Towards control of task-level & cluster-level in the PULSE project

Sophie Cerf[1] & Eric Rutten[2]

[1]Inria Spirals

[2]Inria Ctrl-a

VELVET days

2023-12-14 Nantes

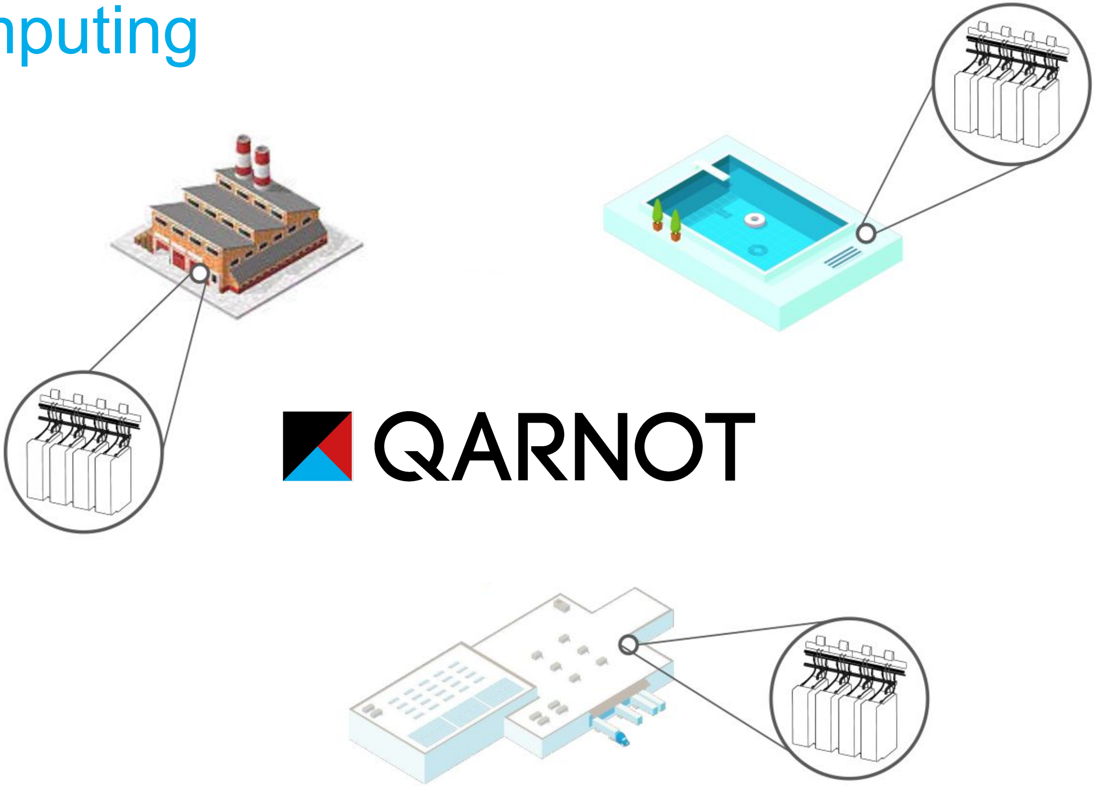
PULSE

PUshing Low-carbon Services towards the Edge

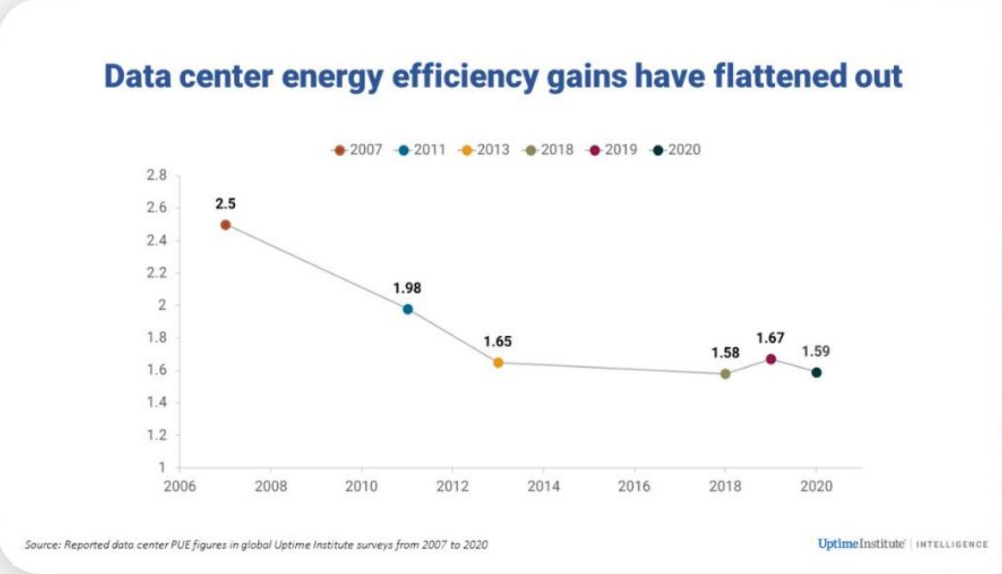
Défi Inria - Qarnot

Teams: Avalon, **Ctrl-A**, **Spirals**, Stack, Storm, Topal

Qarnot Computing

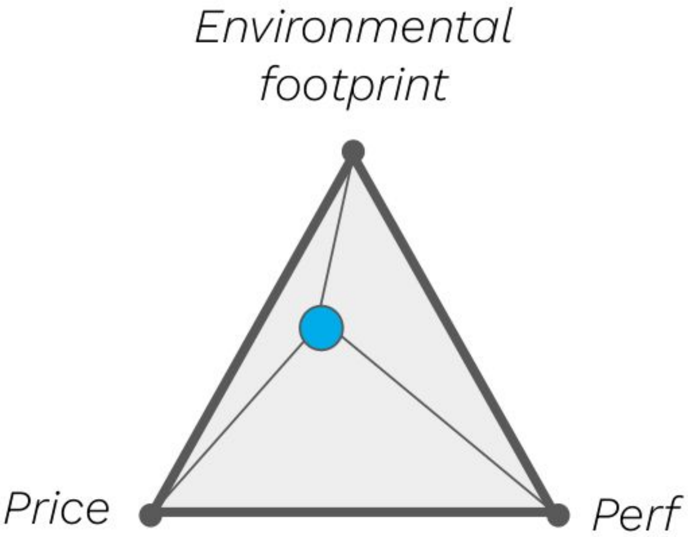


Challenge


Data center energy efficiency gains have flattened out
2007 2011 2013 2018 2019 2020
2.8
2.6
2.4
2.2
2
1.8
1.6
1.4
1.2
1
2.5
1.98
1.65
1.58
1.67
1.59
2006 2008 2010 2012 2014 2016 2018 2020
Source: Reported data center PUE figures in global Uptime Institute surveys from 2007 to 2020
UptimeInstitute | INTELLIGENCE

PULSE’s Ambition

Yesterday



Tomorrow

Objectives

Axis 1: Analysis of the environmental impact of distributed computing

Project 1 - Modeling

Project 2 - Instrumentation, estimation and monitoring

Axis 2: Implementing greener computing services

Project 3 - Software-hardware matching

Project 4 - Data storage

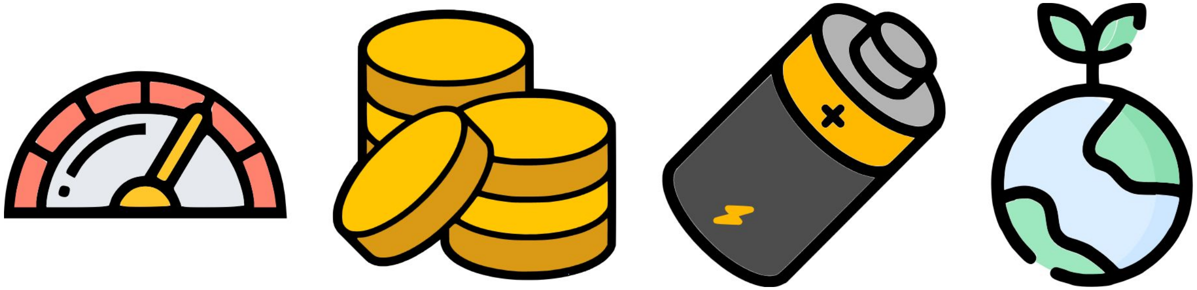
Project 5 - Control at task level

Project 6 - Control at cluster level

Control at the task-level

Control at the task level

- Objectives



- Sensors ?
- Actions ?
- Controller ?

Control at the task level

- Background on power regulation

- Objective in terms of allowed performance degradation

Sustaining Performance While Reducing Energy Consumption: A Control Theory Approach

Sophie Cerf[1](✉), Raphaël Bleuse[1], Valentin Reis[2], Swann Perarnau[2], and Éric Rutten[1]

[1] Université Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
{sophie.cerf,raphael.bleuse,eric.rutten}@inria.fr
[2] Argonne National Laboratory, Lemont, USA
{vreis,swann}@anl.gov

Abstract. Production high-performance computing systems continue to grow in complexity and size. As applications struggle to make use of increasingly heterogeneous compute nodes, maintaining high efficiency (performance per watt) for the whole platform becomes a challenge. Alongside the growing complexity of scientific workloads, this extreme heterogeneity is also an opportunity: as applications dynamically undergo variations in workload, due to phases or data/compute movement between devices, one can dynamically adjust power across compute elements to save energy without impacting performance. With an aim toward an autonomous and dynamic power management strategy for current and future HPC architectures, this paper explores the use of control

Dynamic Management of HPC Systems

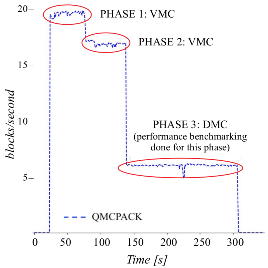
Highly variable systems ...

Offline

- HW spec.
- Aging

At runtime

- Phases
- Failures
- Temperature



(Ramesh et al. 2019)

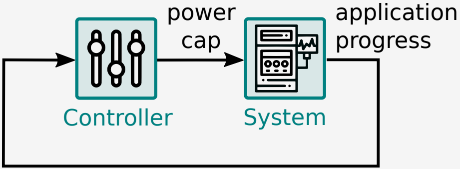
... require dynamic management

How Scheduling, Autonomic computing, Machine Learning, Feedback **Control Theory**

Autonomic Computing Approach

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

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

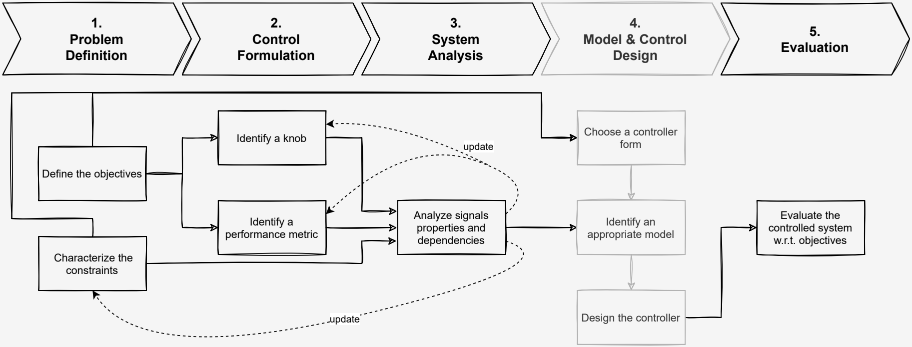


…using Control Theory

How Low-intrusive supervision

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

Control Theory Methodology



Signals

Power actuator

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

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

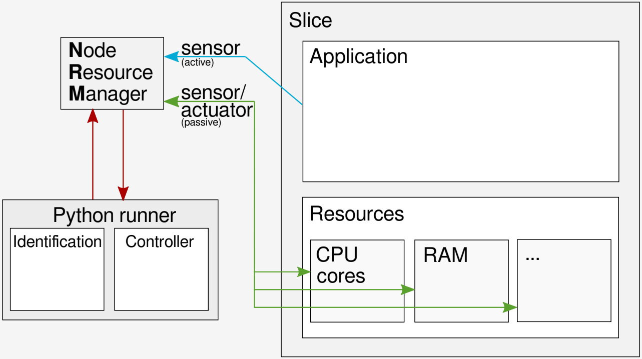
Performance sensor

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

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

Software Architecture

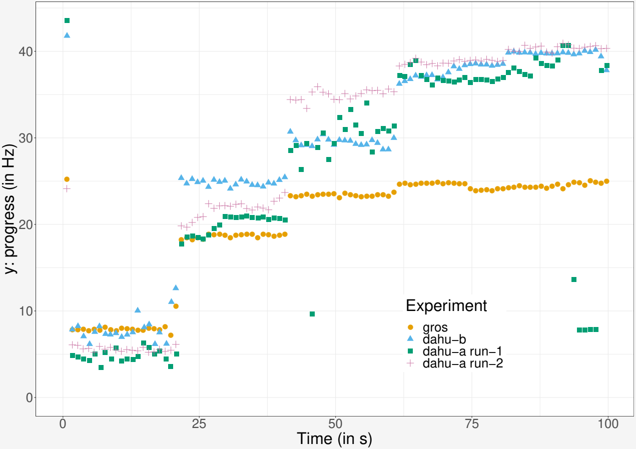
Software Stack Argo NRM resource management framework

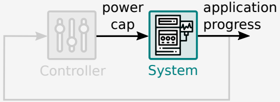


Platform 3 clusters from Grid5000 with various nb. of sockets

Benchmark STREAM (McCalpin 1995)

Uncontrolled System Analysis (Identification)



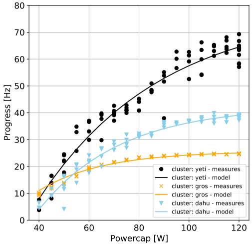


Many Sources of Variations

- 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

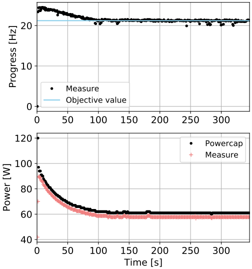
$$\text{progress}_L(t_{i+1}) = \frac{K_L(t_{i+1} - t_i)}{t_{i+1} - t_i + \tau} \cdot \text{pcap}_L(t_i) + \frac{\tau}{t_{i+1} - t_i + \tau} \cdot \text{progress}_L(t_i)$$

Shape set by control theory, parameters optimized offline

Experimental Evaluation

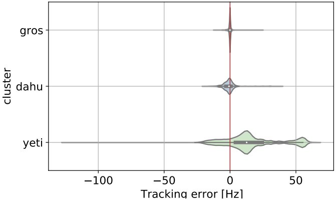
Time-local behavior

Illustration



- Progress reaches the objective level $\epsilon = 0.15$

Analysis



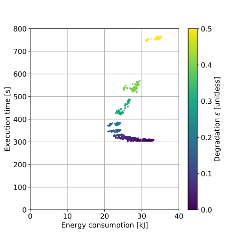
gros, dahu unimodal, centered near 0, narrow dispersion

yeti 2nd mode (model limitations at approx. 10Hz)

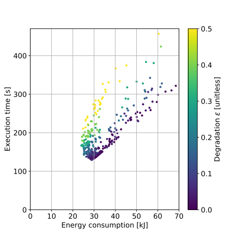
Experimental Evaluation

Post-mortem analysis

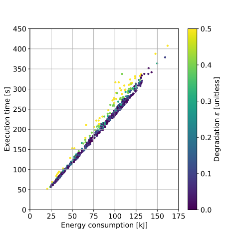
12 degradation levels, min. 30 repetitions each



gros



dahu



yeti

Pareto Front

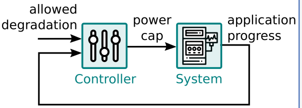
gros, dahu Family of trade-off from 0% to 15% degradation level
gros with $\epsilon = 0.1$: -22% energy, +7% execution time

yeti no front, no negative impact of the controller

Conclusion

Objective Reducing energy consumption while sustaining performance

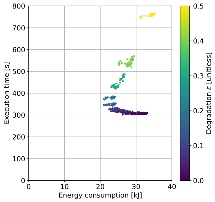
Approach Dynamic power regulation using Control Theory



Contributions

- Control methodology for HPC systems
- Offline model identification
- Controller design
- Euro-Par Artifact Evaluated Experimental validation on several clusters

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



Control at the task level

Perspectives

- System
  - handle different benchmarks, phased applications
  - towards distributed and heterogeneous systems
- Sensors
  - avoid instrumenting applications
  - measure other aspects: contention, etc.
  - include carbon measures
- Actuators
  - beyond power cap
  - VM, overcommitment
- Controller
  - allow setting carbon budget

Towards control of task-level & cluster-level in the PULSE project

Eric Rutten
Ctrl-A team, Inria / LIG, Grenoble, France

Energy-Efficient runtime hybridization of heterogeneous computing tasks

- FaaS (Function-as-a-Service)

greatly diversifying typology of computing models

from continuous data stream (100ms) to HPC (mn, days)

contraints on physical location, initialization times

strong variations along time in needs for computing resources

→ cohabitation of models can favor resources exploitation

- Runtime autonomic management in a loop at <u>cluster level</u>

Control Theory leveraged to optimize pooling heterogeneous tasks

wrt users multiple objectives : price, QoS, environmental footprint

→ minimal impact within user-spec graceful degradation

Approach (i) : environmental impact trade-offs

- environmental concerns of providers as well as users
- identify and design mechanisms for trade-off

    between cost, quality (with an acceptation of perturbation)

    and environmental impact (improvement of resource use)

- resource harvesting, allocation to flexible tasks

    injection regulated to avoid or minimize perturbation

    acceptable level itself dynamically adaptable (e.g., emergencies)

- equipment aging impacted by maximal resource use

    integrated in the taking into account of environmental impact

Approach (iia) : autonomic resource mgmt

- **Background** on HPC resource harvesting

  tasks partitioned into prioritary and "best-effort"

Control-based runtime management of HPC systems with support for reproducible experiments

Thesis Defense

Quentin GUILLOTEAU

2023-12-11

Univ. Grenoble Alpes, INRIA, CNRS, LIG

Quentin.Guilloteau@univ-grenoble-alpes.fr

→ **PhD Q. Guilloteau** dec. 11, 2023

Approach (iib) : autonomic resource mgmt

- **Harvesting :** **injecting "best-effort" between prioritary**

HPC Jobs: **Stones**

- Some computations
- Static resource allocation
- Static time allocation

HPC Cluster: **Jar**

- Resources (CPU cores)
- Time (seconds)

Gantt Chart

Idle Resources = Lost Computing Power (and Money)

Approach (iib) : autonomic resource mgmt

- **Autonomic loop :** **injecting without perturbating**

Problem

$\nearrow$ Harvesting $\Longrightarrow$ $\nearrow$ Perturbations (e.g., I/O) $\rightsquigarrow$ **Trade-off**

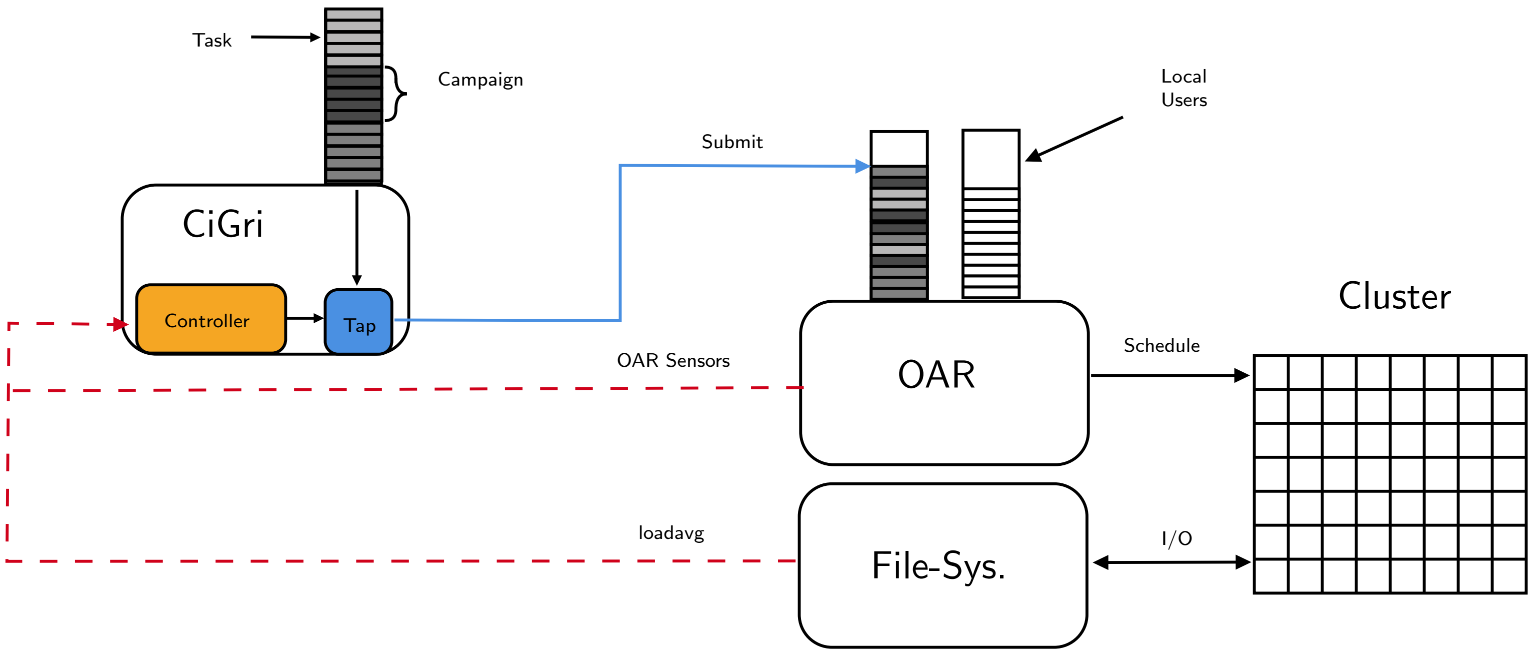
$\hookrightarrow$ Unpredictability $\Longrightarrow$ **runtime management**

This Thesis

1. **Regulate** the injection of CiGri jobs to harvest idle resources with **contolled** disturbances
2. Investigate ways to improve the cost and **reproducibility** of experiments

Approach (iic) : autonomic resource mgmt

- **Control loop :** **injecting without perturbating**



Reference value $\rightsquigarrow$ acceptable load on the File-System $\simeq$ overhead on I/O ops

Approach (iid) : autonomic resource mgmt

- **PI(D) Controller :** sensors, actuators, control law

What are we looking for?

First, a Model ... (i.e., how does the system behave without Control)

$$\mathbf{y}(k+1) = \sum_{i=0}^{k} a_i \times \mathbf{y}(k-i) + \sum_{j=0}^{k} b_j \times \mathbf{u}(k-j)$$

... then a (PID) Controller (i.e., the Closed-Loop behavior)

$$\mathbf{u}(k) = \mathbf{K}_p \times Error(k) + \mathbf{K}_i \times \sum_{i}^{k} Error(i) + \mathbf{K}_d \times (Error(k) - Error(k-1))$$

Sensors & Actuators

- Actuator: #jobs to sub $\rightsquigarrow$ $\mathbf{u}$
- Sensor: FS Load $\rightsquigarrow$ $\mathbf{y}$
- $Error(k) = Reference - Sensor(k)$

Method

1. Open-Loop experiments (fixed $\mathbf{u}$)
2. Model parameters (a_i, b_j)
3. Choice controller behavior $(\mathbf{K}_*)$

Approach (iie) : autonomic resource mgmt

- **Controller tuning :** choose gains & behavior

Closed-Loop Behavior

Controller Gains are ...
functions of the model and

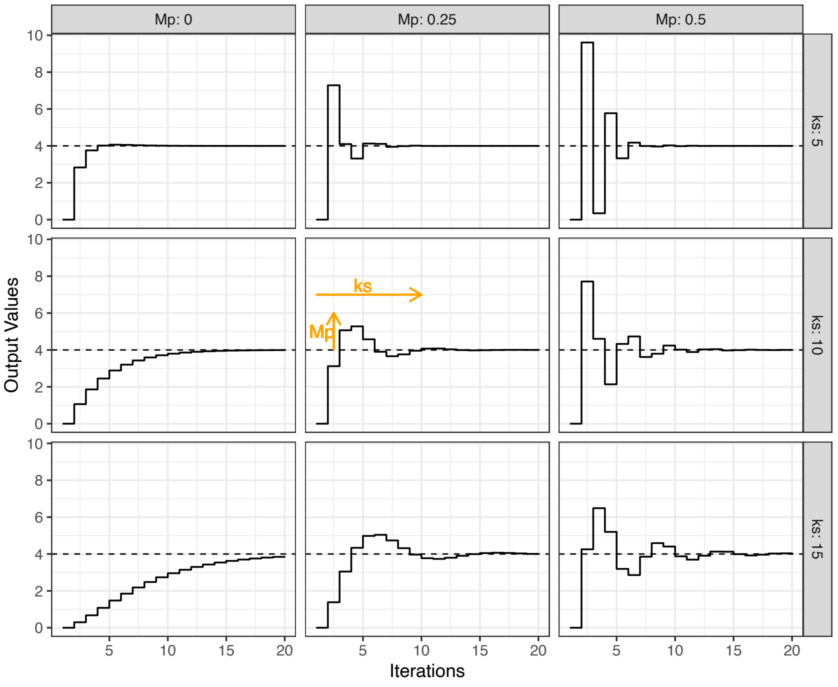
- k_s: max **time** to steady state
- M_p: max **overshoot** allowed

You choose the behavior!

Non-Intrusive Harvesting

- no overshoot
- but "fast" response

Closed loop behaviour of our system for different values of (ks, Mp)

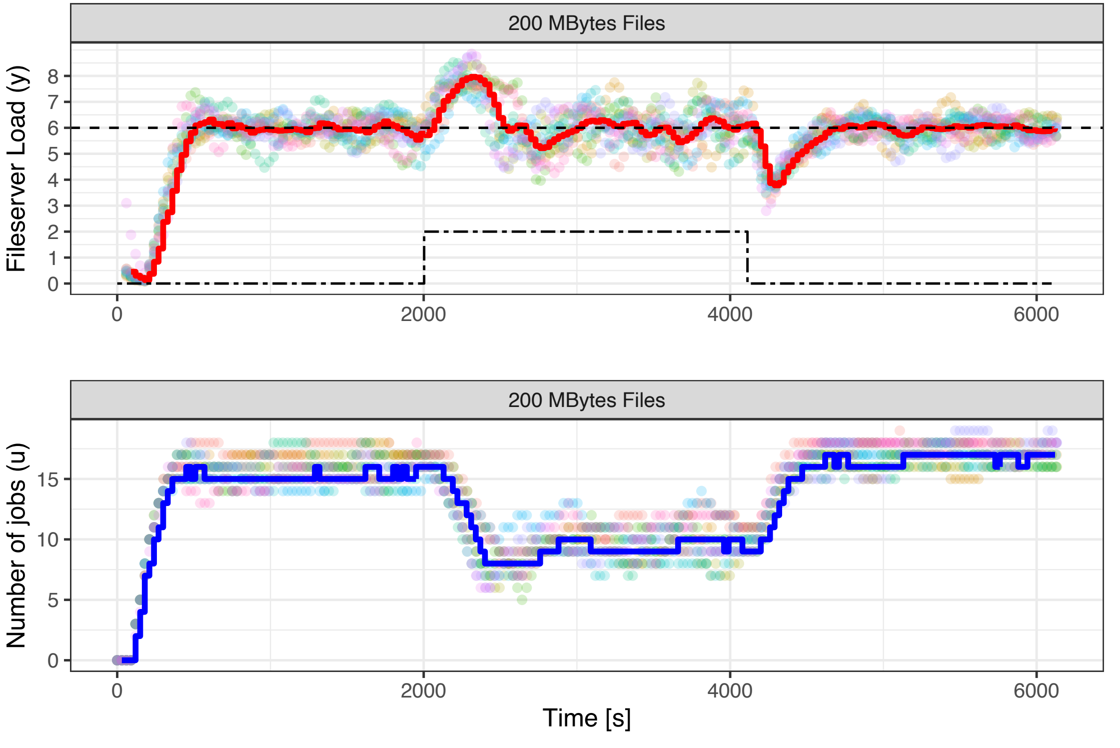


Approach (iif) : autonomic resource mgmt

- **Controller validation**

Evaluation with Synthetic Jobs

Response of the Controlled System to a Step Perturbation



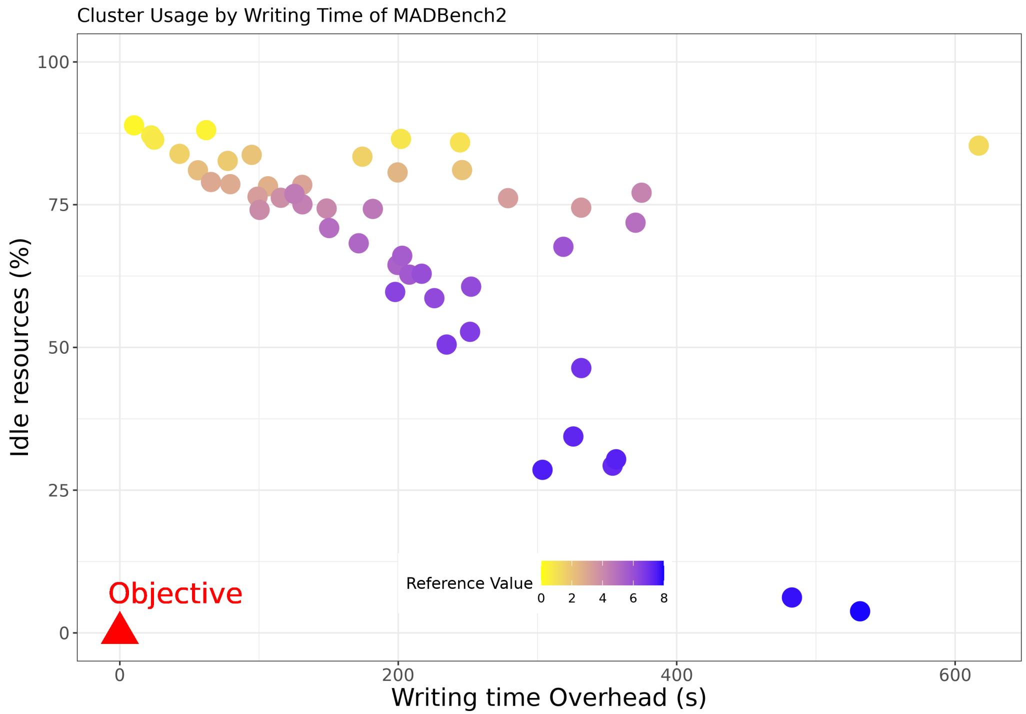
- constant reference
- synthetic jobs
- step disturbance

Manage to regulate the load of the File-System

Approach (iig) : autonomic resource mgmt

- **Controller & trade-offs**

Trade-off: Harvesting vs. Overhead



- MADBench2 [Bor+07]
- different references
- compute idle resources
- compute I/O overhead

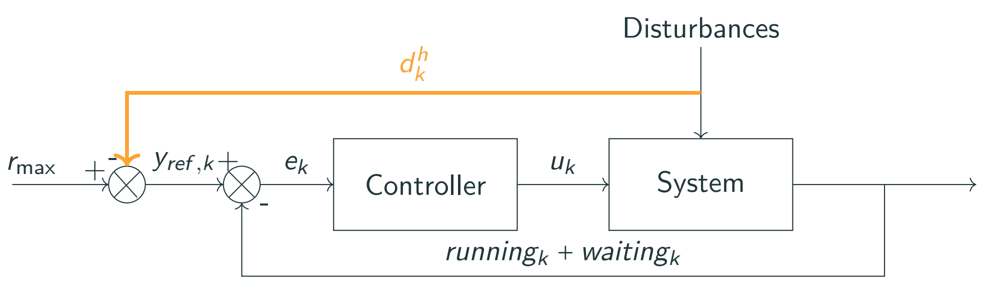
Trade-off between Harvesting & I/O overhead through the reference value

Approach (iih) : autonomic resource mgmt

- **Coordinating Controller & Scheduler :** **first touch**

Beyond Idle Resources

Lost computing power: Idle resources, but also **killed jobs**!



- **anticipate** variations in available resources
- new sensor (modify OAR)
- previsionnal Gantt chart
- **horizon**

Can reduce **both idle and killed** time, and energy usage!

Approach (iii) : extensions in the Pulse project

- generalized to a greater diversity of types of
  - Computation tasks/jobs : more than just 2 (Best Effort/Prio)
  - Resources : computing/comunication/memory
    within limits (themselevs variable)
  - Applications : e.g., HPC, FaaS, stream, …
    with phases of different consumption, requirement, …

- coordinating loops :
  - Control & Scheduling :
    info from scheduler to feed-Forward control
    regulating Scheduler input w.r.t. complexity
    with or w/o preemption (abort/suspend)

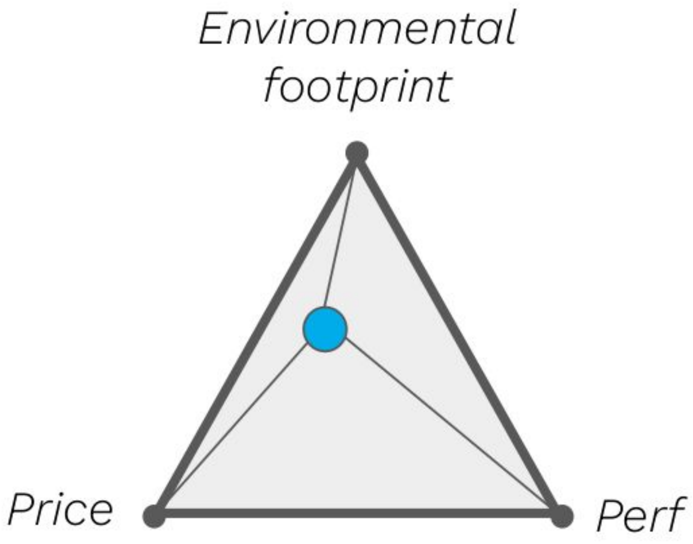
  - Coordinating loops with different functionalities
    - Functionalities, better separated for simplicity
      - Self-optimization, self-configuration : placement & scheduling
      - Self-healing, self-protection : resilience & cybersecurity
    - Coordination, for coherence and efficiency
      - Resilience by adapting placement/scheduling

- Real world targets @Qarnot computing

Conclusion

- Control at task-level
  - From allowing perf degradation by playing with power
  - Towards carbon-driven computation without instrumenting applications
- Control at cluster-level
  - From : injecting small Best-Effort jobs between prioritary scheduled large tasks
  - Towards : more diversity of cohabitating jobs, control combined with scheduling
    with user-decided trade-off perf(QoS)/impact (dynamically)

To ensure computing within limits



Users' objectives