Autonomic Approach to the Runtime Management of HPC Clusters Resources

Quentin GUILLOTEAU, Olivier RICHARD, Eric RUTTEN

Univ. Grenoble Alpes, Inria, CNRS, LIG, F-38000 Grenoble France

Context & Applicative Problem

Gricad

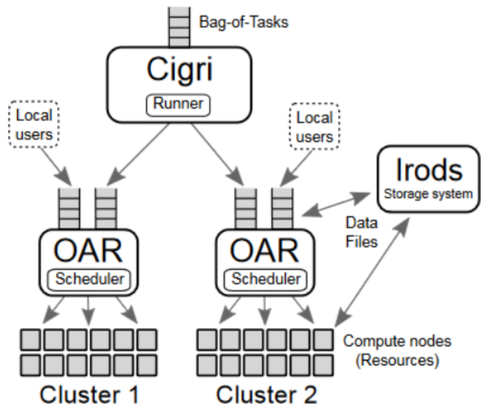
- Grenoble Mesocenter
- 169 TFLOPS
- 5700 CPUs on **several clusters**
- $>$ 1200 users

Idle HPC Resources $\Longrightarrow$ Lost Computing Power
$\hookrightarrow$ **How to Harvest?**

Various Job Behaviours
$\hookrightarrow$ Variability $\Longrightarrow$ Requires **Runtime Management**

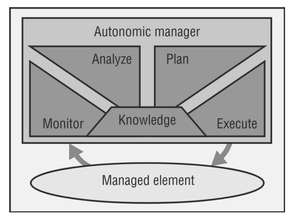
A Solution: *CiGri*

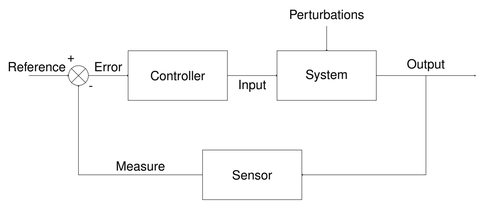
- **bag-of-tasks**: many, multi-parametric jobs
- **Best-effort Jobs**: Lowest priority
- **Objective**: Collect grid idle resources in a non-intrusive way for the users



MAPE-K Loop & Control Theory

Auto-regulating Dynamical Systems given **high-level objectives**
$\hookrightarrow$ Control Theory as an Interpretation of the MAPE-K Loop





Control Theory drives system to a desired state by adapting inputs while **minimizing delay, overshoot, or steady-state error** and ensuring control **stability**

Perturbations from Harvesting

Cannot simply submit huge amount of jobs.
$\nearrow$ Harvesting $\Longrightarrow$ $\nearrow$ Perturbations (e.g., I/O)
$\hookrightarrow$ **Trade-off to exploit**

Instrumentation

- Actuator: #jobs to submit from CiGri to OAR
- Sensors: OAR Waiting Queue length
  File-System Load (`loadavg`)

Controllers

Controller = function between objective (or *Reference*) error and system input.

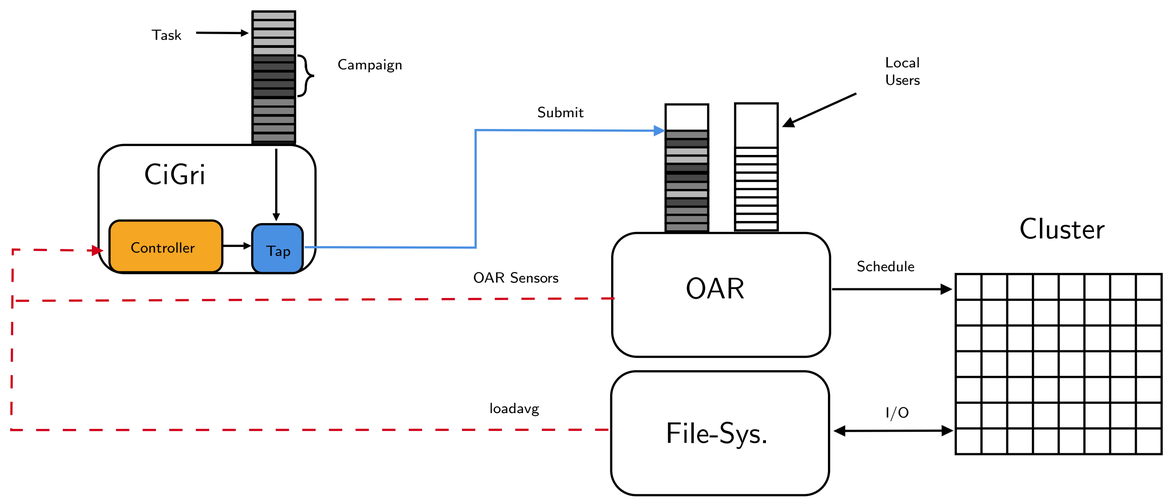
$$Error_k = Reference - \textbf{Sensor}_k$$

Different types of Controllers (PID, RST, MFC)

$$\textbf{Actuator}_{k+1} = \mathbf{K}_p \times Error + \mathbf{K}_i \times \sum_k Error_k + \mathbf{K}_d \times (Error_k - Error_{k-1})$$

Research Problem

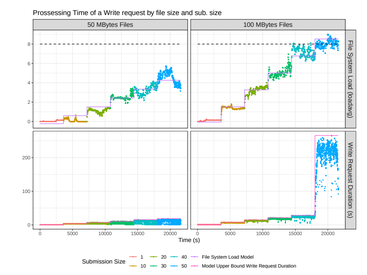
Use Control Theory tools to dynamically **harvest idle HPC resources** in a **non-intrusive** way for the users



Methodology

Open-Loop Experiments $\Longrightarrow$ Find Model (1st order) $\mathbf{y}(k+1) = a\mathbf{y}(k) + b\mathbf{u}(k)$ $\Longrightarrow$ Choice of Closed-Loop Behaviour $\Longrightarrow$ Deduce Controller Gains $\mathbf{K}_p, \mathbf{K}_i, \mathbf{K}_d$,

Open-Loop & Identification



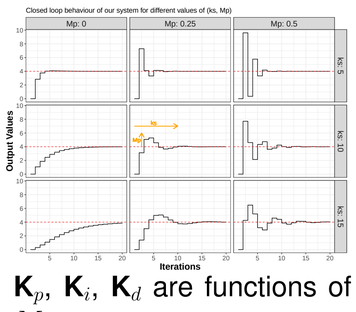
1. Vary nb jobs submitted
2. Look at FS load
3. Model

$\hookrightarrow$ Saturation when File-System Load > 8

Closed-Loop Behaviour

Can pick the **desired system behaviour**:

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



Non-Intrusive Harvesting means:

- no overshoot
- "fast" response

$\mathbf{K}_p$, $\mathbf{K}_i$, $\mathbf{K}_d$ are functions of the model, k_s and M_p

Response to Perturbations

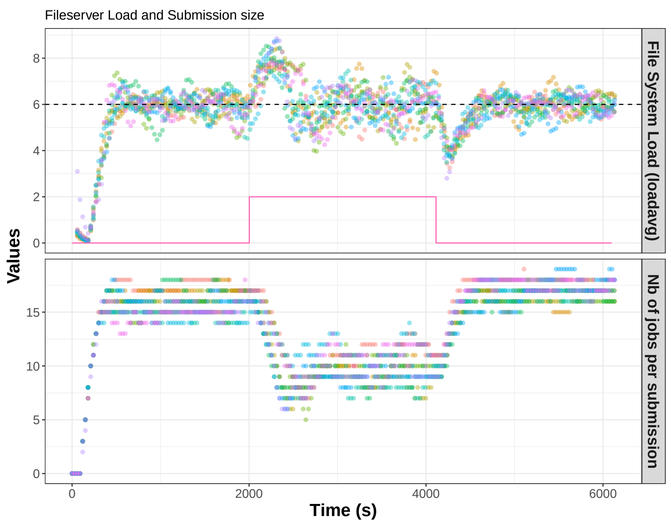


Figure: Response of the closed-loop system (top) to a synthetic step perturbation to keep the load of the File-System around the value 6. Bottom is the number of jobs submitted by CiGri.

Exploiting the Trade-Off

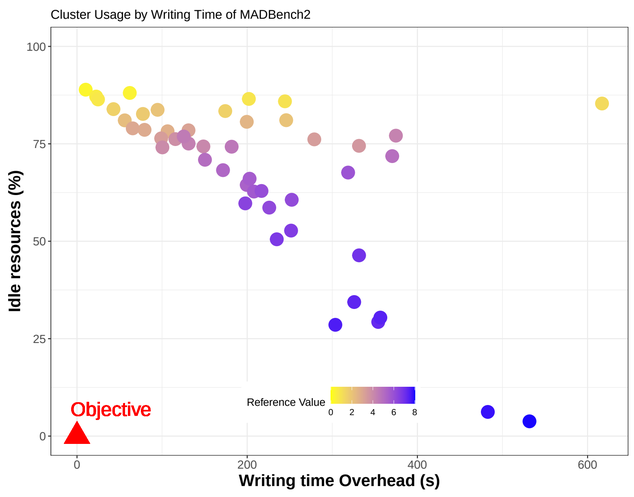


Figure: Writing Overhead of an application (MADbench2 Benchmark) with CiGri based on Reference Value

$\hookrightarrow$ We have **Control over the Degradation and Harvesting** with the choice of Reference Value

Perspectives

- Adaptive Controllers to deal with different I/O Profiles
- Interaction with Scheduler to anticipate the arrival of new Priority jobs
- Minimize the amount of Best-effort jobs killed (also represents lost computing power)
- Reproductible Experiments with Nix and NixOS

References

Q. Guilloteau, O. Richard, B. Robu and E. Rutten. *Controlling the Injection of Best-Effort Tasks to Harvest Idle Computing Grid Resources*, ICSTCC 2021, hal-03363709

Acknowledgements: Experiments presented in this poster were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr).

CTRL-A
Control for
safe Autonomic computing systems

Laboratoire d'Informatique de Grenoble

2022