Modeling the Allocation of Heterogeneous Storage Resources on HPC Systems

PhD Student:
Julien Monniot – *julien.monniot@inria.fr*
INRIA Rennes / Université Rennes 1

Advisors:
François Tessier – INRIA Rennes
Gabriel Antoniu – INRIA Rennes

Inria

UNIVERSITÉ DE RENNES 1

CONTEXT AND MOTIVATIONS

Up to 3EB/year by 2027 — SKA Telescope
1 PB/day by 2026 — ECMWF Forecast & Digital Twin Earth
2.2 PB/day by 2035 — ITER Reactor

Burst-buffers / staging area (SSD, NVMEoF, HDD, ...)
Node-local / Platform integrated (SSD, NVRAM, ...)
Hot / Cold
PFS / Archives (HDD, Tapes)

Examples:
***Perlmutter** 35PB **all flash storage** at NERSC (2021)*
***Aurora** DAOS Storage with Intel Optane **persistent memory** at ALCF (2022)*
***Summit** compute nodes with 1.6TB of **NVRAM** at OLCF (2018)*

- **Data deluge** from new large-scale scientific workflows
- ↗ PFlops ↘ TBps

- **Deeper storage hierarchy**
- New underlying **technologies**
- **Hybrid** platforms / workflows

- ↗ **Complexity** & **Underutilization** of resources

Figure 1. Top500: I/O bandwidth to computing power ratio for the top 3 HPC platforms, last 10 years.

From HPC... + ...to Cloud

Optimal Resource **Scheduling**
②

MAIN AXES OF WORK · · · WHAT DO WE AIM FOR?

①

- Design of **scheduling algorithms** for storage allocations
- **Representation** of heterogeneous storage infrastructures
- Analysis of **storage related metrics**

Enable dynamic allocation of heterogeneous storage resources (real or simulated)
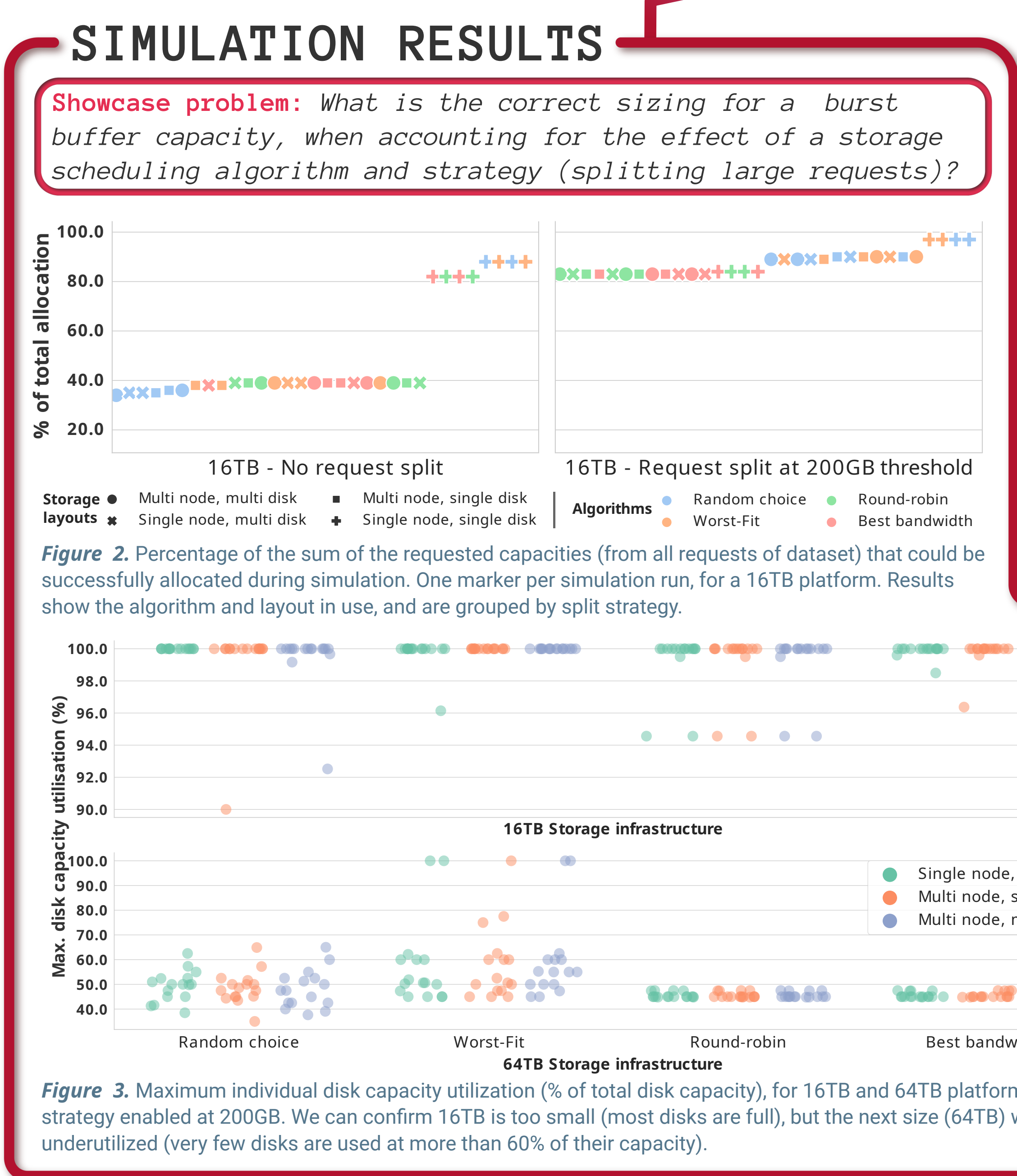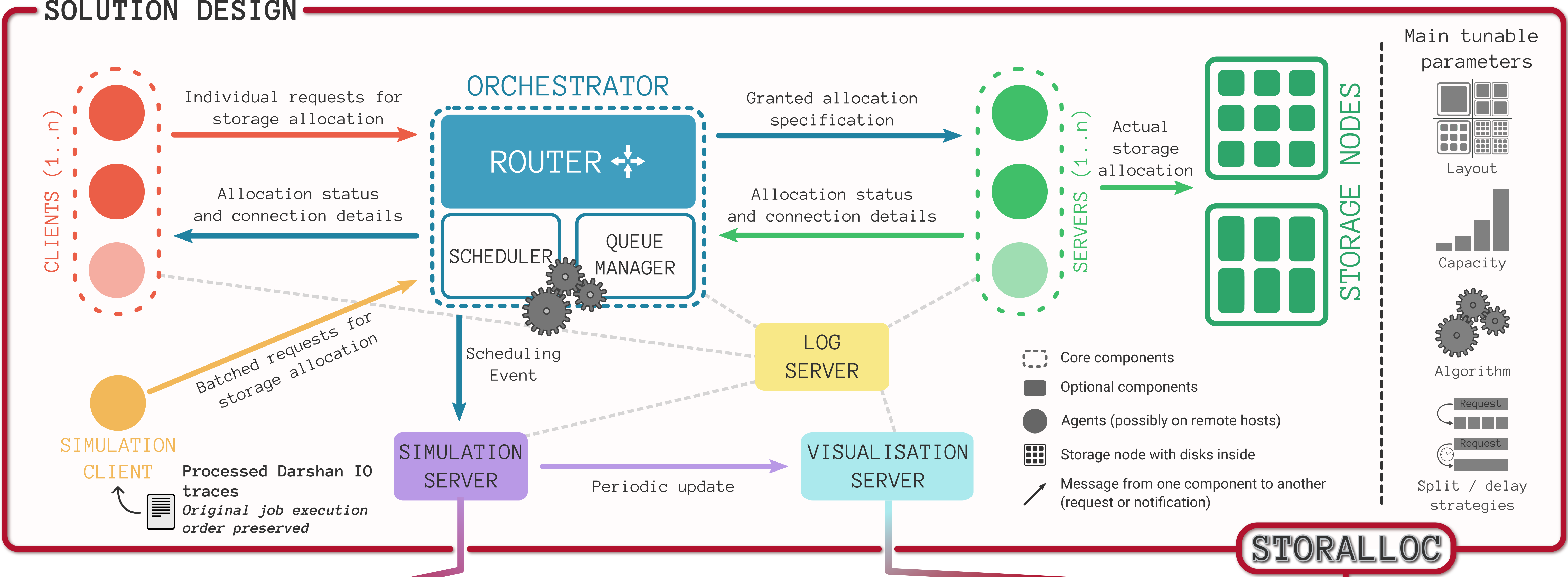
Dealing with resource **heterogeneity**
③

Fair and **efficient** use of resources

SOLUTION DESIGN

ORCHESTRATOR

CLIENTS (1..n)
Individual requests for storage allocation →
ROUTER ✛
← Allocation status and connection details

SCHEDULER | QUEUE MANAGER

Granted allocation specification →
SERVERS (1..n)
← Allocation status and connection details

Actual storage allocation →
STORAGE NODES

SIMULATION CLIENT
Batched requests for storage allocation
Processed Darshan IO traces
Original job execution order preserved

Scheduling Event →
SIMULATION SERVER → Periodic update → VISUALISATION SERVER

LOG SERVER

- Core components
- Optional components
- Agents (possibly on remote hosts)
- Storage node with disks inside
- Message from one component to another (request or notification)

Main tunable parameters
Layout
Capacity
Algorithm
Split / delay strategies

STORALLOC

SIMULATION RESULTS

Showcase problem: *What is the correct sizing for a burst buffer capacity, when accounting for the effect of a storage scheduling algorithm and strategy (splitting large requests)?*

16TB - No request split | 16TB - Request split at 200GB threshold

Storage layouts: Multi node, multi disk · Single node, multi disk · Multi node, single disk · Single node, single disk
Algorithms: Random choice · Worst-Fit · Round-robin · Best bandwidth

Figure 2. Percentage of the sum of the requested capacities (from all requests of dataset) that could be successfully allocated during simulation. One marker per simulation run, for a 16TB platform. Results show the algorithm and layout in use, and are grouped by split strategy.

16TB Storage infrastructure
Single node, multi disk · Multi node, single disk · Multi node, multi disk

64TB Storage infrastructure
Random choice · Worst-Fit · Round-robin · Best bandwidth

Figure 3. Maximum individual disk capacity utilization (% of total disk capacity), for 16TB and 64TB platforms, split strategy enabled at 200GB. We can confirm 16TB is too small (most disks are full), but the next size (64TB) would be underutilized (very few disks are used at more than 60% of their capacity).

IN SITU VISUALISATION

Real-time plotting of main metrics

Total Allocated Volume at time *t*

Number of concurrent allocations at time *t*
+ Allocation details per disk / nodes

Random choice · Worst-Fit · Round-Robin · Best bandwidth
Accepted · Allocated · Failed · Refused

Figure 4. Number of split requests accepted, allocated, failed (during simulation) and refused (before simulation), grouped by algorithm. Random choice and Worst-fit are more permissive and rarely refuse requests, which eventually leads to more allocations. On the other hand, they result in more failed requests (error from the algorithm caught during allocation), which could prove costly for the user.

Conclusion: *The required storage capacity is not far over 16TB. Splitting the requests and using an algorithm well suited for this strategy is however necessary (random choice or worst-fit is this case).*

Storage-aware job scheduler simulator

- Extensible with new algorithms
- Independent components design
- Common messaging interface
- Abstraction of heterogeneous storage hardware

IMPLEMENTATION

- Python3
- SimPy (DES simulation)
- ZMQ (Messaging)
- bokeh (Real-time plotting)
- seaborn (Plotting)
- Darshan (IO traces)

GitHub
hephtaicie/storalloc