Composition of Scheduling and Control-Theory Techniques

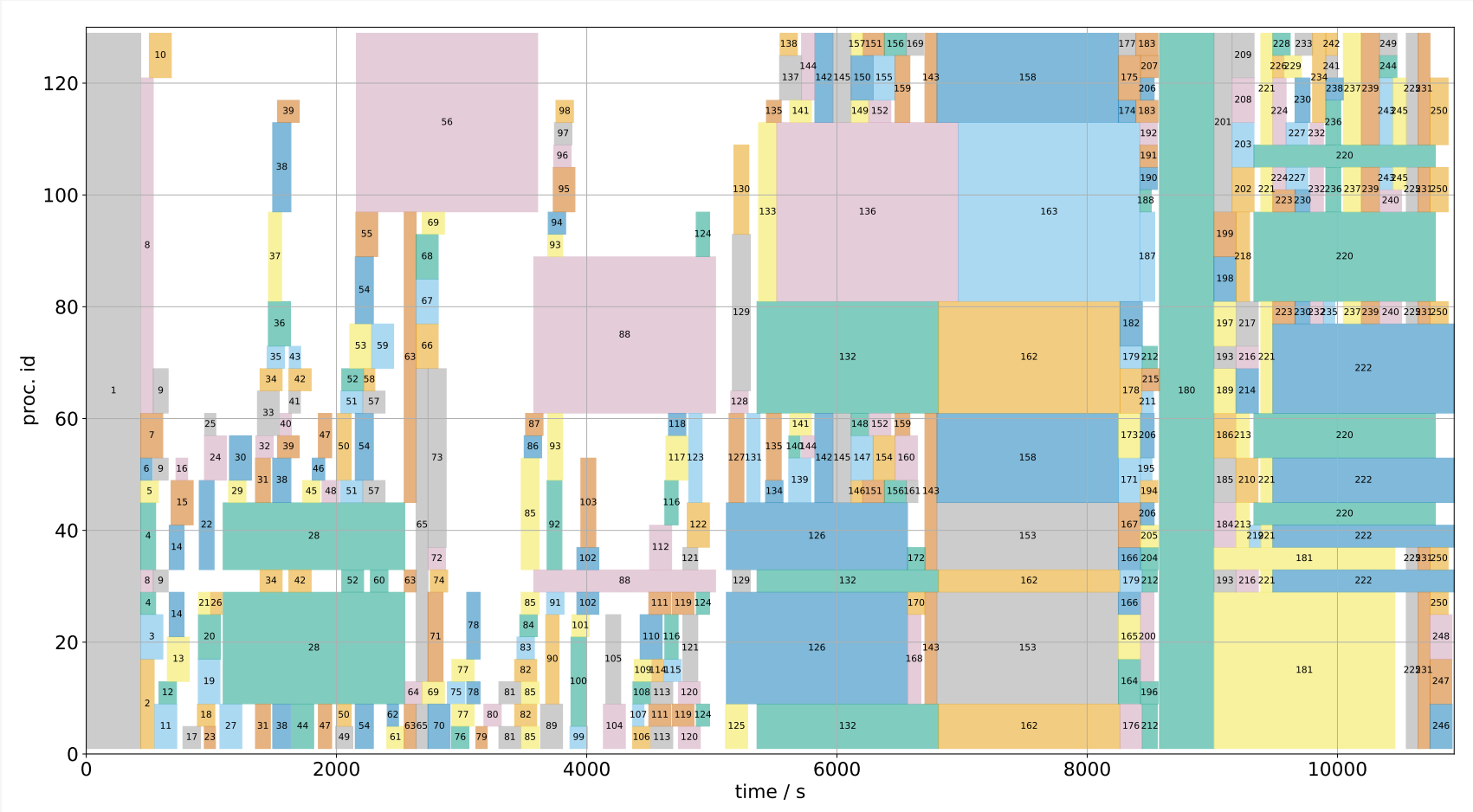
Raphaël BLEUSE

Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, France

New Challenges in Scheduling Theory
Aussois 2024

2024-05-13

Scheduling (~ Resource Allocation)



platform tasks/jobs objective

$\Longrightarrow$ where? & when?

Coping with reality?

complex systems

- network topology: fat tree, $\star$-fly, torus
- intricate software stacks: kernel, runtimes, application code

complex behaviors

- provisioning/deprovisioning jobs
- sensitive dependence on initial conditions

extrinsic variability

- offline: hardware spec., aging, ...
- online: phases (e.g., I/O, compute), adaptive routing, failures, temperature, ...

Simplistic ∧ Tangled Models

some answers

- ignore the "complex" parts: $1 \mid\mid C_{\max}$, $P \mid\mid C_{\max}$, ...
- offline *vs.* online *vs.* clairvoyant
- introducing uncertainty: stochastic models

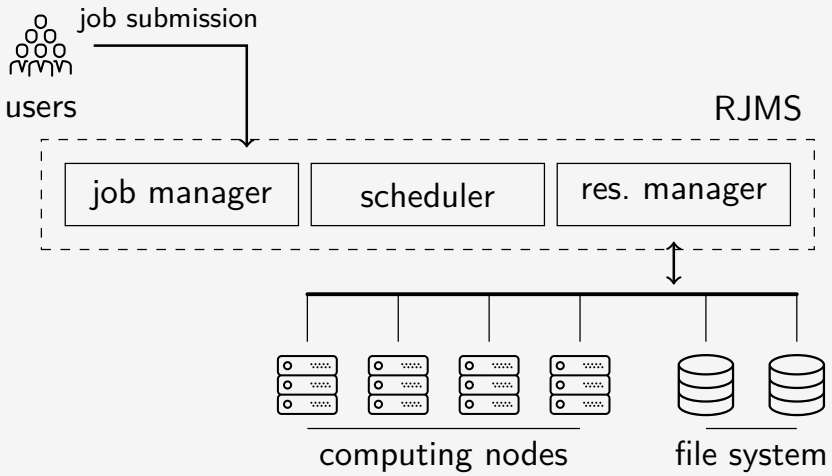
rough spots

- non-representative models (ivory tower?)
- non-instantiable models
- time & dynamics of the system

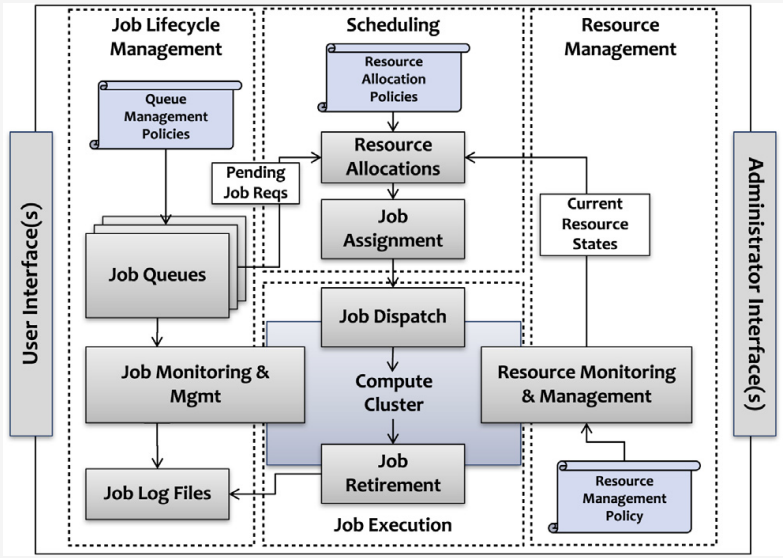
$\Rightarrow$ hard to maintain guarantes

Scheduler/RJMS structure

Let's take a step back, and consider the *whole* system.



(Georgiou 2010)



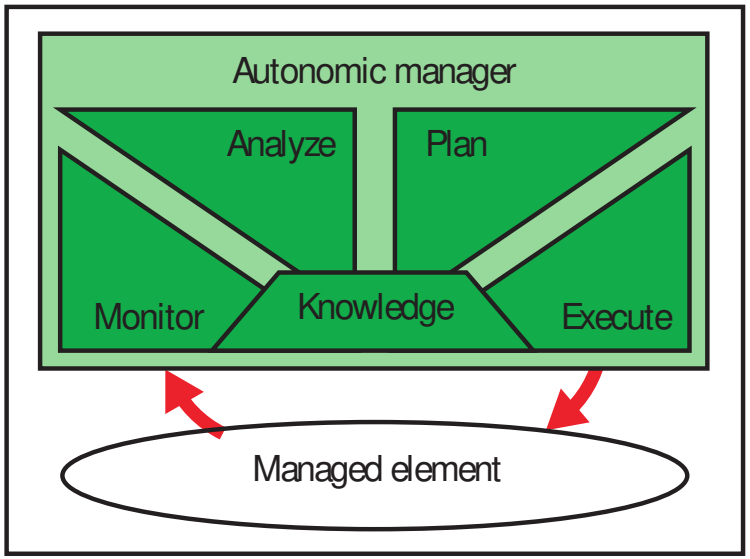
(Reuther et al. 2018)

$\Rightarrow$ open-loop (no feedback) *vs.* closed-loop

Feedback loops (*à la* Control Theory)

Autonomic Computing (Kephart et al. 2003)

1. monitor system state
2. choose/adapt (online) parameters
3. `GOTO 1`



Control Theory

Challenges

- stability, accuracy, explainability
- transient *vs.* steady-state behavior

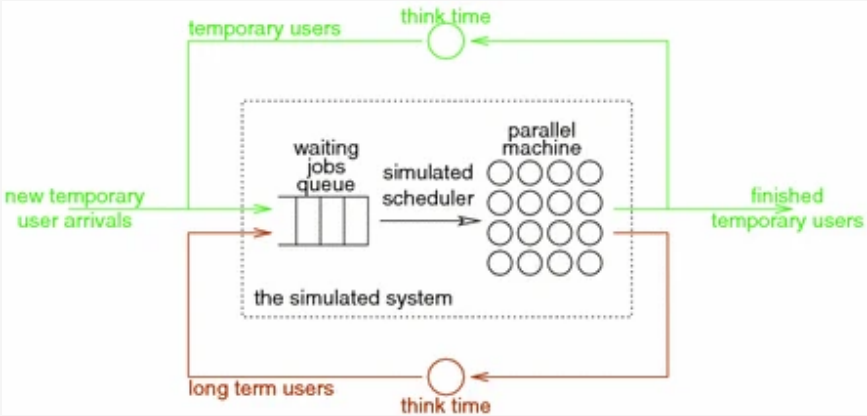
Gains

- simpler models
- adapt to actual behavior

Unexploited loops

Most scheduling approaches rely on an *unrealist* open-loop model.
They struggle/fail to catch for example:

- congestion
- user behavior (Feitelson 2016; Madon et al. 2022)



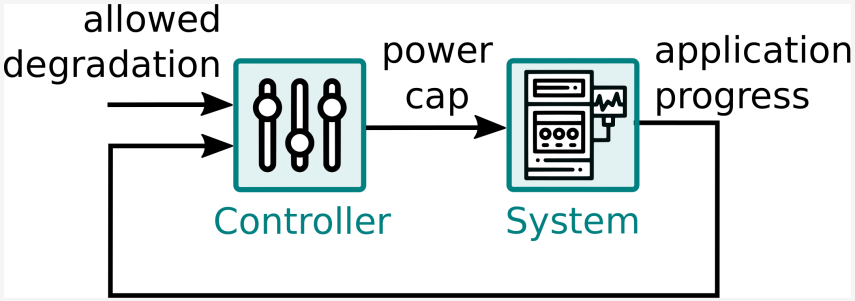
Classic workarounds: "cyclic" scheduling, more variables (increases models' complexity), ...

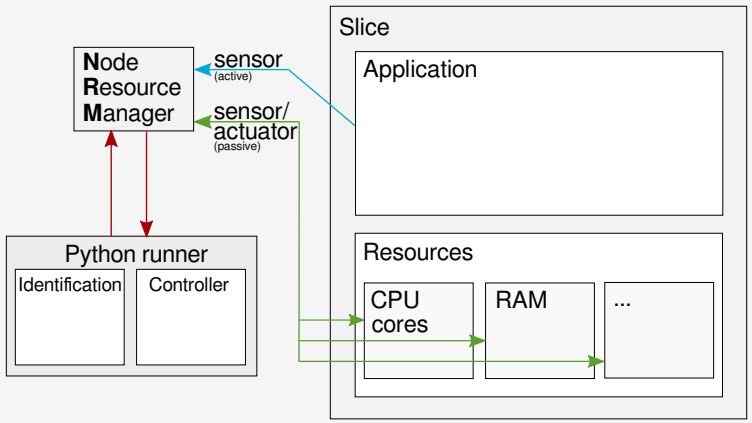
Existing usages/compositions: RAPL×NRM

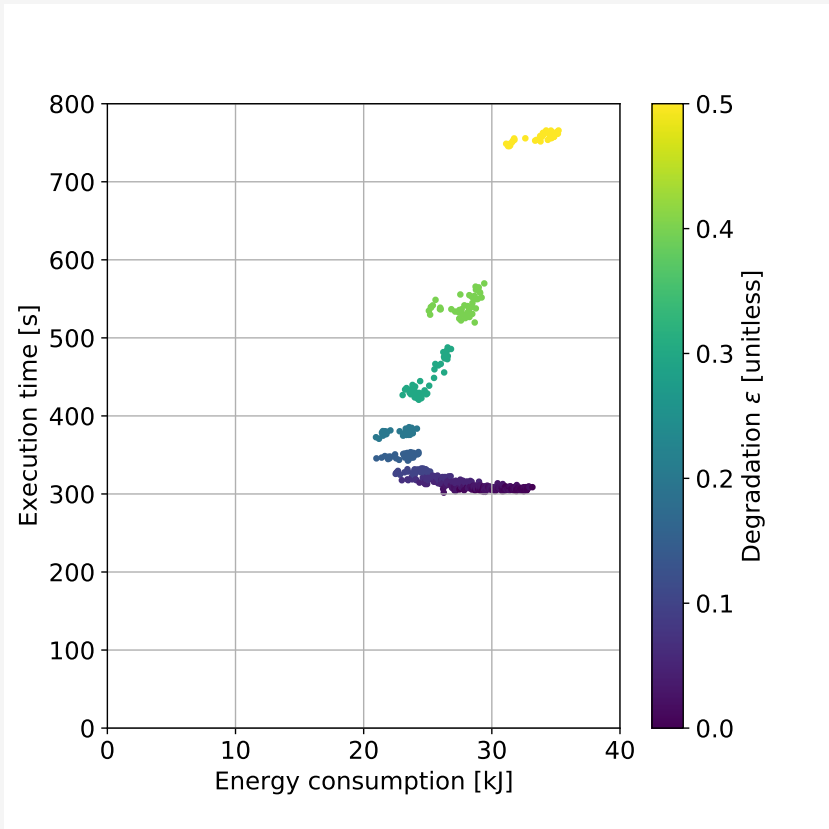
(Cerf et al. 2021; Hawila et al. 2022), ongoing collab. ANL/Inria

Objective Reducing energy consumption while sustaining performance

Approach Dynamic power regulation using Control Theory





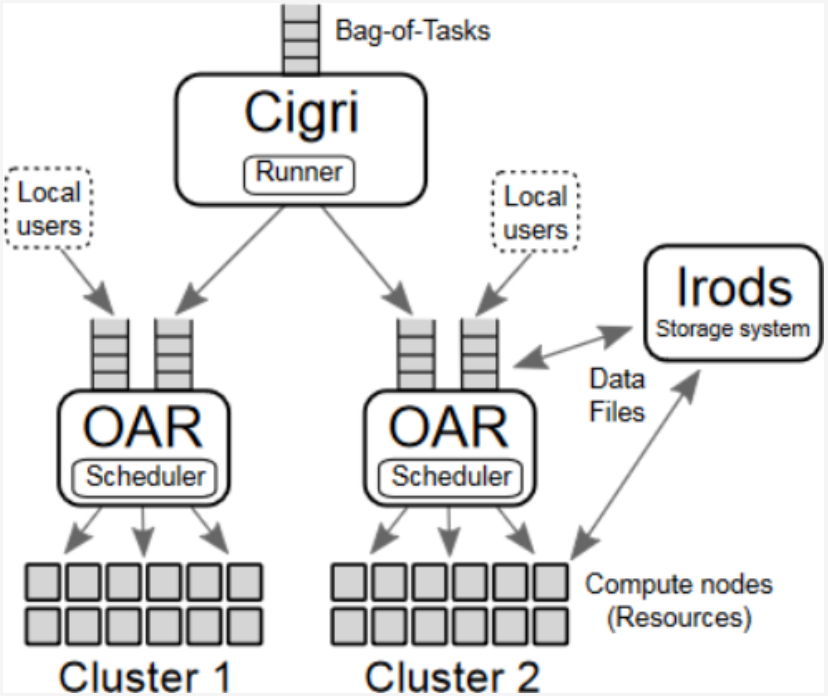


Existing usages/compositions: Cigri

(Guilloteau et al. 2022)

Objective Minimize wasted resources (idle nodes, killed jobs)

Constraint No (low?) impact on local users



best effort jobs

- lowest priority
- killable

- increase scheduling time
- induce load on file system

Existing usages/compositions: Cigri

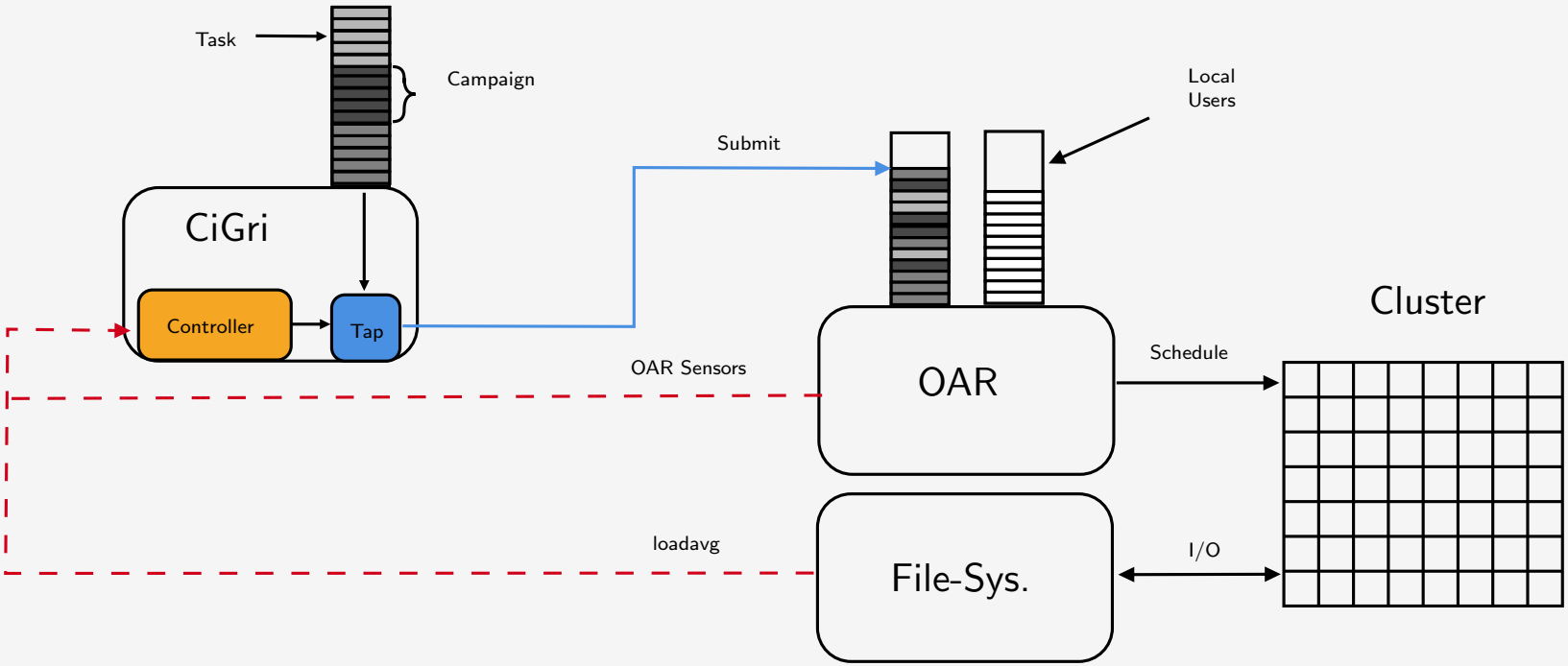
(Guilloteau et al. 2022)

How?

actuator number of best effort jobs "revealed"/sent

sensors queue size, load of the file system

$\Rightarrow$ control law to bind actuator & sensors



Existing usages/compositions: Cigri

(Guilloteau et al. 2022)

Designing the controller (control law) takes three main steps:

1. **system modeling (open loop)**:
   $\text{measure}(k+1) = \sum_{i=0}^{k} a_i \cdot \text{measure}(k-i) + \sum_{i=0}^{k} b_i \cdot \text{action}(k-i)$

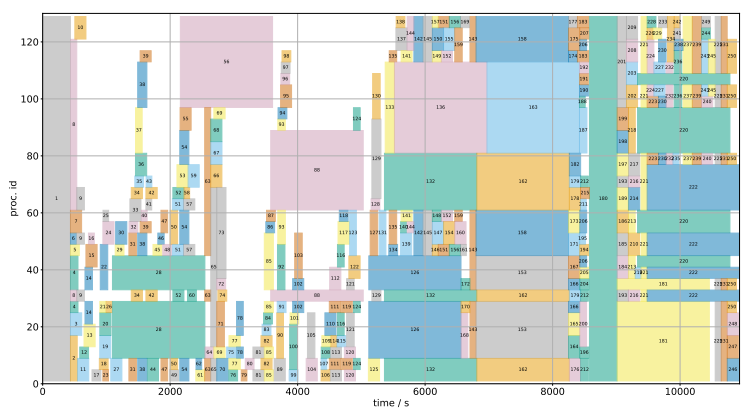
   control law type (closed loop) – e.g., PID:
   $\text{action}(k+1) = K_p\,\text{err}(k) + K_i \sum_{i=0}^{k} \text{err}(i) + K_d\,(\text{err}(k) - \text{err}(k-1))$

2. open-loop experiments (fixed action)

3. fitting of model & choice of controller parameters

Potential combinations

Scheduling × Control Theory



×

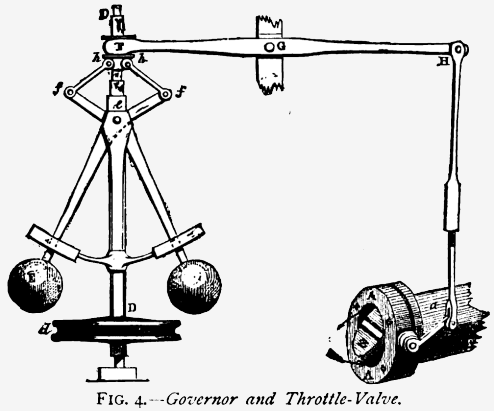
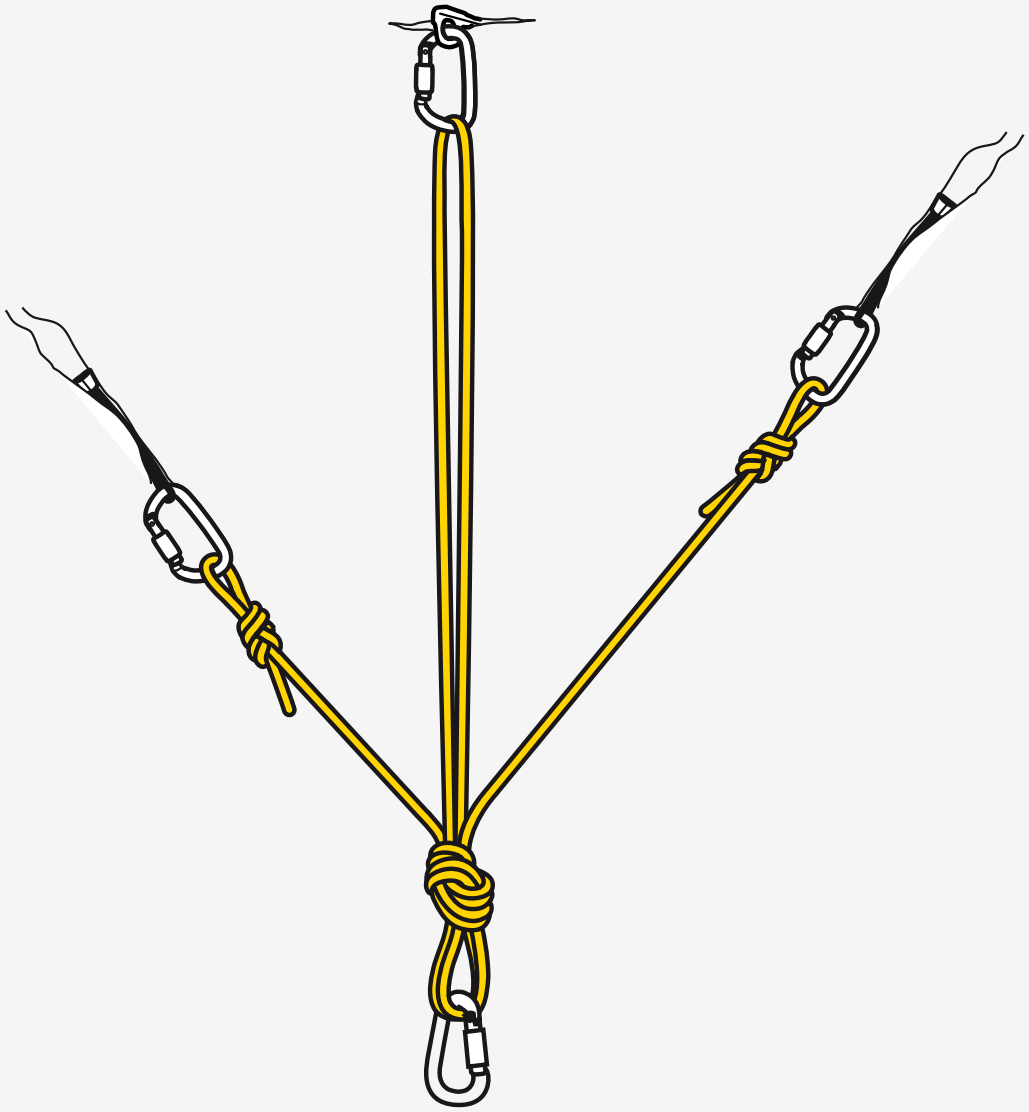


FIG. 4.—*Governor and Throttle-Valve.*

Open questions / Directions to explore

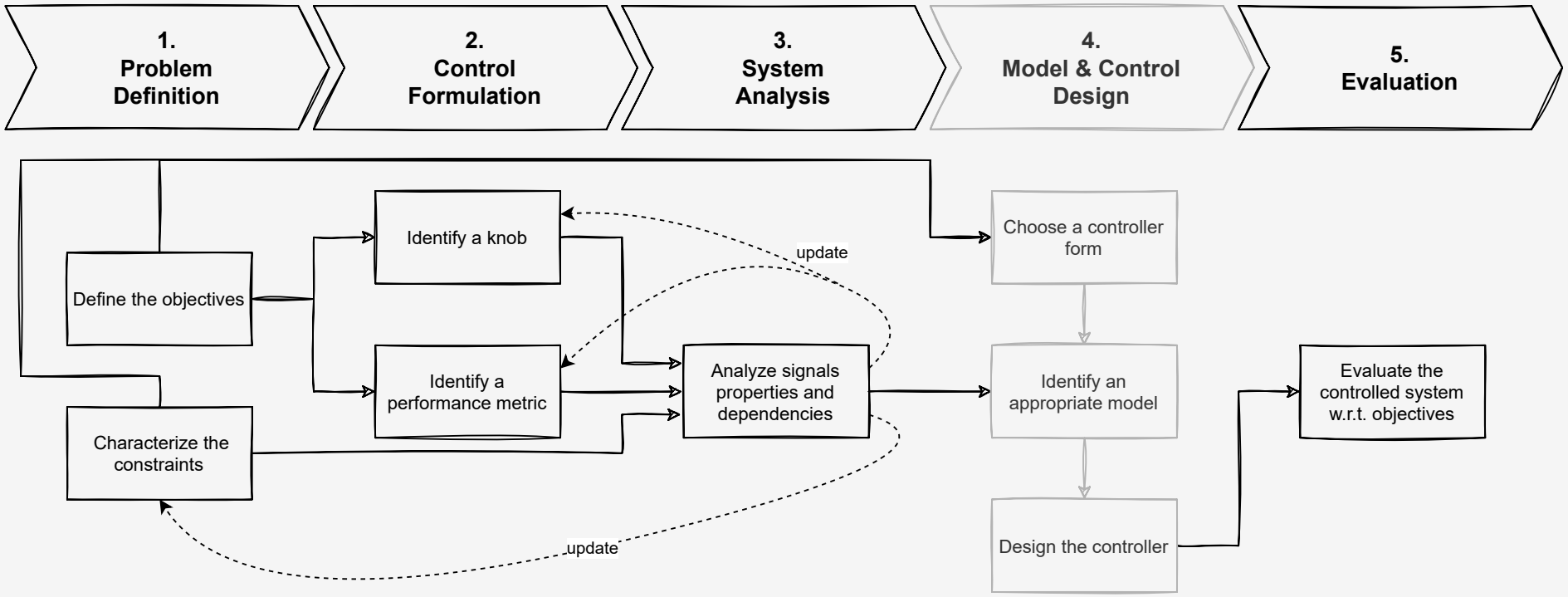
- what are the interesting properties to enforce?
- how to structure the loops?
- what to measure? what to control?
- what is the dynamic at play?

"All models are wrong but some are useful." (Box 1979)



Backup slides

Control Theory Methodology



References I

G. E. P. Box, "Robustness in the Strategy of Scientific Model Building," in *Robustness in Statistics*, edited by R. L. Launer and G. N. Wilkinson (Academic Press, 1979), pp. 201–236.

S. Cerf et al., "Sustaining Performance While Reducing Energy Consumption: A Control Theory Approach," in Euro-par, Vol. 12820, Lecture Notes in Computer Science (2021), pp. 334–349.

D. G. Feitelson, "Resampling with Feedback, A New Paradigm of Using Workload Data for Performance Evaluation," in Euro-par, Vol. 9833, Lecture Notes in Computer Science (Aug. 2016), pp. 3–21.

Y. Georgiou, "Contributions for Resource and Job Management in High Performance Computing," PhD thesis (LIG, Univ. Grenoble Alpes, France, Nov. 2010).

Q. Guilloteau, B. Robu, C. Join, M. Fliess, É. Rutten, and O. Richard, "Model-Free Control for Resource Harvesting in Computing Grids," in CCTA (Aug. 2022), pp. 384–390.

I. Hawila, S. Cerf, R. Bleuse, S. Perarnau, and É. Rutten, "Adaptive Power Control for Sober High-Performance Computing," in CCTA (Aug. 2022), pp. 403–410.

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

M. Madon, G. Da Costa, and J. Pierson, "Characterization of Different User Behaviors for Demand Response in Data Centers," in Euro-par, Vol. 13440, Lecture Notes in Computer Science (2022), pp. 53–68.

References II

A. Reuther, C. Byun, W. Arcand, D. Bestor, B. Bergeron, M. Hubbell, M. Jones, P. Michaleas, A. Prout, A. Rosa, and J. Kepner, "Scalable system scheduling for HPC and big data," Journal of Parallel and Distributed Computing **111**, 76–92 (2018).