Scheduling at the Edge

Clement Mommessin, Giorgio Lucarelli, Denis Trystram

Scheduling at the Edge

Clément Mommessin

Giorgio Lucarelli Denis Trystram

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

June 27th, 2019

Credits: `https://www.gironde.fr/actualites/residence-florestine-innovation-technologique-et-sociale`

From Heater to Computing Resources

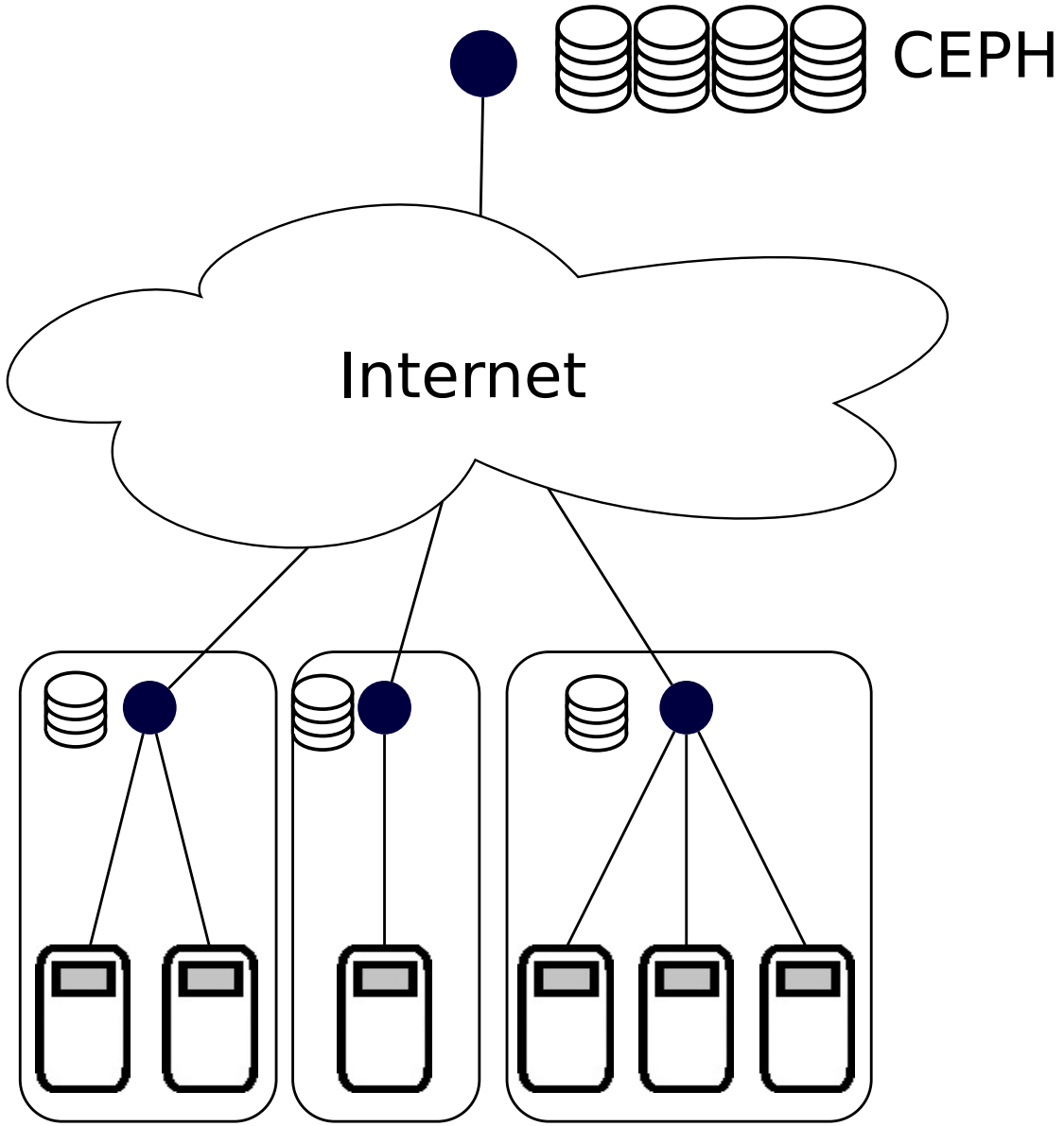
"A disruptive solution to turn IT waste heat into a viable heating solution for buildings."

The Qarnot platform:

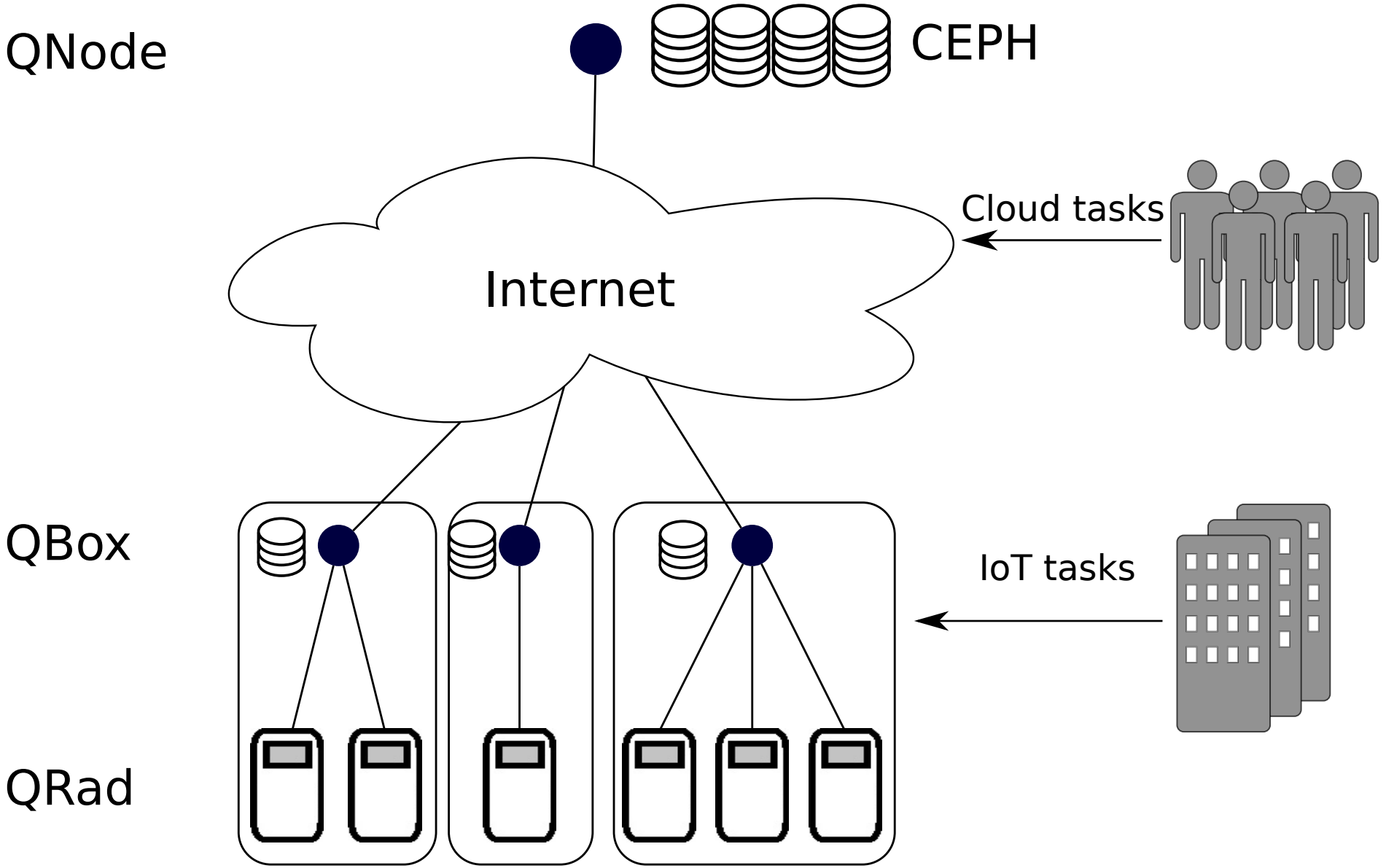
- ∼1,000 distributed QRads embedding ∼3,000 diskless computing units (QMobos)
- ∼20 local servers (QBoxes) with disks
- 1 global server (QNode) with a centralized storage server (CEPH)

Credits: `https://www.qarnot.com`

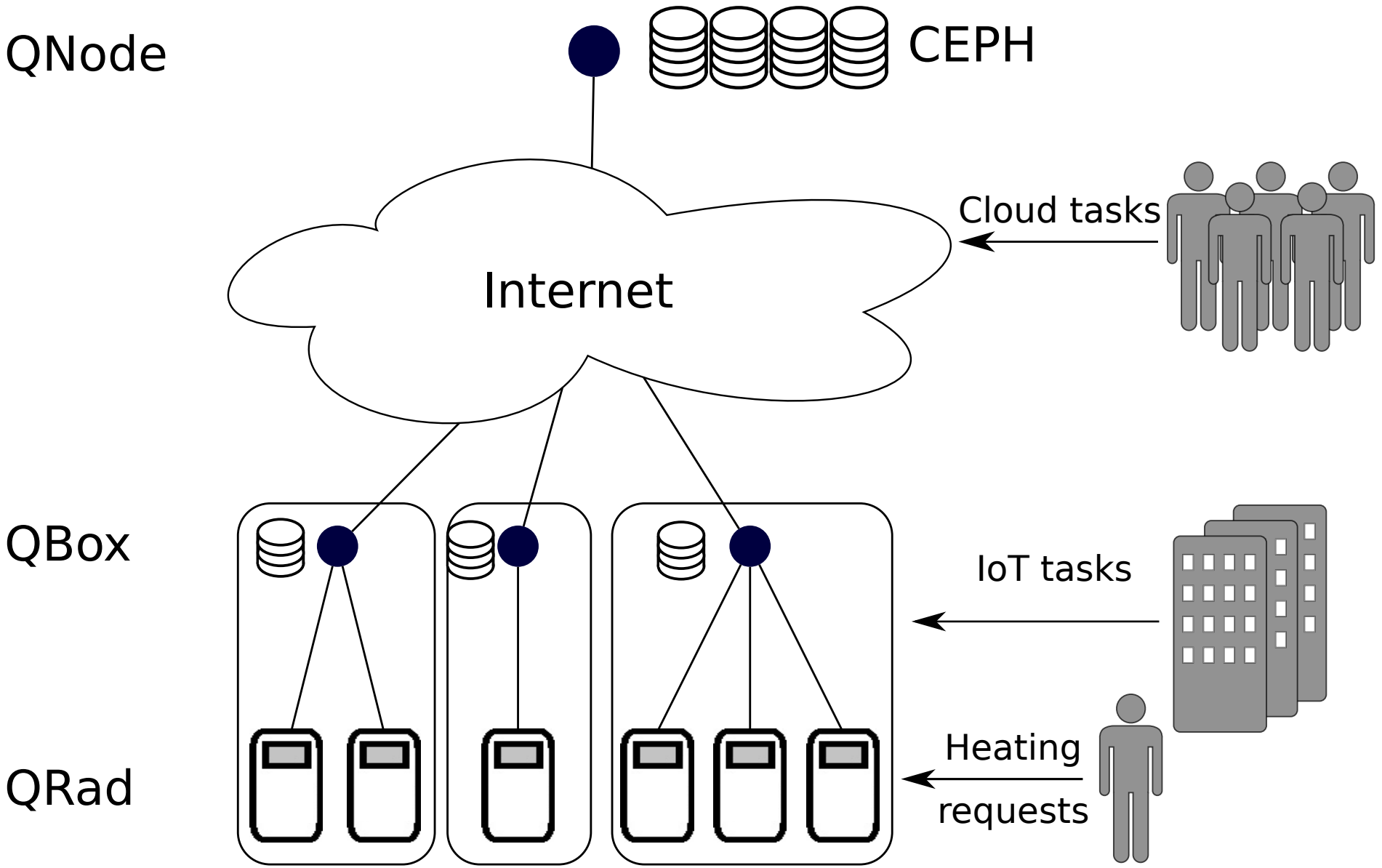
Qarnot Platform



Qarnot Platform



Qarnot Platform



Computing Requests

Cloud tasks:

- Submitted to the QNode
- Have data-set dependencies in the centralized storage (CEPH)
- Have different priorities (low or high)

IoT tasks:

- Submitted to a QBox
- Have data-set dependencies in the QBox disk
- Have different priorities (low, high or very high)
- Should be executed locally

Tasks (= groups of **sequential instances**) are submitted on-line.

Platform Dynamicity

Resources appear and disappear over time: **the inhabitants decide!**

- Available resources when heating is required (QRad is On)
- Unavailable resources when ambient air is too warm (QRad is Off)

$\rightarrow$ Also depends on the task priority

Network uncertainties:

- Link failures
- Congestion/contention

Two-Level Scheduling Problem

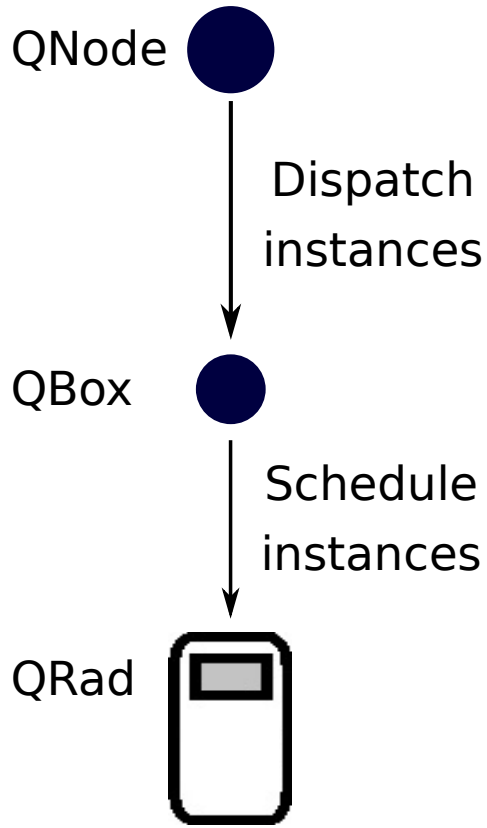
The only computing power is in the QRad

Make **global decisions** at QNode-level:

- Decide where to dispatch (groups of) **instances**
- Ensure global load-balancing

Make **local decisions** at QBox-level:

- Schedule instances on QRads
- Regulate room temperature (via DVFS)
- Ensure heating needs are satisfied



Two-Level Scheduling Problem

The only computing power is in the QRad

Make **global decisions** at QNode-level:

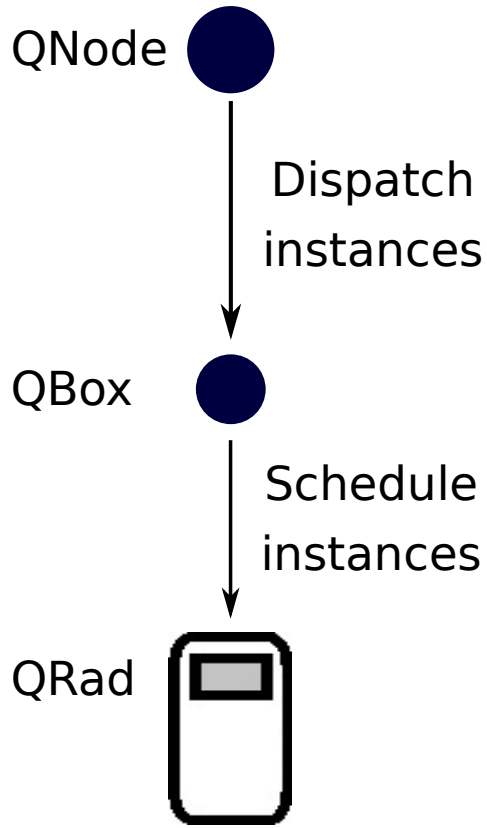
- Decide where to dispatch (groups of) **instances**
- Ensure global load-balancing

Make **local decisions** at QBox-level:

- Schedule instances on QRads
- Regulate room temperature (via DVFS)
- Ensure heating needs are satisfied

Need to reach 100% of platform usage.

$\Rightarrow$ **An idle QRad is a lack of heating**



Multiple Objectives

Different objectives for different users:

- Cloud users: Minimize waiting/completion time of tasks
- IoT tasks: "Nullify" responsiveness
- Inhabitants: Minimize distance to target temperature
- Qarnot: Maximize tasks throughput, minimize lack of heating

Qarnot Solution: Go On-line

Periodic reports ($\sim$30 s) from QBox to QNode with:

- Number of resources available for each task priority
- Free space on disk

QNode-scheduling:

- Sort QBoxes by least available resources first (no temperature knowledge)
- Sort tasks by highest priority first
- For each task, dispatch as much instances as possible

QBox-scheduling:

- Retrieve data-set dependencies
- Schedule high priority instances on coolest QRads
- Schedule low priority instances on warmest QRads

Temperature Regulation

Frequency and temperature regulator in each QRad:

- In general: DVFS on QMobos to adapt power consumption
- When too hot: instances are killed and re-submitted to the QNode
- When too cold (lack of heating): *"background"* compute-intensive instances are generated (best-effort blockchain mining ☹)

Our objectives

Main goal: design, implement and test different placement and scheduling policies at both QNode- and QBox-level.

Main problem: testing on a production platform is not conceivable and takes time.

Our objectives

Main goal: design, implement and test different placement and scheduling policies at both QNode- and QBox-level.

Main problem: testing on a production platform is not conceivable and takes time.

$\Rightarrow$ **Simulation is what you need!**

Simulation Tools

SimGrid[1]: Large-scale distributed system simulator with execution and communication models.
→ Used to simulate platform and tasks execution.

Batsim[2]: Infrascructure simulator for jobs and I/O scheduling.
→ Used to drive the simulation, submit tasks and communicate with the decision process.

Pybatsim[3]: Batsim's Python API exposing methods to easily communicate with the Batsim process.
→ Used to implement the QNode and QBox schedulers.

[1] `https://github.com/simgrid/simgrid`
[2] `https://gitlab.inria.fr/batsim/batsim/tree/temperature`
[3] `https://gitlab.inria.fr/batsim/pybatsim/tree/temperature`

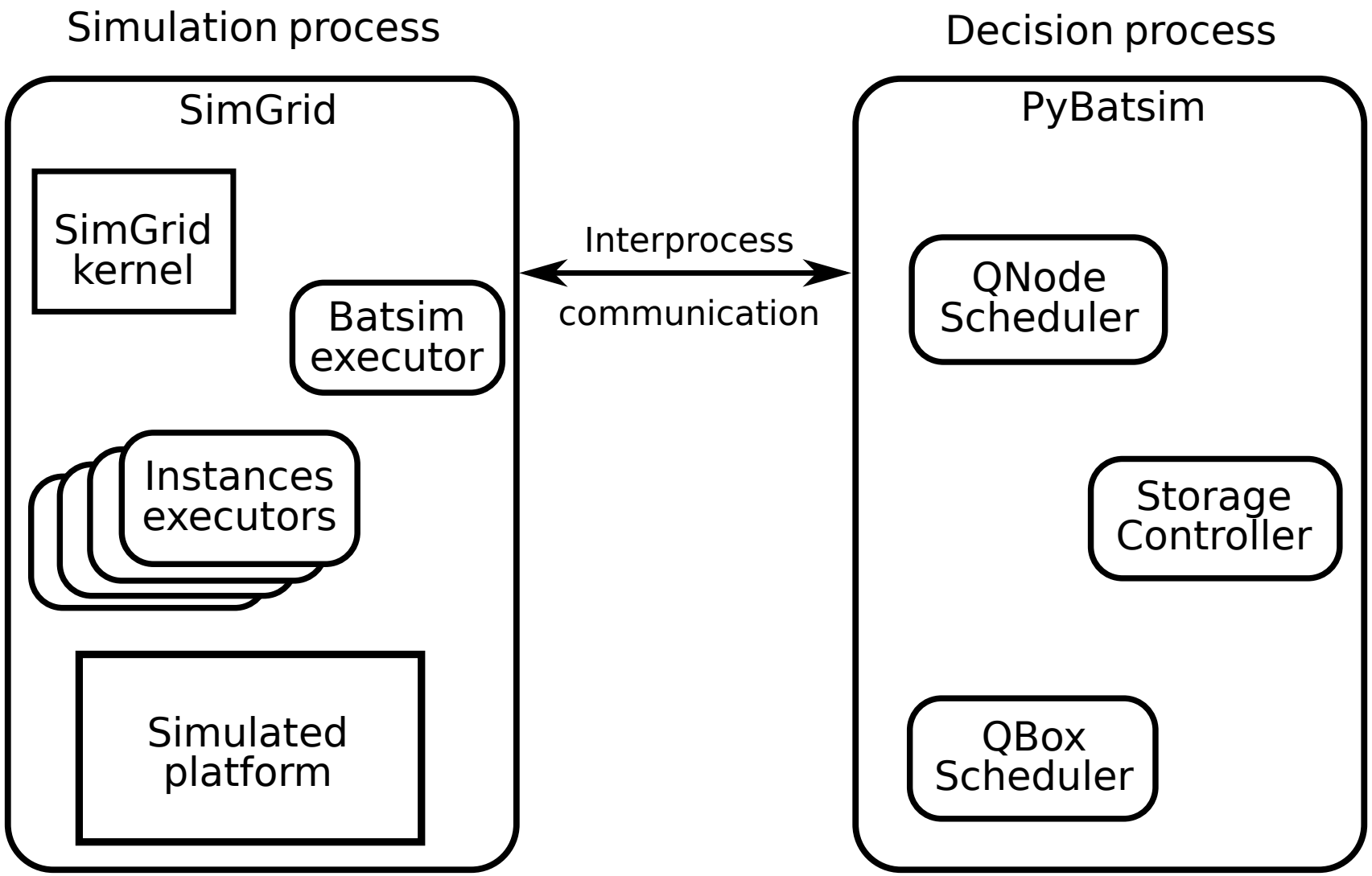
Developed Extensions

Temperature model: to compute the temperature of the QRad and ambient air, based on thermodynamics formulae.

External events injector: to replay a machine failure or a temperature change.

Storage controller: to manage storage entities and data movements.

Simulation Overview



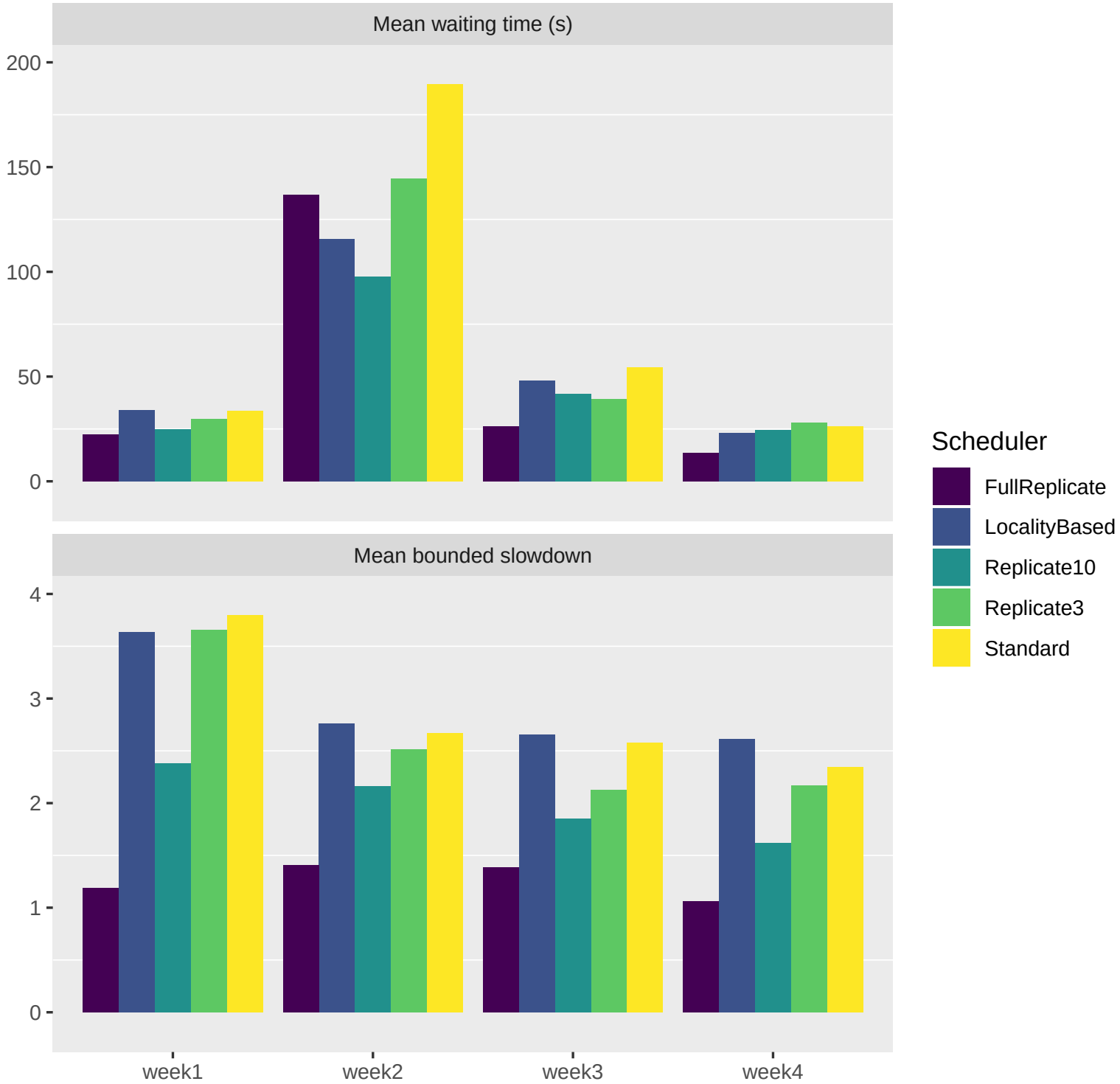
Investigating Tasks and Data Placements

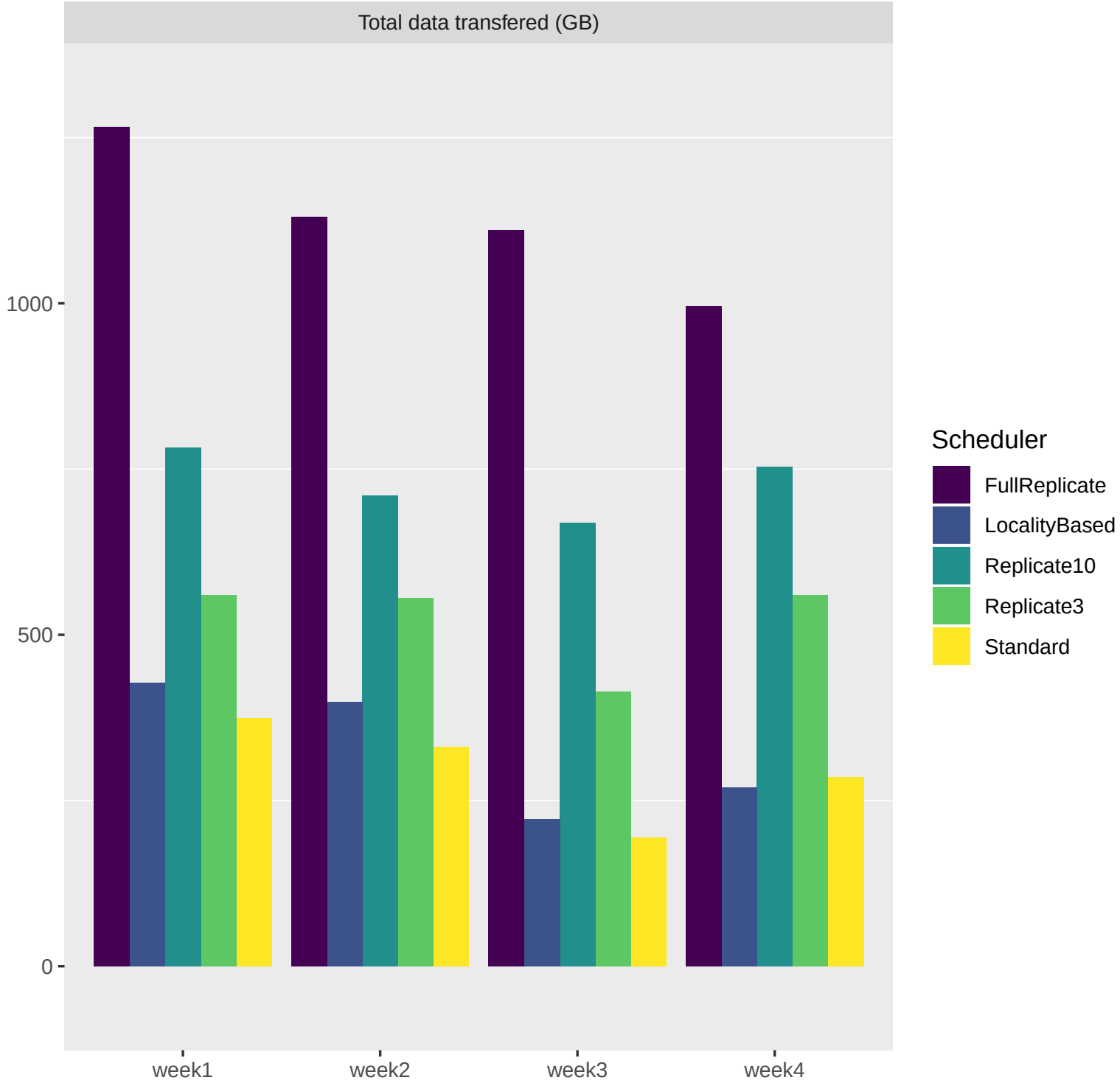
Variants of the QNode dispatcher:

- *Standard*
- *LocalityBased*
- *Replicate3LeastLoadedDisk*
- *Replicate10LeastLoadedDisk*
- *FullReplicate* (instantaneous transfers)

Standard Qarnot scheduler at QBox-level.

1-week simulation inputs from real logs of Qarnot.


Mean waiting time (s)
200
150
100
50
0
Mean bounded slowdown
4
3
2
1
0
week1
week2
week3
week4
Scheduler
FullReplicate
LocalityBased
Replicate10
Replicate3
Standard


Total data transfered (GB)
1000
500
0
week1
week2
week3
week4
Scheduler
FullReplicate
LocalityBased
Replicate10
Replicate3
Standard

QBox-level Scheduling

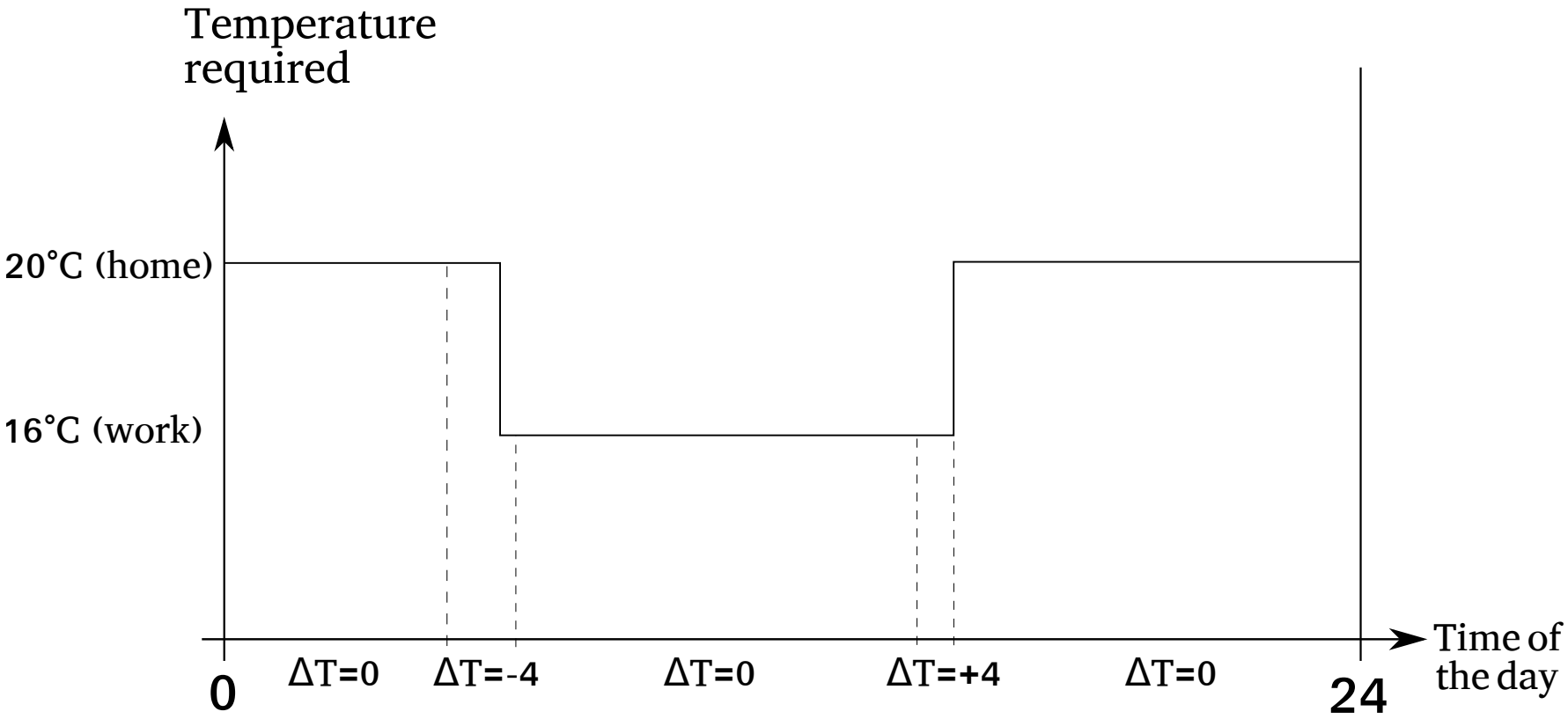
List of n instances from Cloud or IoT, with for each instance j:

- An estimation of work $Work_j$
- A priority value w_j
- A release date r_j (max arrival time of the dependent data-sets)

List of m QRads, with for each:

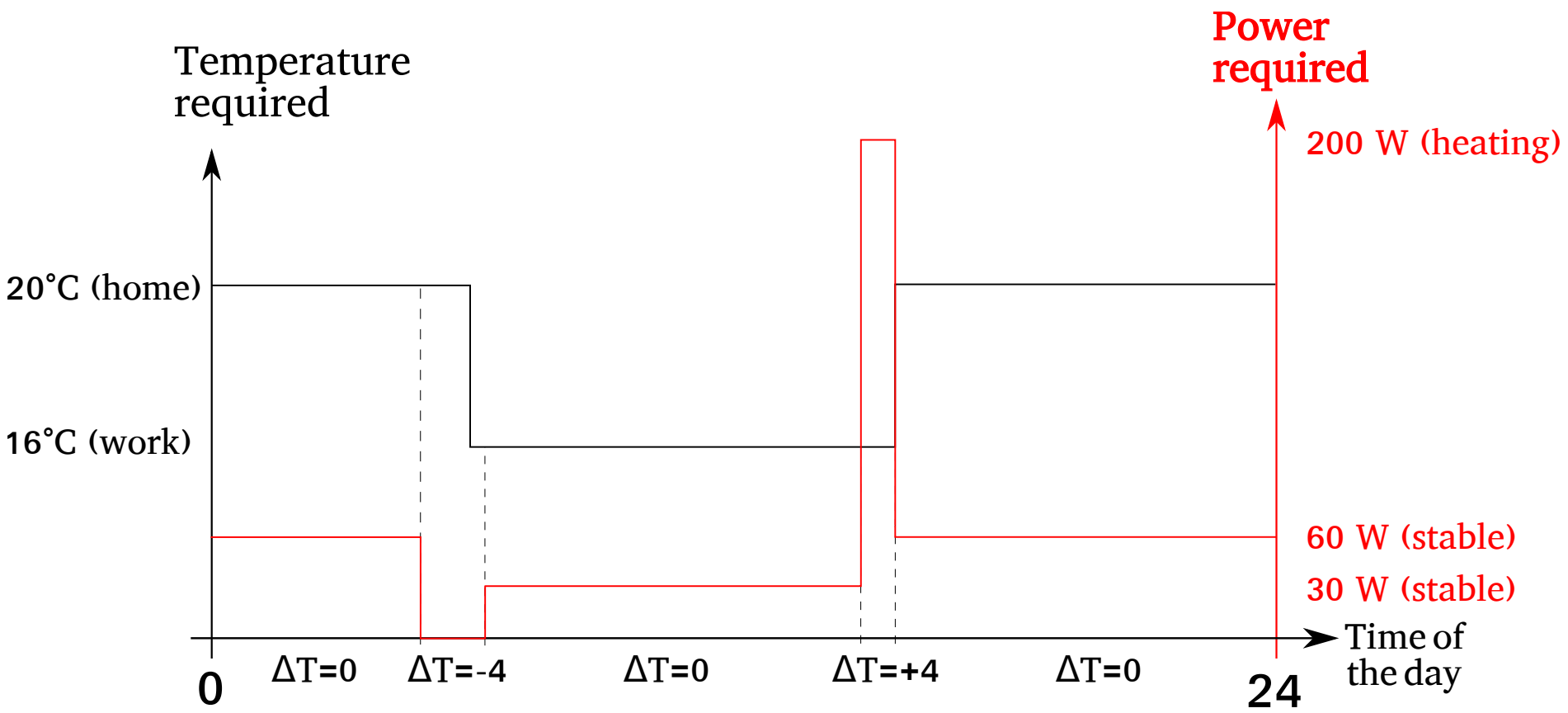
- A number of QMobos
- A list of possible speeds of a QMobo
- The corresponding power consumption of each speed
- A diagram of target temperature over a day/a week
- The corresponding power diagram over a day/a week

QBox-level Scheduling



Target temperature diagram

QBox-level Scheduling



Power diagram

QBox-level Scheduling

THE questions to answer:

- Where to execute an instance?
- When to execute an instance?
- **At which speed?**

$\Rightarrow$ **Similar to power capping problems**

Objectives:

- Minimize $\sum_j w_j C_j$ (where C_j is the completion time)
- Minimize power consumption distance to the power diagrams

Conclusions and Prospects

Preliminary results:

- Scheduling at the edge depends highly on heterogeneity and dynamicity
- Simulating edge platforms is possible thanks to Batsim/SimGrid
- Data placement is not trivial

$\Rightarrow$ Short paper submitted in IEEE Mascots 2019.

Current work:

- Try other placement policies at QNode-level, communicate more with the QBoxes
- Study the QBox theoretical model
- **Validate the temperature model**

Many Thanks

Looking for temperature and power consumption of machines/cores in a cluster

This work is supported by the ANR Greco project.

Credits: `https://www.la-cnem.org/we-need-you/`

Scheduling at the Edge

Clément Mommessin

Giorgio Lucarelli Denis Trystram

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

June 27th, 2019