The Problem of Scheduling on Smart Heaters

Mommessin Clément

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

June 7th, 2019

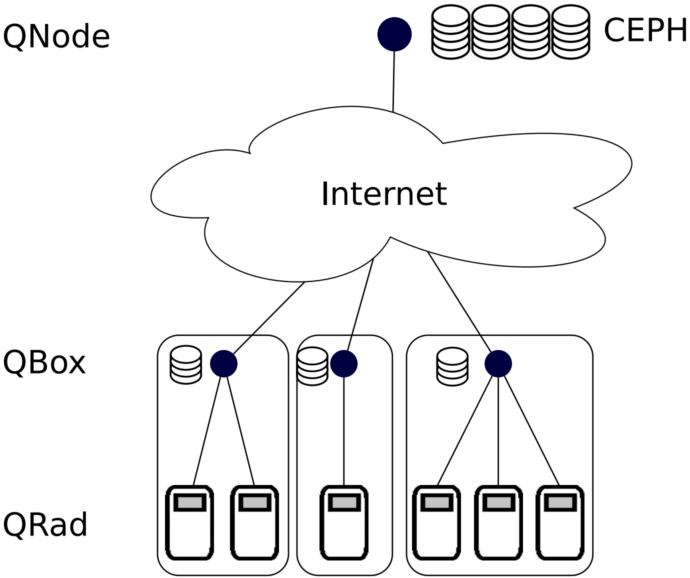
Qarnot Model

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

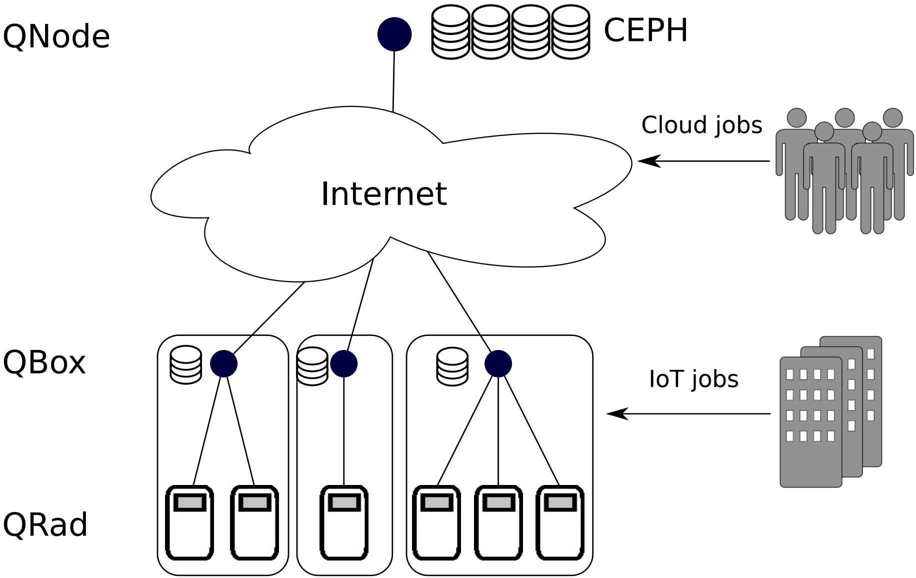
The Qarnot platform:

- ~1,000 distributed Smart Heaters (QRads), embedding ~3,000 diskless computing units (QMobos), in rooms and offices of buildings
- 20 local servers (QBoxes) managing the resources and hosting a storage disk
- 1 global server (QNode) to control and manage the platform, with gateways for the users and a centralized storage server (CEPH)

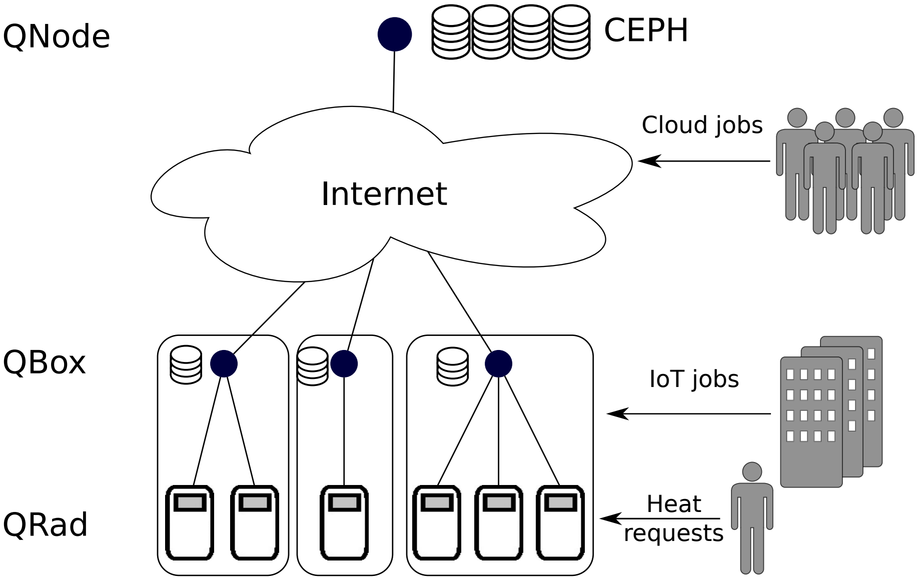
Qarnot Platform



Qarnot Platform



Qarnot Platform



Platform Dynamicity

Temperature constraints:

- Available resources when heating is required
- Unavailable resources when ambient air is too warm

Network uncertainties:

- Link failures
- Congestion/contention

Two-Level Scheduling Problem

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

Make global decisions at the QNode-level:

- Decide where to replicate data-sets
- Decide where to dispatch (groups of) instances
- Ensure global load-balancing w.r.t. heating needs

Make local decisions at the QBox-level:

- Schedule received instances on QRads
- Regulate room temperature and CPU frequencies
- Ensure heating needs are satisfied

Qarnot Solution: Go On-line

Perioric reports (∼30 s) from QBox to QNode with:

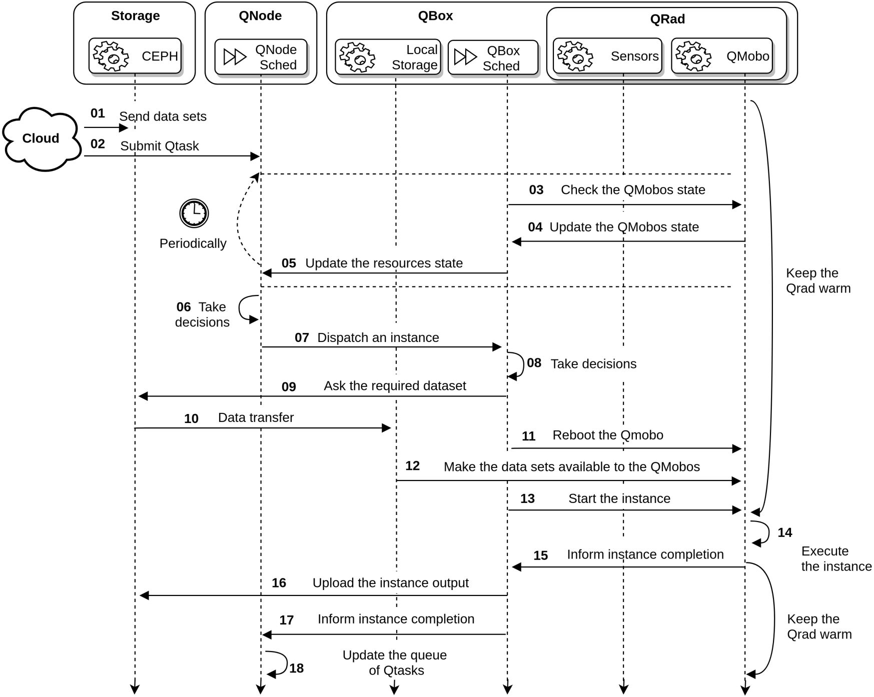
- Number of resources available by priority
- Free space on disk

QNode dispatcher:

- Sort QBoxes by increasing number of available resources
- Sort tasks by decreasing priority
- Send as much instances as possible of the first task on QBoxes
- Repeat with next task until no more available resource

QBox scheduler:

- Download input data-sets upon receiving new instances
- Schedule instances on available resources
- Start instances when data-sets are on disk and regulate frequencies
- Spawn compute-intensive jobs to complement heating


Storage
CEPH
QNode
QNode Sched
QBox
Local Storage
QBox Sched
QRad
Sensors
QMobo
Cloud
01 Send data sets
02 Submit Qtask
Periodically
03 Check the QMobos state
04 Update the QMobos state
05 Update the resources state
06 Take decisions
07 Dispatch an instance
08 Take decisions
09 Ask the required dataset
10 Data transfer
11 Reboot the Qmobo
12 Make the data sets available to the QMobos
13 Start the instance
14
Execute the instance
15 Inform instance completion
16 Upload the instance output
17 Inform instance completion
18
Update the queue of Qtasks
Keep the Qrad warm
Keep the Qrad warm

Multiple Objectives

Different objectives for different users:

- Cloud users: Minimize waiting/completion time
- Inhabitants: Minimize distance to target temperature
- Qarnot: Maximize profit, minimize energy consumption

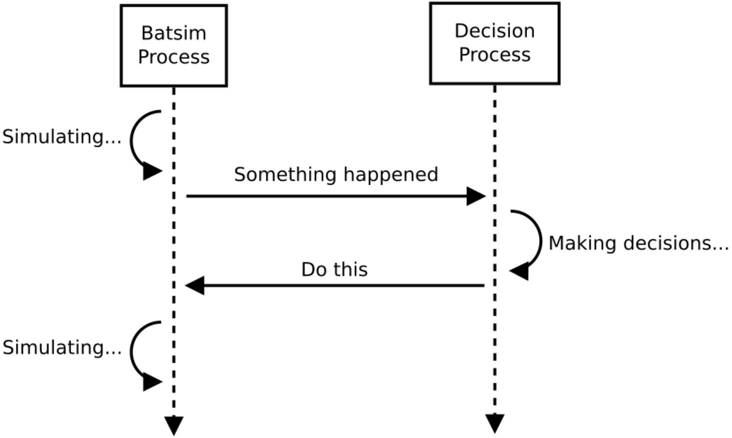
Simulation Tools

SimGrid: generic platform simulator with realistic network and communication model. Additional plug-in to compute energy consumption of resources and temperature of ambient air.

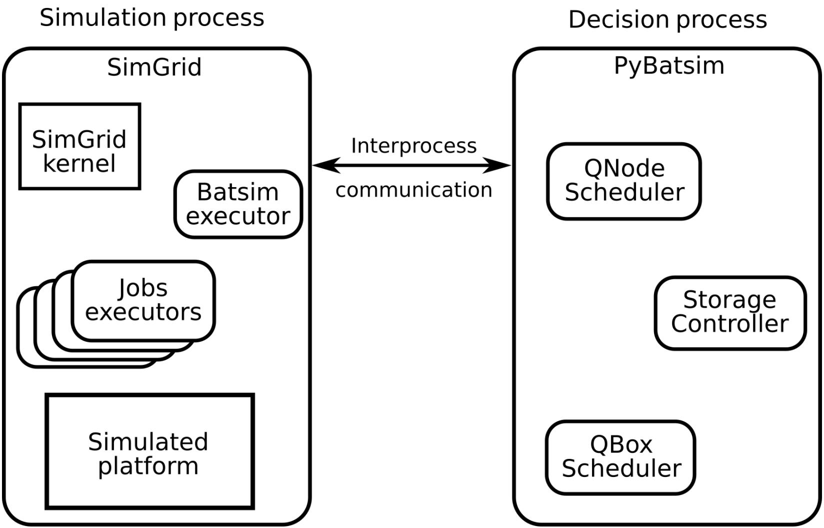
Batsim: infrastructure simulator for jobs and I/O scheduling, on top of SimGrid. Simulates jobs submissions and occurrences of external events.

PyBatsim: Python API to write a scheduler and communicate with Batsim. Receives simulation events and answers with scheduling decisions.

Batsim Communication Protocol



Simulation Overview



Logs Extraction

From Qarnot database to SimGrid/Batsim input files:

- Platform: List of QBoxes with network links and disk capacities, QRads with number of resources
- Workload: List of instances with submition time, number of flops, data-set dependencies
- External events: Target temperature changes, resource or network failures
- Datasets: List of data-sets uploaded by users with name and size

Extracted data for different dates and time intervals: 1 day, 3 days and 1 week.

Experiments

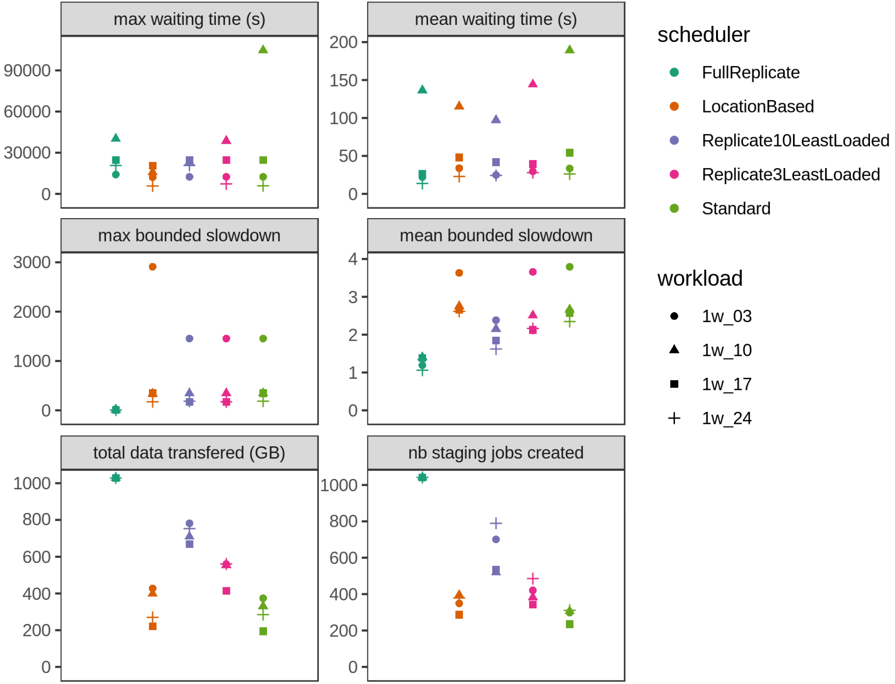
A few simulations to:

- Compare Qarnot schedulers behaviour VS reality
- Design, test, compare different scheduling policies

Considered QNode dispatching policies:

- Standard: dispatch on QBoxes with least available resources
- FullReplicate: Standard with data-sets replicated on every disk

- Locality-based: dispatch first on QBoxes having all required data-sets
- Replicate3: Locality-based with replication on 3 least loaded disks
- Replicate10: Locality-based with replication on 10 least loaded disks


max waiting time (s)
mean waiting time (s)
max bounded slowdown
mean bounded slowdown
total data transfered (GB)
nb staging jobs created
90000
60000
30000
0
200
150
100
50
3000
2000
1000
4
3
2
1
800
600
400
scheduler
FullReplicate
LocationBased
Replicate10LeastLoaded
Replicate3LeastLoaded
Standard
workload
1w_03
1w_10
1w_17
1w_24

To Conclude

Takeaway:

- Generic simulation framework with energy/temperature support
- Easy testing and comparison of scheduling policies

- Put more efforts on the data placements
- Uncertainties make scheduling not easy!

In the near future:

- Work on theoretical model at QBox-level, with a queue of tasks
- Explore and test other schedulers at QBox-level

- Refine QNode scheduler with more info on heating needs

Thank you!