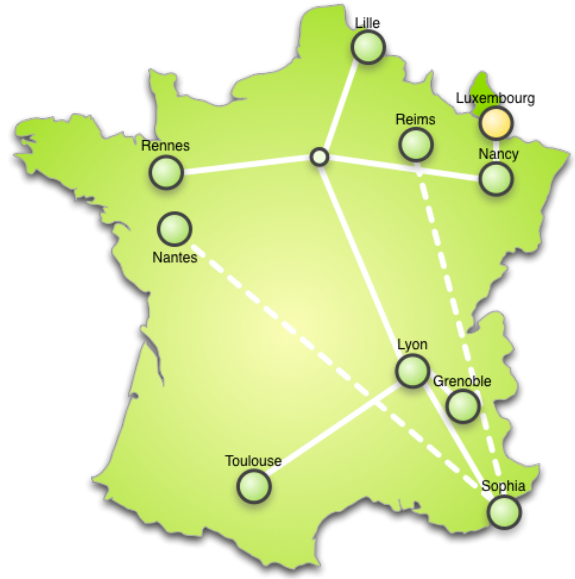
Resources management on the Grid'5000 testbed

Pierre Neyron and Lucas Nussbaum

Context: the Grid'5000 testbed



- **A large-scale distributed testbed for distributed computing**
  - 8 sites, 32 clusters, 894 nodes, 8490 cores
  - Dedicated 10-Gbps backbone network
  - 550 users and 100 publications per year
- A meta-grid, meta-cloud, meta-cluster, meta-data-center:
  - Used by CS researchers in HPC, Clouds, Big Data, Networking
  - To experiment in a fully controllable and observable environment

Roots of resources management on Grid'5000

- Grid'5000 original scope, back in 2003:
  - **HPC (high performance computing) testbed**
    - Flexible, reconfigurable HPC infrastructure
    - To experiment on all layers (inc. OS-level $\rightsquigarrow$ bare-metal)
- Logical software foundation:
  HPC resources and jobs management system (OAR)
  - Usually used in HPC clusters to:
    - Manage queues of jobs
    - Schedule jobs on cluster nodes/cores

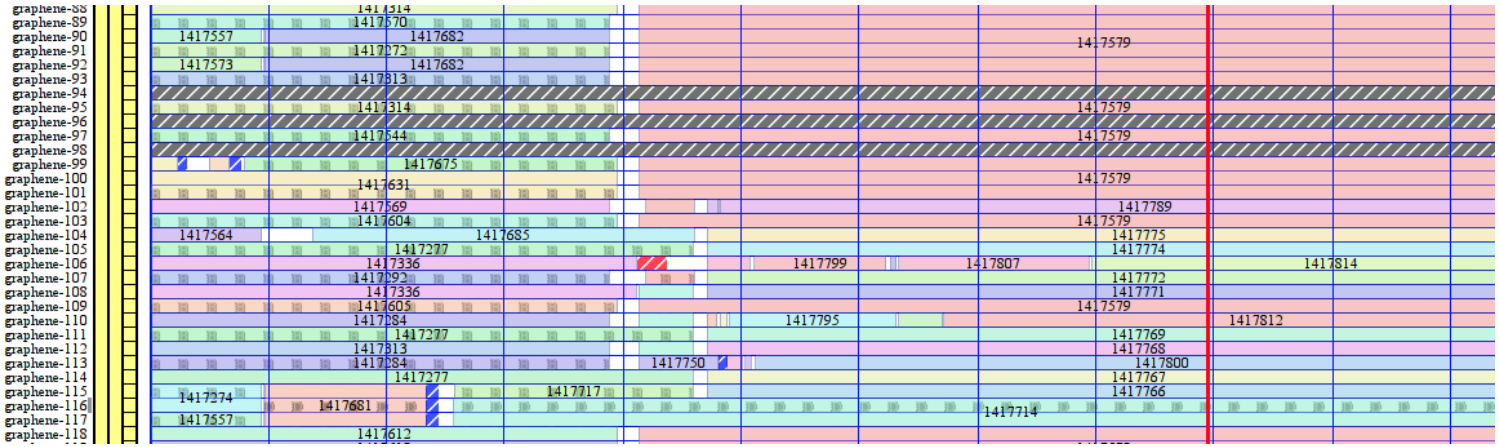
Resources description & selection

- Each resource has attributes
  - hardware properties, location, connections, etc.
- Powerful selection mechanism based on attributes
  - Using the attributes for hierarchy
    `switch=1/nodes=10` : 10 nodes on the same switch
  - Each attribute is an SQL column $\rightsquigarrow$ SQL to filter resources
    `wattmeter='YES' and gpu='YES' and eth_count >= 4`

Extended to other kinds of resources

- Not just cores and nodes
- Such as:
  - VLANs
  - network subnets (for virtual machines addressing)
  - storage volumes on a storage server
  - local disks on nodes
- The resources manager (OAR) manages the lifecycle for all resources

Resources allocation



- Queue-based: First-In-First-Out
  - With backfill (fill holes due to scheduling)
  - Optionally with priorities based on past usage (karma)
- Advance reservations
  - Typically for large experiments durings nights/week-ends

$\Rightarrow$ Allows for a **complex usage policy**

- Strict sharing during the day, almost no limits during nights/week-ends
- Produces a lot of turnover (efficient use of resources)
- Encourages users to automate their experiments (at least the setup part)

Future improvements

- Quotas and karma per group/project, not just per user
- Easier co-scheduling of different resources (e.g. nodes and disks)
- Better scheduling of large experiments over nights/week-ends
  - Priority-based queue of large experiments, with constraints?
    I'd like those resources for a full night ASAP (except on Thursday)