Coordinating Server Access at the I/O Forwarding Layer

TWINS: Server Access Coordination in the I/O Forwarding Layer, PDP 2017

Jean Luca Bez, Francieli Z. Boito, Lucas M. Schnorr,
Jean-François Mehaut, Philippe O. A. Navaux

{**jean.bez**, fzboito, schnorr, navaux}@inf.ufrgs.br
jean-francois.mehaut@imag.fr

UFRGS
UNIVERSIDADE FEDERAL
DO RIO GRANDE DO SUL

.inf
INSTITUTO
DE INFORMÁTICA
UFRGS

HPC

Summary

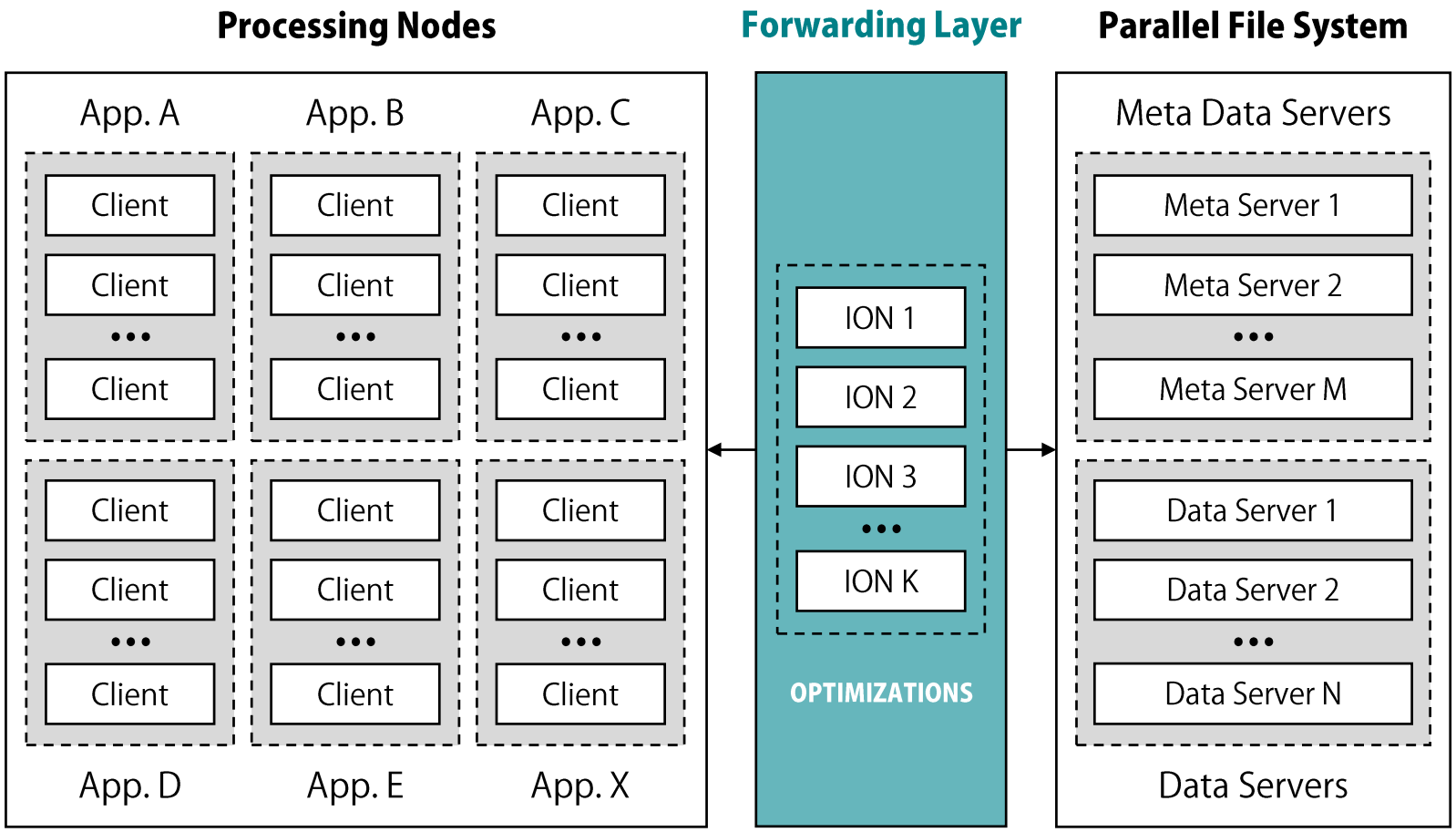
I. The Forwarding Layer

II. Experimental Methodology

III. **TWINS**: Coordinating Server Access

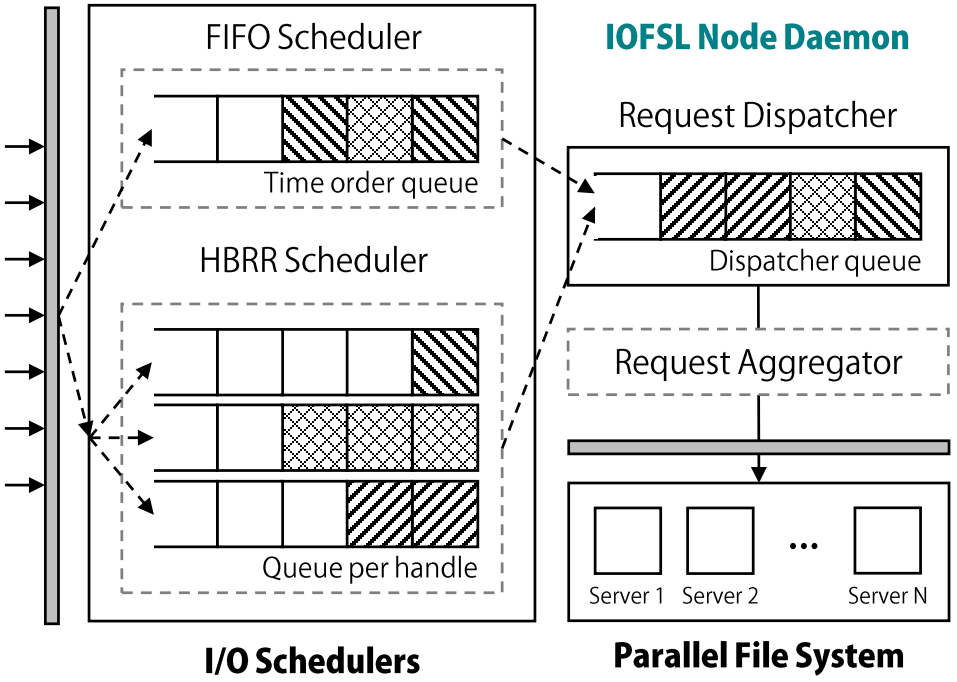
IV. Conclusions

The Forwarding Layer



The Forwarding Layer
IOFSL *Framework*

- → **FIFO**
  - → Single queue
  - → Time order
- → **HBRR**
  - → Multiple queues
  - → One per file handle
  - → Aggregate contiguous requests
  - → Select maximum n requests
- → Requests are aggregated
  - → Same **file** and same **operation**



Experimental Methodology

- Nancy site of **Grid'5000**
  - **Grimoire**

    4 nodes as PVFS servers
  - **Grisou**

    32 nodes as clients

    4 nodes as I/O forwarders
- Configuration
  - 8 core Intel Xeon E5-2630 v3
  - 128GB of RAM
  - 558GB **HDD**
  - 10Gbps Ethernet network

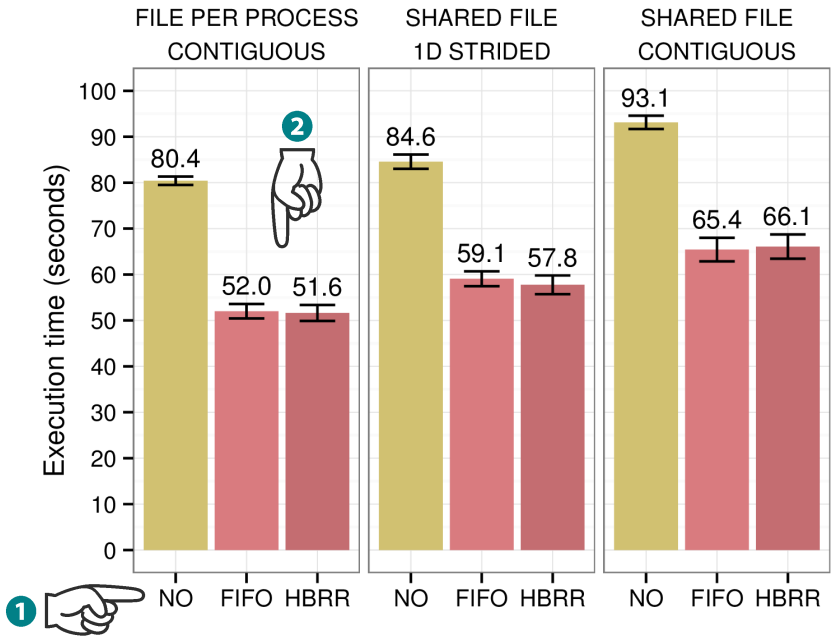
MPI-IO Test Benchmark

- Access patterns:
  - File per process
  - Shared file (1D strided and contiguous)
- 128 processes (4 per node)
- Small (32KB) and large (256KB) requests
- 4GB per test (32MB per process)
- **MAKESPAN**
- At least 8 repetitions
- 99.7% confidence interval

II

Performance Evaluation

Baseline Schedulers



→ Improvements with **READ**

→ Is it because of the **aggregations**?

	Contiguous		1D-strided	
	READ	**WRITE**	**READ**	**WRITE**
Leaving clients	32KB	32KB	32KB	32KB
Leaving I/O nodes	**58KB**	**58KB**	**58KB**	**58KB**
Arriving at servers	43KB	44KB	50KB	49KB

→ Aggregation is not the main factor here

→ **HBRR** does not outperform **FIFO**

→ **READ** requests arrive at a faster pace

→ 26.09μs vs. 50.92μs (**WRITE**)

→ Forwarding layer **funnels** the requests

TWINS: *Coordinating Server Access*

At any given moment, the following **two conditions** hold **true**:

I. an I/O node is focusing its accesses to **only one** data server

II. **different** I/O nodes are focusing on **different** servers

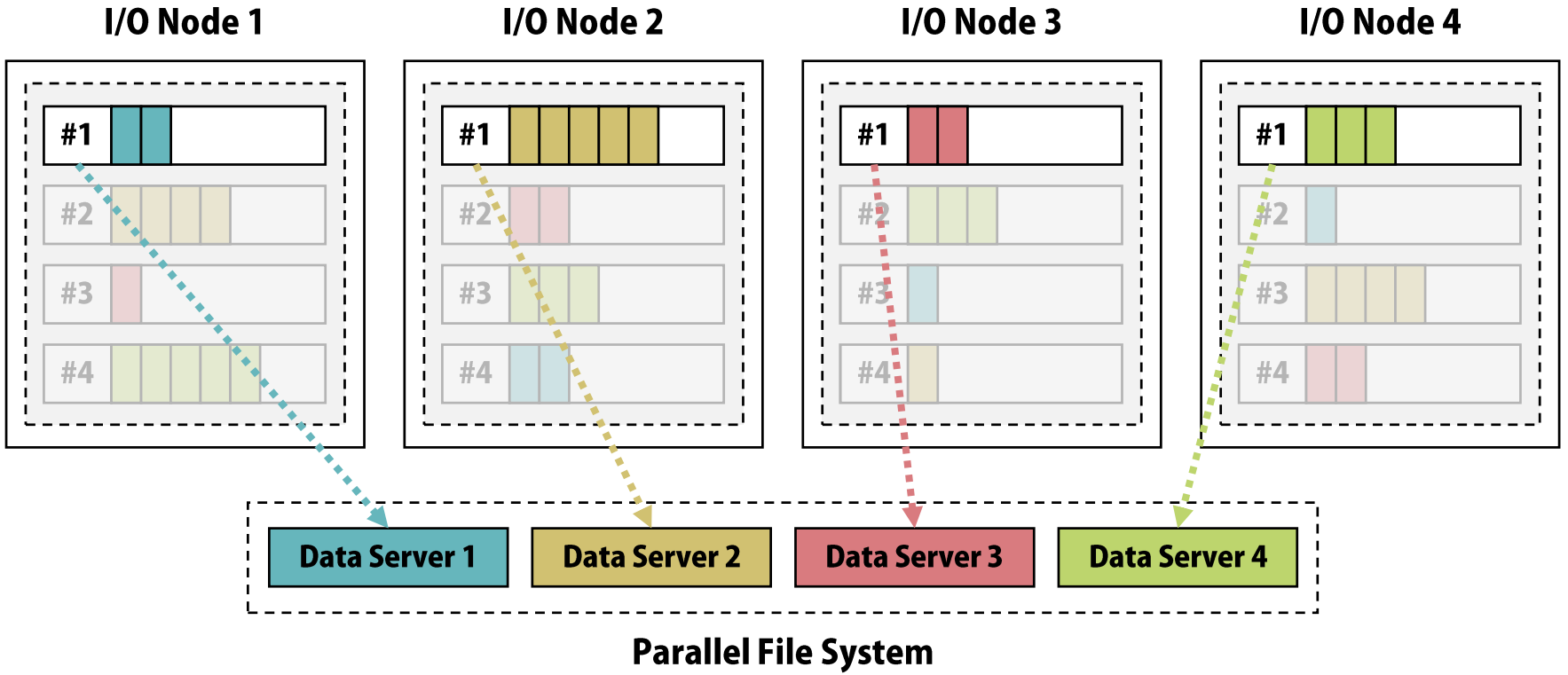
→ Multiple requests queues, **one per data server**

→ **Dedicated** time window for each data server

→ Possible additional waiting times

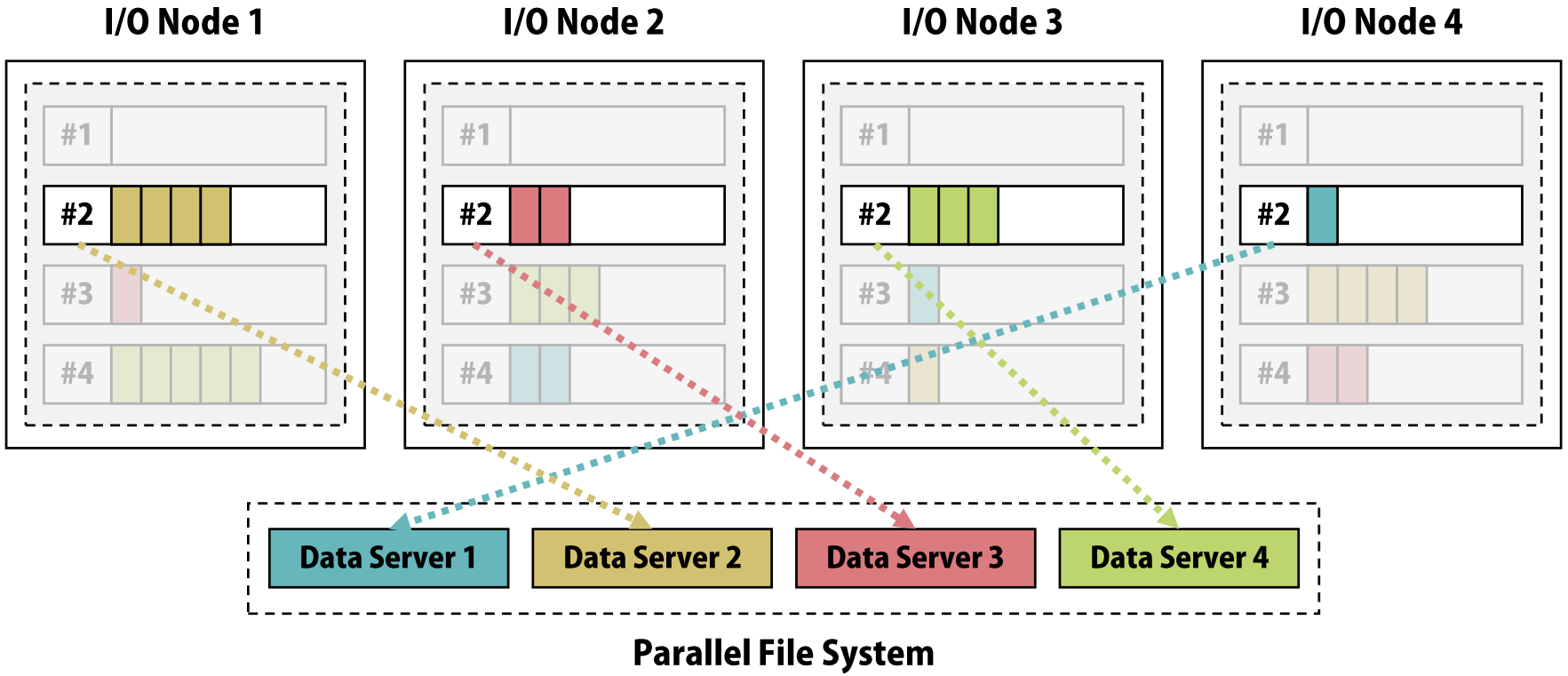
III

TWINS: *Coordination*

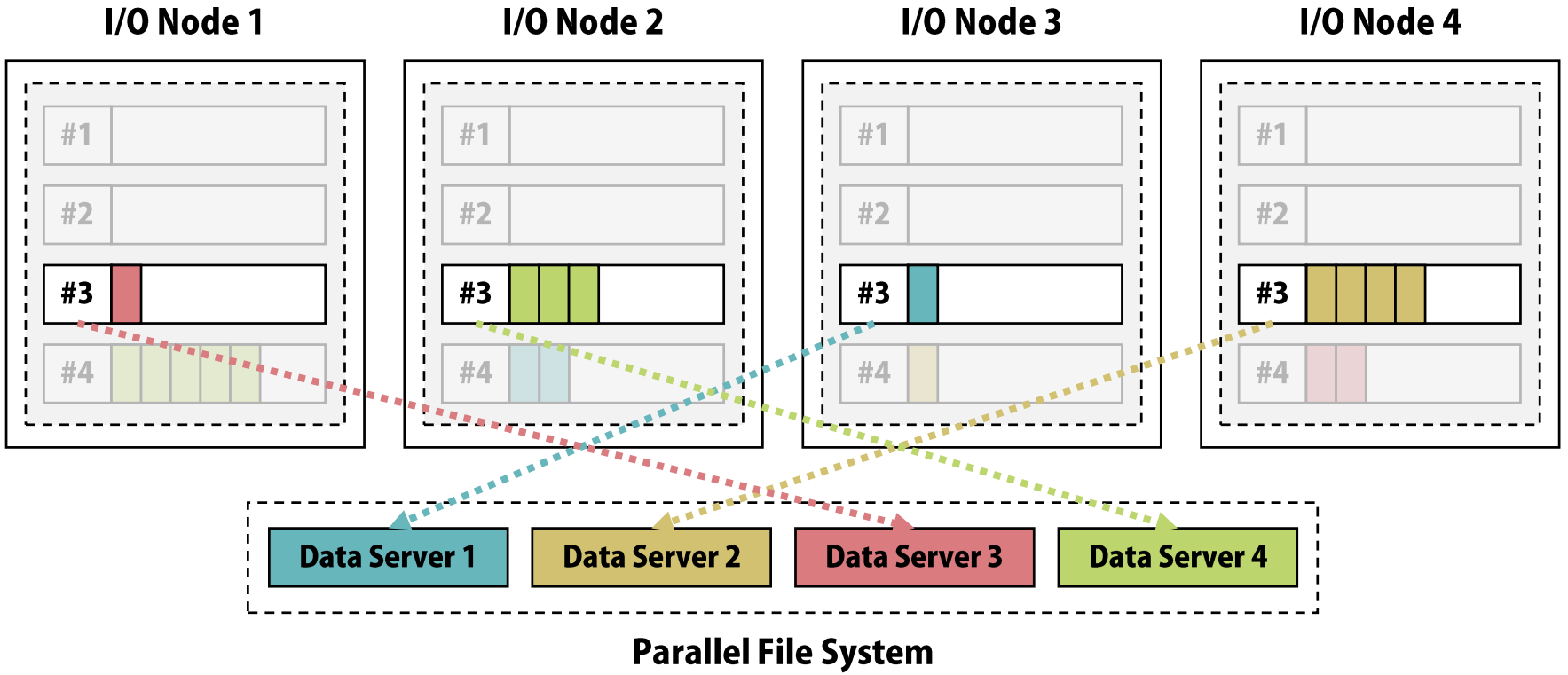


III

TWINS: *Coordination*

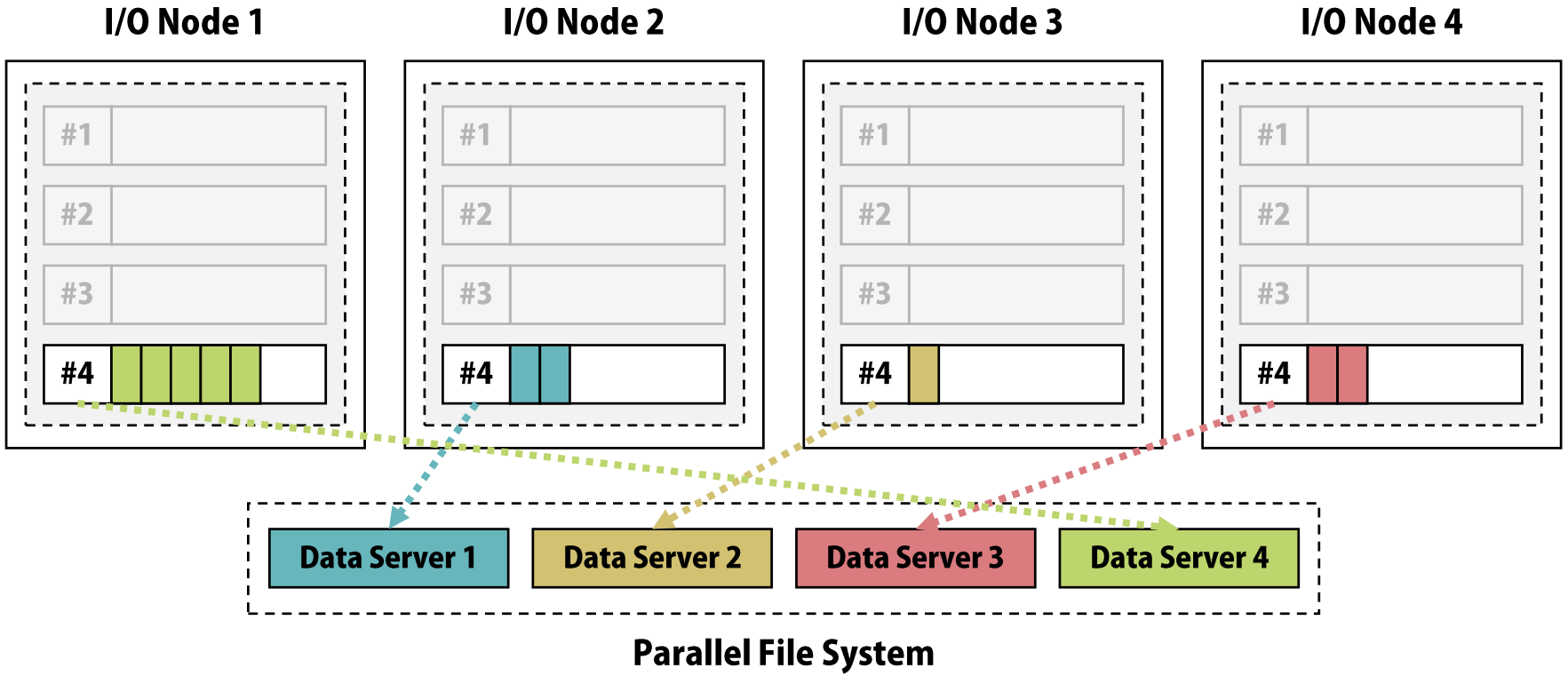


TWINS: *Coordination*



III

TWINS: *Coordination*

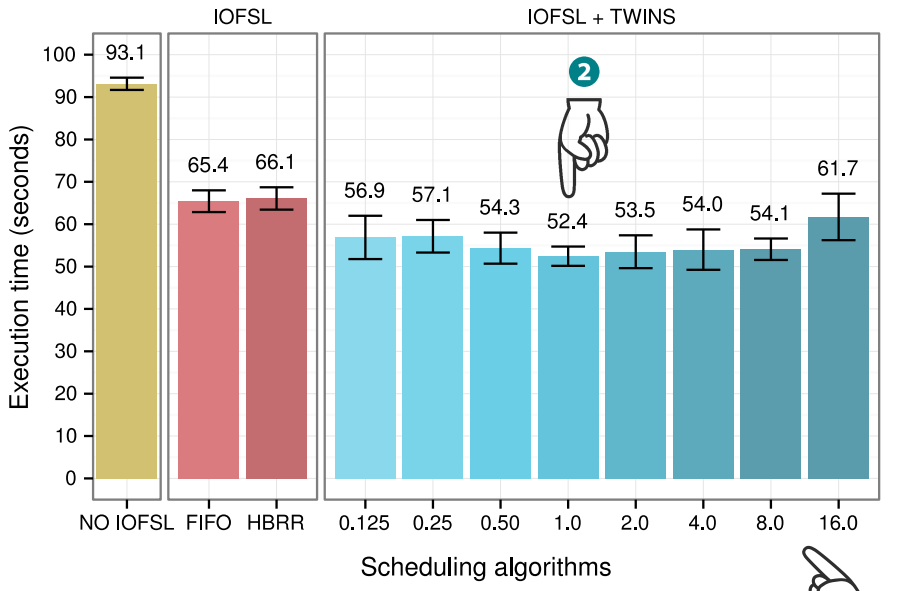


III

Performance Evaluation

TWINS *Read Requests*

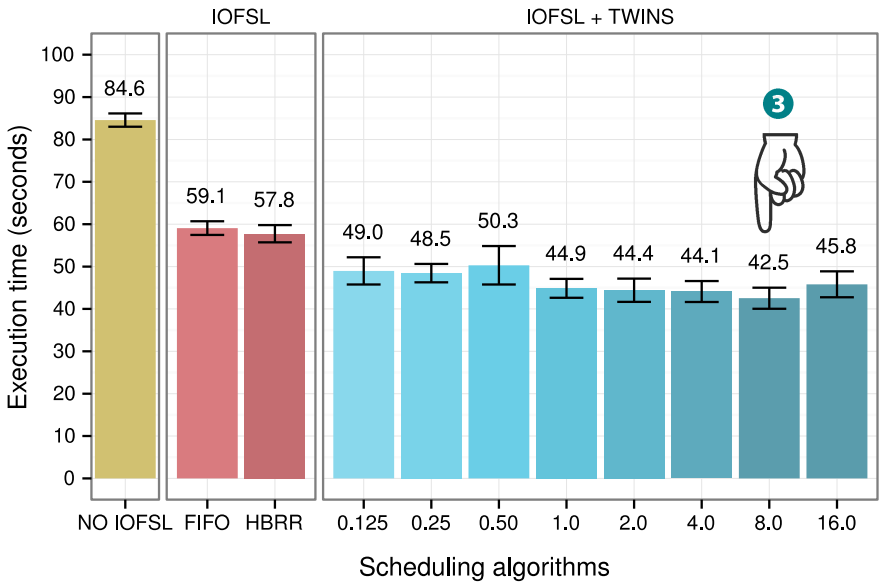
SHARED-FILE (contiguous)



48%
NO FORWARDING

20%
BASELINE

SHARED-FILE (1D-strided)



50%
NO FORWARDING

28%
BASELINE

Performance Evaluation

TWINS *Requests*

SHARED FILE SCENARIO

- → 1D strided small **READ** requests: **8ms window**
  - → **50%** over NO, **28%** over FIFO
  - → Process start their accesses on different servers
  - → Behavior is kept throughout the execution
  - → Scheduler has requests to all servers to perform a meaningful coordination
- → Contiguous small **READ** requests: **1ms window**
  - → **44%** over NO, **20%** over FIFO
- → No differences with **WRITE** requests

III

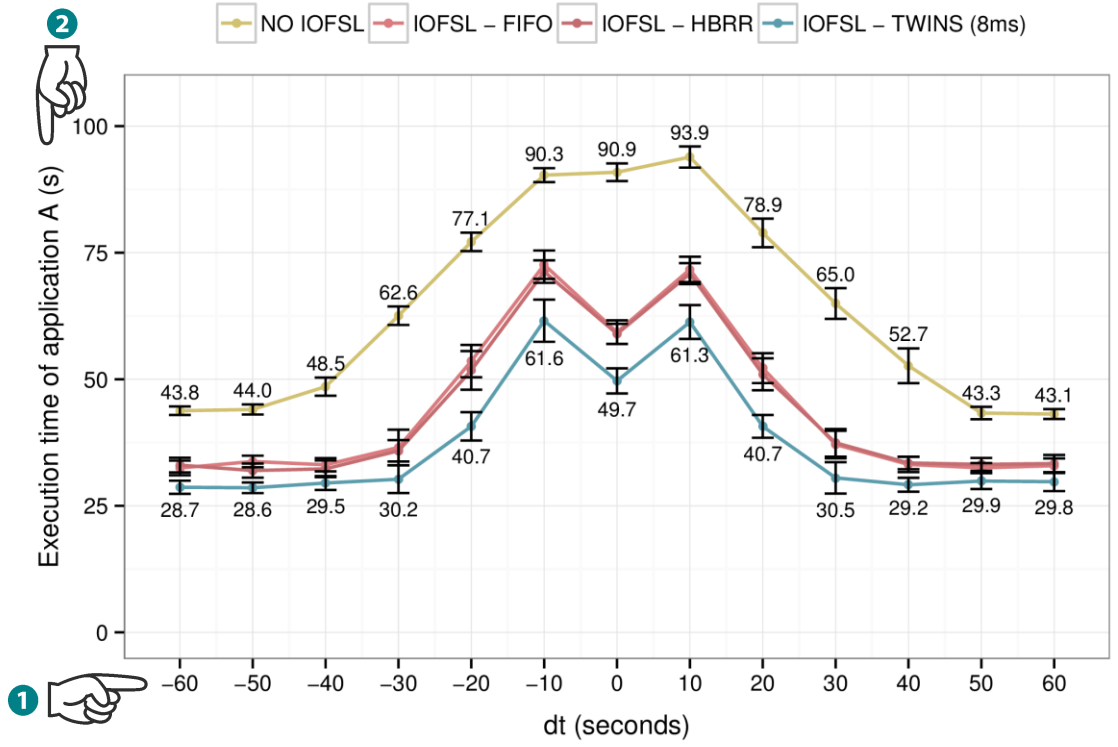
Performance Evaluation

IFER: *Multiple Applications*

- → **IFER** micro benchmark
- → 2 applications
- → 64 processes per application
- → 1D strided access pattern

45%
NO FORWARDING

16%
FIFO/HBRR



1 $dt > 0$: **A starts first**
$dt < 0$: **B starts first**

IV Conclusions

- → Existing schedulers for the I/O forwarding layer are **partially effective**
- → We proposed the **TWINS** scheduler to **coordinate** accesses
- → Improvements of up to **28%** on the **shared file READ** pattern over alternatives
- → Gains of up to **50%** over not forwarding I/O
- → Improvements in the **multi-application** scenario
- → Does not necessarily harm performance in other scenarios

Thank you!

Coordinating Server Access at the I/O Forwarding Layer

TWINS: Server Access Coordination in the I/O Forwarding Layer, PDP 2017

Jean Luca Bez, Francieli Z. Boito, Lucas M. Schnorr, Jean-François Mehaut, Philippe O. A. Navaux

{jean.bez, fzboito, schnorr, navaux}@inf.ufrgs.br
jean-francois.mehaut@imag.fr

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

.inf
INSTITUTO DE INFORMÁTICA UFRGS

HPC4E

Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr/). This research was accomplished in the context of the International Joint Laboratory LICIA. Research has received funding from the EU H2020 Programme and from MCTI/RNP-Brazil under the HPC4E Project, grant agreement n 689772. This work was also supported by STIC-AmSud/CAPES scientific cooperation program under EnergySFE research project grant 99999.007556/2015-02. The authors would also like to thank Kamil Iskra, Rob Latham, and Rob Ross from the Argonne National Laboratory for insights about the IOFSL framework.