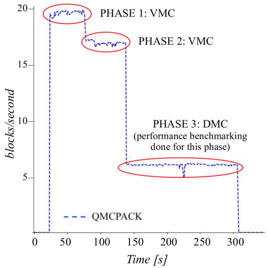
Highly variable systems ...

Offline

- HW spec.
- Aging

At runtime

- Phases
- Failures
- Temperature



(Ramesh et al. 2019)

... require dynamic management

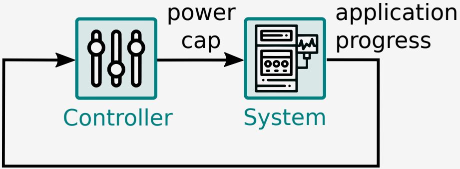
How Scheduling, Autonomic computing, Machine Learning, Feedback **Control Theory** (Hellerstein et al. 2004)

Why Stability, performance guarantees, explainability

Autonomic Computing Approach

The Autonomic Computing approach. . . (Kephart et al. 2003)

- Periodically monitor application progress
- Choose at runtime a suitable power cap for processors



. . . using Control Theory

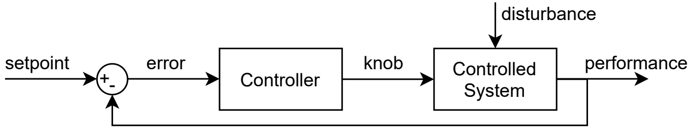
How Low-intrusive supervision

Why Stability, accuracy, transient performance, explainability (Hellerstein et al. 2004)

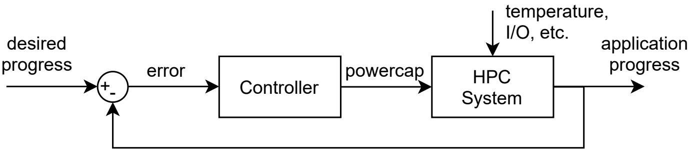
Principle of Control Theory

Feedback loops

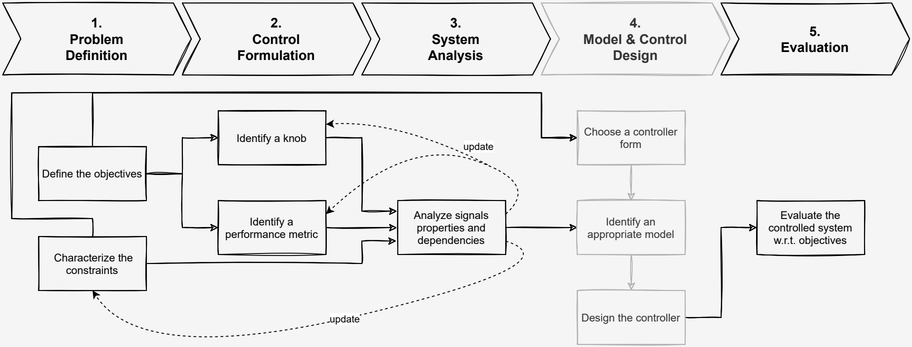
Measure **performance** and react according to the **error** w.r.t. the desired **setpoint** by leveraging system's **knob**.



Power control in HPC



Control Theory Methodology



Dynamic Power Management

Global Objectives

- Sustain execution time
- Minimize energy usage

The Runtime Perspective

- Sustain application progress
- Minimize power usage

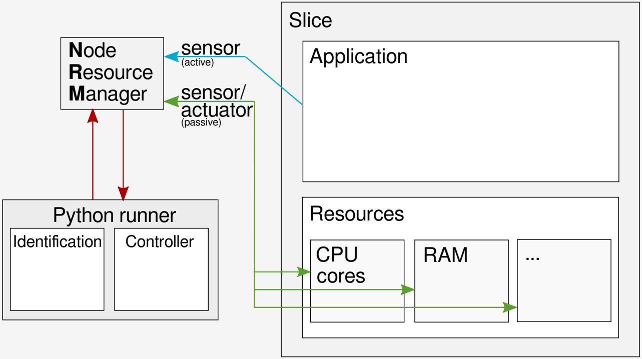
Actuator and Sensor

Power regulation DVFS (Imes et al. 2015; Imes et al. 2019);
DDCM (Bhalachandra et al. 2015);
RAPL (David et al. 2010; Rotem et al. 2012)

App. behavior Measuring progress with heartbeats (Ramesh et al. 2019)

Software Architecture

Software Stack Argo NRM resource management framework



Platform 3 clusters from Grid5000 with various nb. of sockets

Benchmark STREAM (McCalpin 1995)

Signals

Power actuator

RAPL's **power** limitation (David et al. 2010; Rotem et al. 2012)

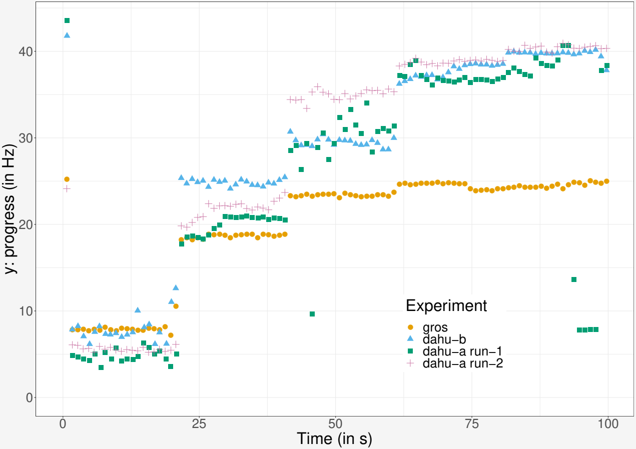
$$\text{pcap}(t_i)$$

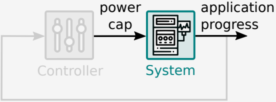
Performance sensor

Application's **progress** (Ramesh et al. 2019)

$$\text{progress}(t_i) = \underset{\forall k,\, t_k \in [t_{i-1}, t_i[}{\text{median}} \left(\frac{1}{t_k - t_{k-1}} \right)$$

Uncontrolled System Analysis (Identification)



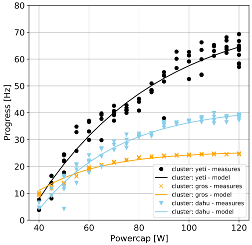


Many Sources of Variations

- Cluster
- Node
- Run
- Exogenous factors (temp., I/O)

Modeling

Static Characteristic (time averaged behavior)



$$\text{progress} = K_L \left(1 - e^{-\alpha(a \cdot \text{pcap} + b - \beta)}\right)$$

a, b: characterizing RAPL actuator

K_L, α, β: cluster- and application-specific

Dynamic perspective

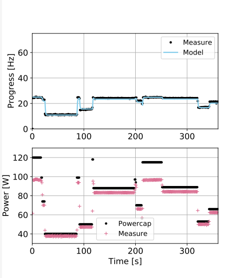
$$\text{progress}_L(t_{i+1}) = \frac{K_L(t_{i+1} - t_i)}{t_{i+1} - t_i + \tau} \cdot \text{pcap}_L(t_i) + \frac{\tau}{t_{i+1} - t_i + \tau} \cdot \text{progress}_L(t_i)$$

Shape set by control theory, parameters optimized offline

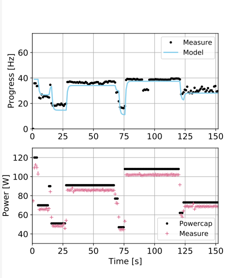
Experimental Evaluation

Measure of the Model Accuracy

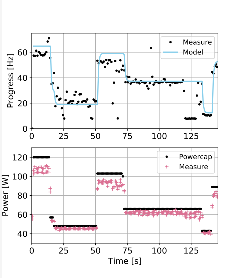
Not a prediction model but used to tune the controller



gros



dahu



yeti

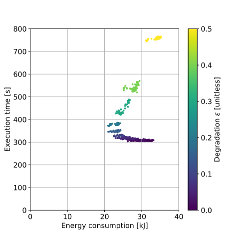
Observations

- Good accuracy.
- The model performs better on clusters with few sockets.

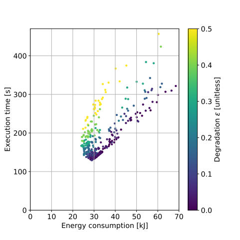
Experimental Evaluation

Post-mortem analysis

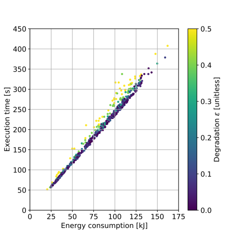
12 degradation levels, min. 30 repetitions each



gros



dahu



yeti

Pareto Front

gros, dahu Family of trade-off from 0% to 15% degradation level
gros with $\epsilon = 0.1$: -22% energy, +7% execution time

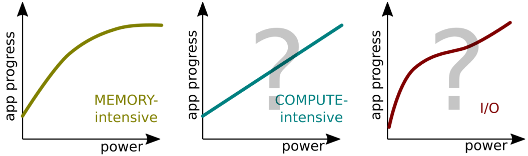
yeti no front, no negative impact of the controller

Perspectives

Applications' phases

Adaptation steps

1. Phases characterization

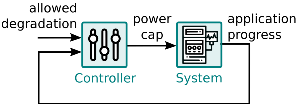


2. Online phase detection (?)
3. Dedicated control: robust, adaptive or hybrid

Conclusion

Objective Reducing energy consumption while sustaining performance

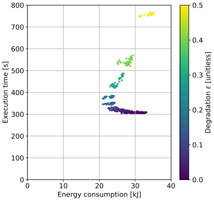
Approach Dynamic power regulation using Control Theory



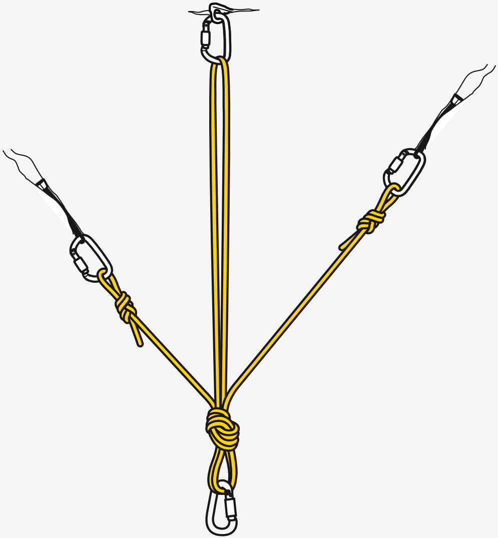
Contributions

- Control theory × HPC systems
- Experimental validation on several clusters

https://doi.org/10.6084/m9.figshare.14754468



Open **post-doc/engineer** positions @CTRL-A!



Backup slides

Related Works

On power regulation in HPC

Different objective or static schema

(Eastep et al. 2017) application-oblivious monitoring

On using control theory for power regulation

Applications: web servers (Abdelzaher et al. 2008), cloud (Zhou et al. 2016), real-time systems (Imes et al. 2015)

Metrics: RAPL (Imes et al. 2019; Lo et al. 2014)
Progress metric (Santriaji et al. 2016)

Model and Controller Parameters

Description	Notation	Unit	gros	dahu	yeti
RAPL slope	a	[1]	0.83	0.94	0.89
RAPL offset	b	[W]	7.07	0.17	2.91
	α	[1/W]	0.047	0.032	0.023
power offset	β	[W]	28.5	34.8	33.7
linear gain	K_L	[Hz]	25.6	42.4	78.5
time constant	τ	[s]	1/3	1/3	1/3
	τ_{obj}	[s]	10	10	10
lower power limit	$pcap_{min}$	[W]	40	40	40
higher power limit	$pcap_{max}$	[W]	120	120	120

PI Controller Parameters Computation

K_P and K_I are based both on the model parameters K_L and τ and on a tunable parameter τ_{obj} (Åström et al. 1995):

$$K_P = \tau / (K_L \cdot \tau_{\text{obj}})$$

$$K_I = 1 / (K_L \cdot \tau_{\text{obj}})$$

with τ_{obj} defining the desired dynamical behavior of the controlled system.

The controller is chosen to be nonaggressive:

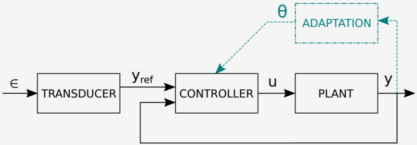
$$\tau_{\text{obj}} = 10\,\text{s} > 10\tau$$

Model Reference Adaptive Control

Model $y(k+1) = b_0 u(k) + a_0 y(k)$

Parameter vector $\theta = [s_0]$

Regression vector $\phi(k) = [y(k)]$



Control Law (Akhtar et al. 2005)

$$u(k) = -\frac{1}{b_0}\left[\phi^T(k)\hat{\theta}(k) - b_m y_{\text{ref}}\right]$$

$$\hat{\theta}(k) = \hat{\theta}(k-1) + \frac{1}{\phi^T(k-1)\phi(k-1)}\Big[a_m y(k-1) - b_0 u(k-1) - \phi^T(k-1)\hat{\theta}(k-1)\Big]\phi(k-1)$$

References I

T. Abdelzaher et al., “Introduction to Control Theory And Its Application to Computing Systems,” in *Performance Modeling and Engineering* (Springer, 2008), pp. 185–215.

S. Akhtar and D. S. Bernstein, “Lyapunov-stable discrete-time model reference adaptive control,” Int. J. Adapt. Control Signal Process. **19**, 745–767 (2005).

K. J. Åström and T. Hägglund, *PID Controllers: Theory, Design, and Tuning*, Second (International Society of Automation, 1995).

S. Bhalachandra et al., “Using Dynamic Duty Cycle Modulation to Improve Energy Efficiency in High Performance Computing,” in IPDPS workshops (May 2015), pp. 911–918.

H. David et al., “RAPL: Memory Power Estimation and Capping,” in ISLPED (2010), pp. 189–194.

J. Eastep et al., “Global Extensible Open Power Manager: A Vehicle for HPC Community Collaboration on Co-Designed Energy Management Solutions,” in ISC, Vol. 10266, Lecture Notes in Computer Science (June 2017), pp. 394–412.

J. L. Hellerstein et al., *Feedback Control of Computing Systems*, (Wiley, 2004).

C. Imes et al., “CoPPer: Soft Real-time Application Performance Using Hardware Power Capping,” in ICAC (2019), pp. 31–41.

References II

C. Imes et al., "POET: A Portable Approach to Minimizing Energy Under Soft Real-time Constraints," in RTAS (2015), pp. 75–86.

J. O. Kephart and D. M. Chess, "The Vision of Autonomic Computing," IEEE Computer **36**, 41–50 (2003).

D. Lo et al., "Towards Energy Proportionality for Large-Scale Latency-Critical Workloads," in ISCA (June 2014), pp. 301–312.

J. D. McCalpin, "Memory Bandwidth and Machine Balance in Current High Performance Computers," IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter (1995).

S. Ramesh et al., "Understanding the Impact of Dynamic Power Capping on Application Progress," in IPDPS (2019), pp. 793–804.

E. Rotem et al., "Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge," IEEE Micro **32**, 20–27 (2012).

M. H. Santriaji and H. Hoffmann, "GRAPE: Minimizing Energy for GPU Applications with Performance Requirements," in MICRO (Oct. 2016), 16:1–16:13.

Y. Zhou et al., "CASH: Supporting IaaS Customers with a Sub-core Configurable Architecture," in ISCA (June 2016), pp. 682–694.