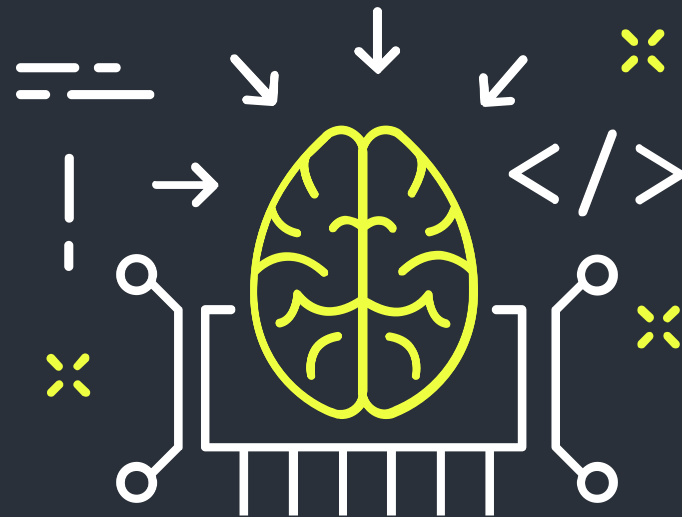


A Reinforcement Learning Strategy to Tune Request Scheduling at the I/O Forwarding Layer

Jean Luca Bez, Francieli Zanon Boito, Ramon Nou,
Alberto Miranda, Toni Cortes, and Philippe O. A. Navaux

jean.bez@inf.ufrgs.br

HPC-IODC HPC I/O in the Data Center Workshop 2020

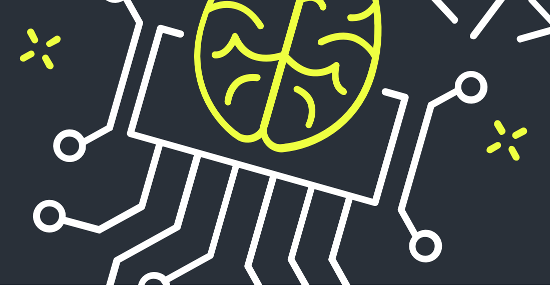
UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

informatiques mathématiques
inria

INTRODUCTION

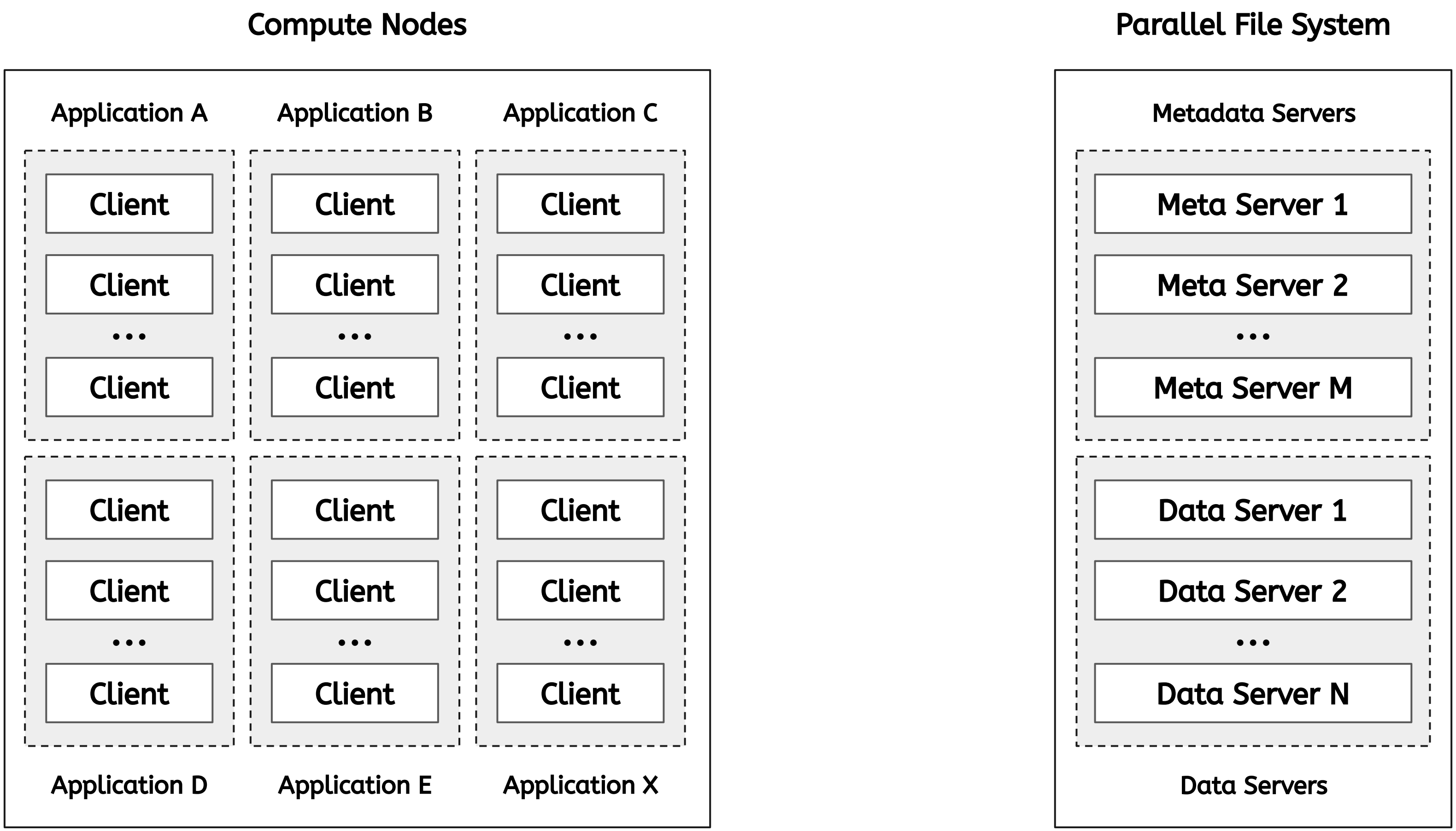
AGENDA



- The I/O Forwarding Layer
- Motivation
- Case Study: TWINS Scheduling Algorithm
- Adaptive I/O Forwarding Layer
- Results
- Conclusion

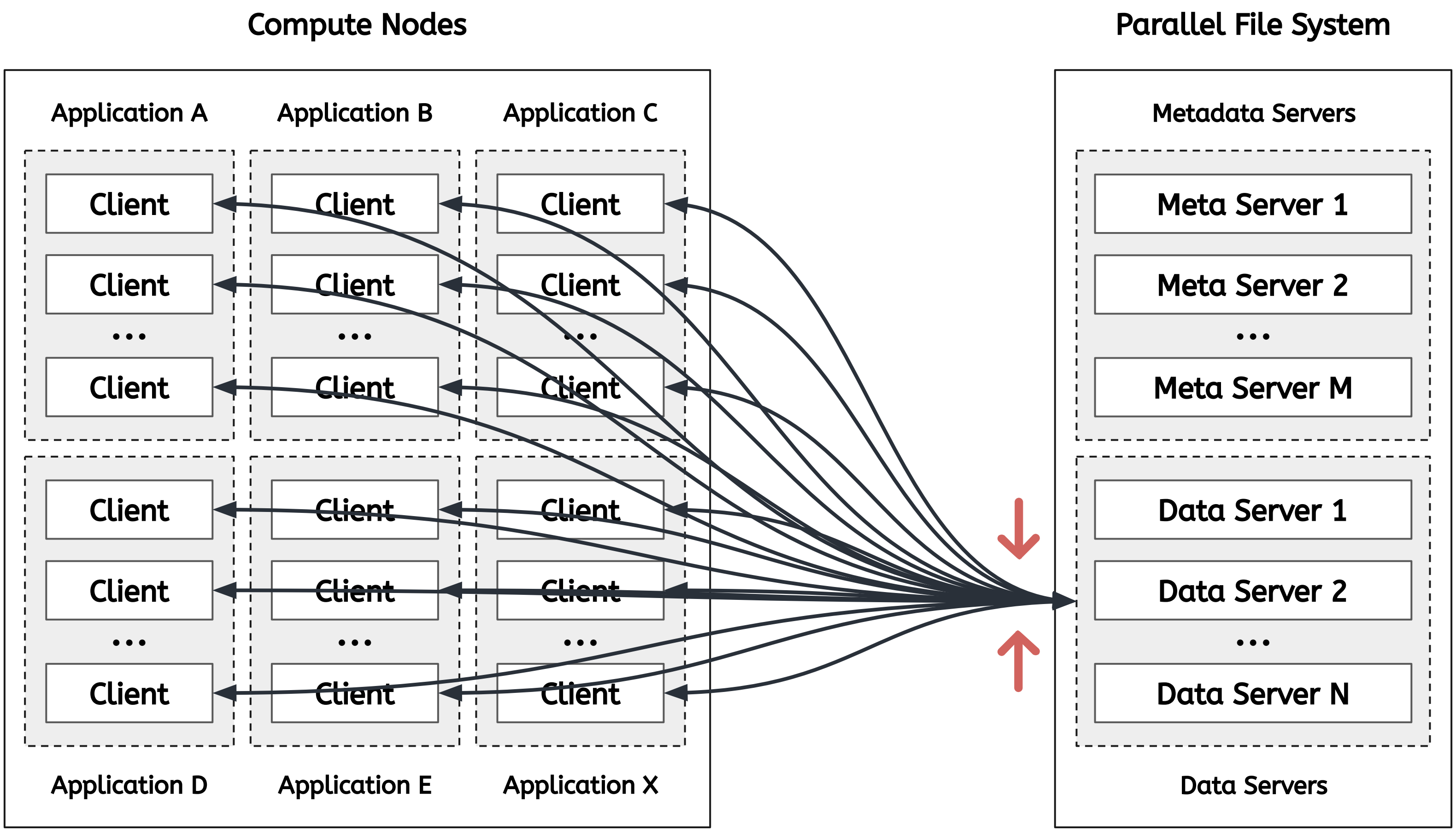
INTRODUCTION

The I/O Forwarding Layer



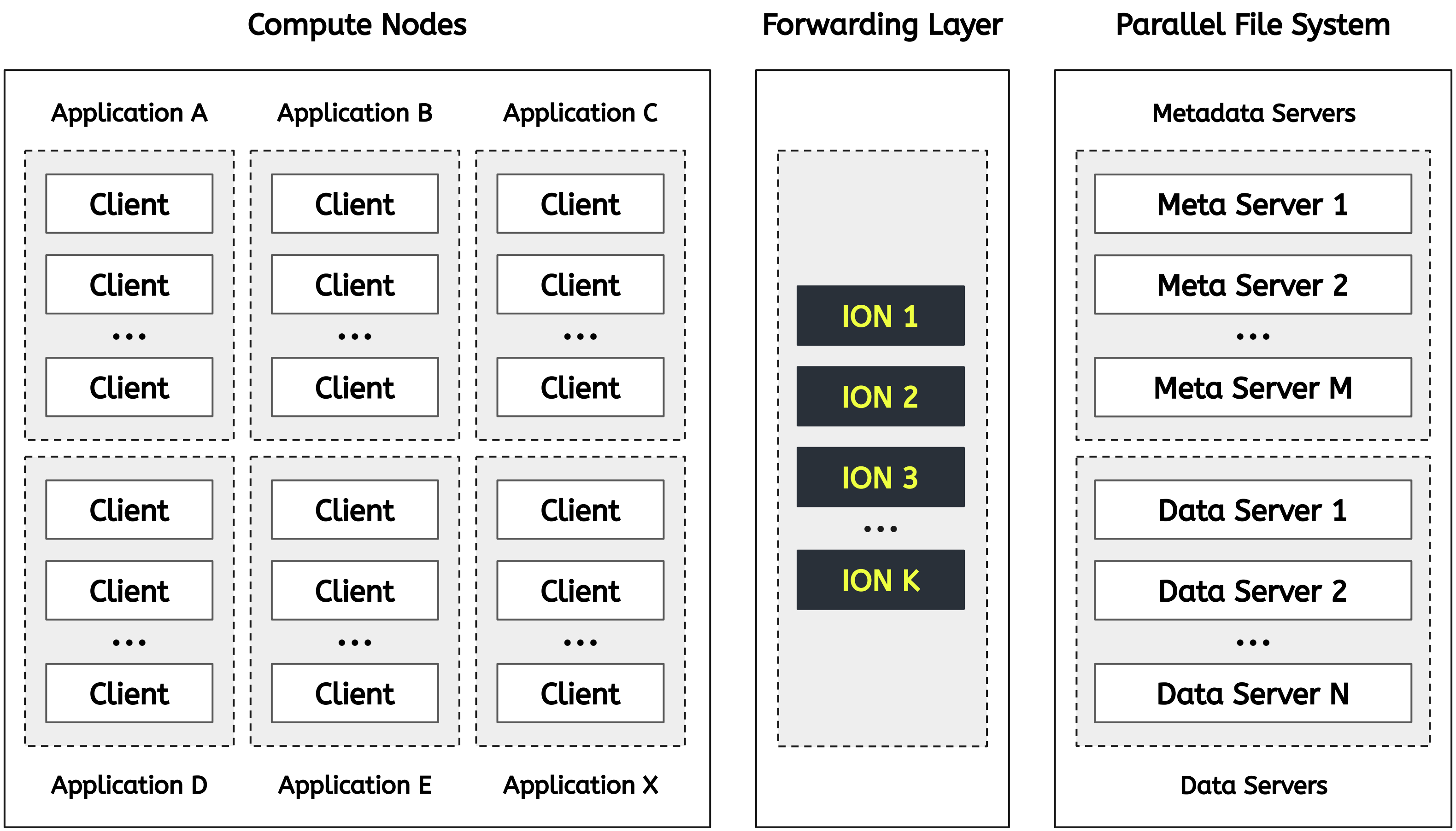
INTRODUCTION

The I/O Forwarding Layer



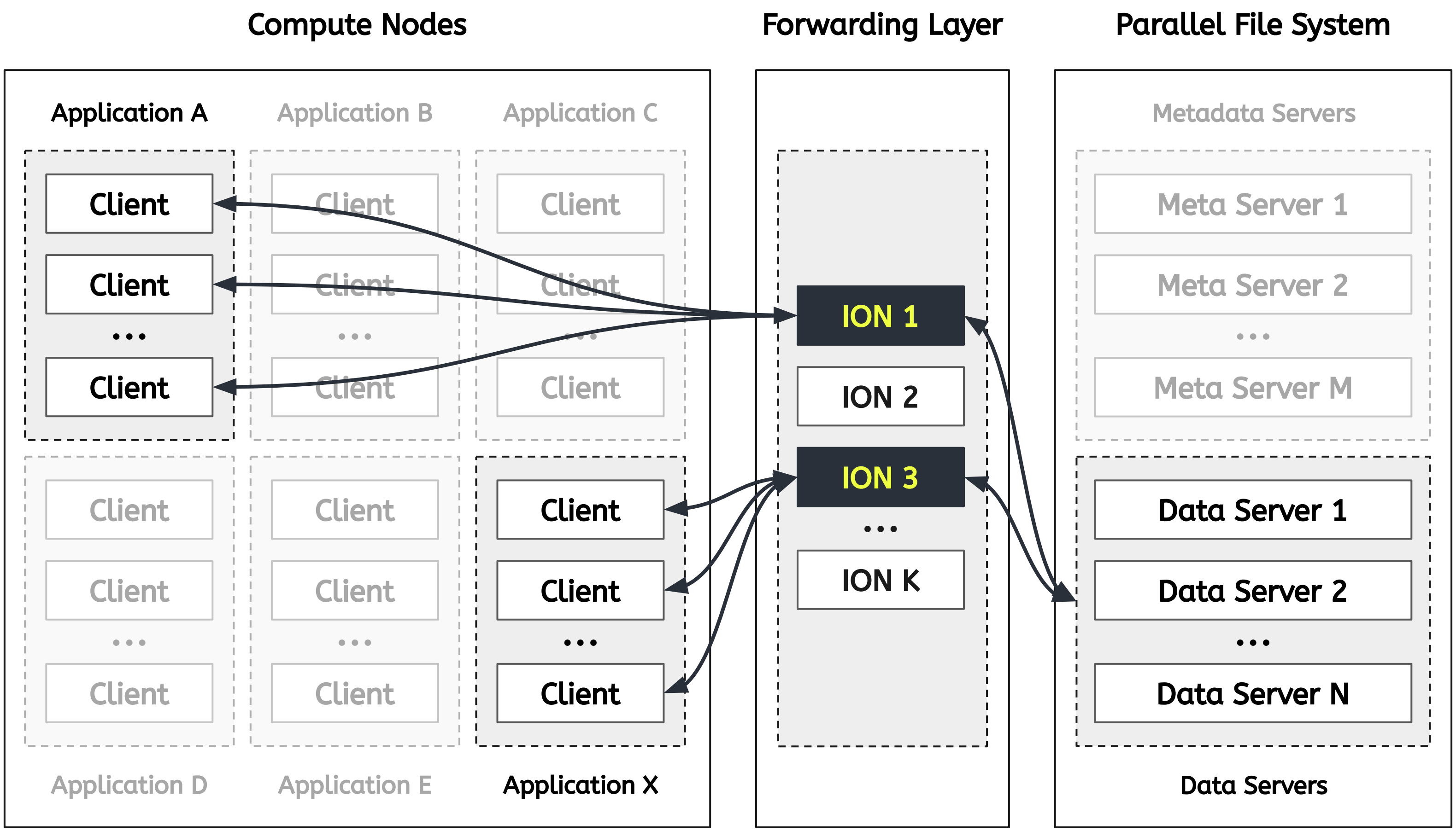
INTRODUCTION

The I/O Forwarding Layer



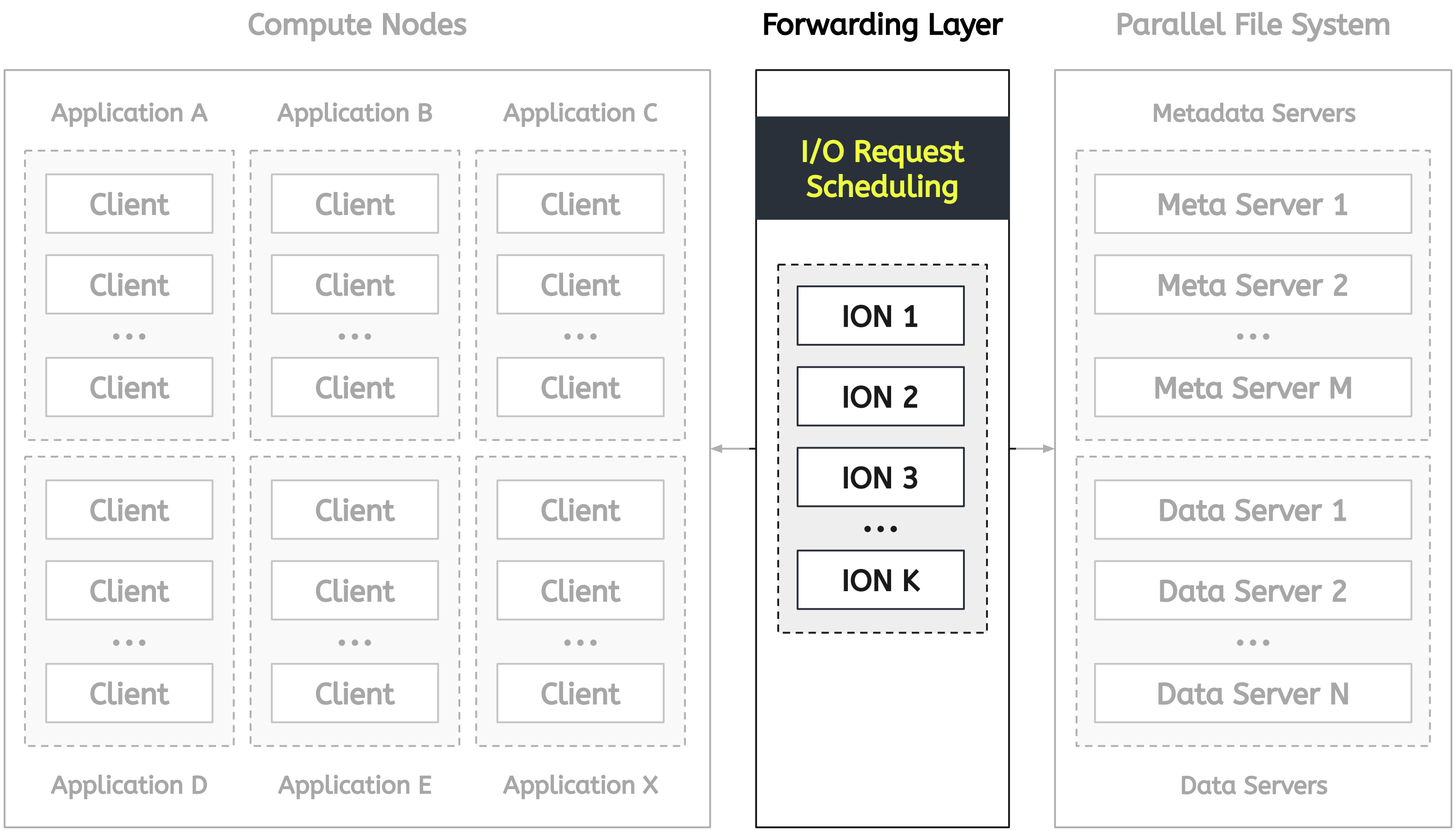
INTRODUCTION

The I/O Forwarding Layer



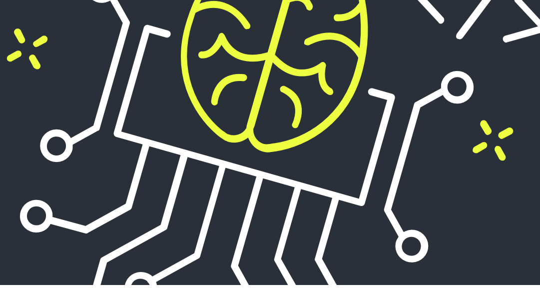
INTRODUCTION

The I/O Forwarding Layer



MOTIVATION

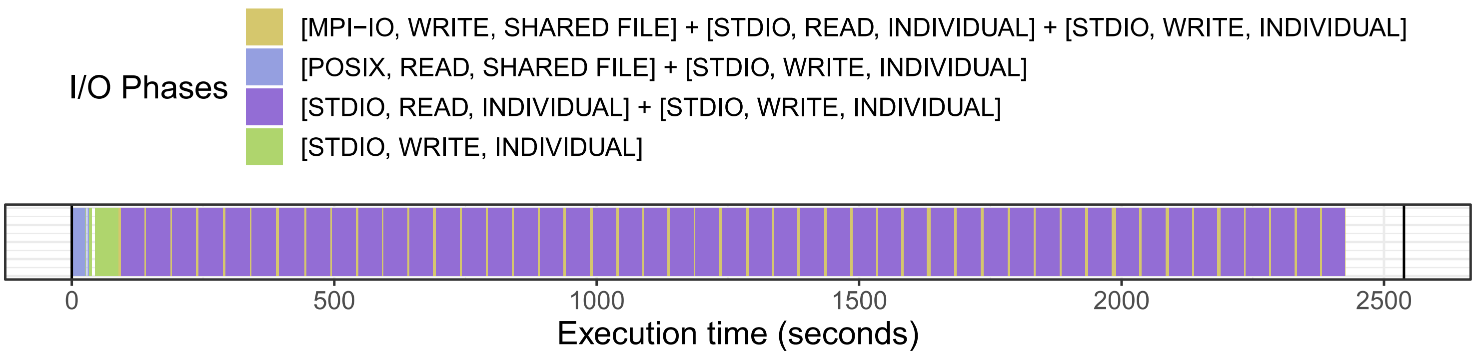
Adaptation



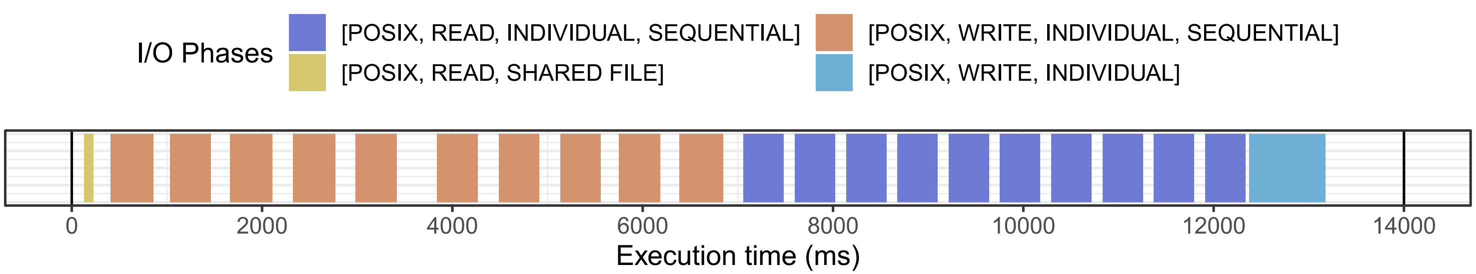
- I/O optimization techniques
  - Fail to provide improvements for all access patterns
  - Designed to explore specific system and workload characteristics
- Essential to adapt the system to a changing workload
- **Our goal is to adapt the forwarding layer to the current I/O workload**
  - Access pattern detection
  - Reinforcement learning technique
  - During runtime
- Tune any optimization technique that depends on the **access pattern**

MOTIVATION

I/O Characterization



The Ocean-Land-Atmosphere Model (OLAM) application at the Santos Dumont Supercomputer (LNCC)



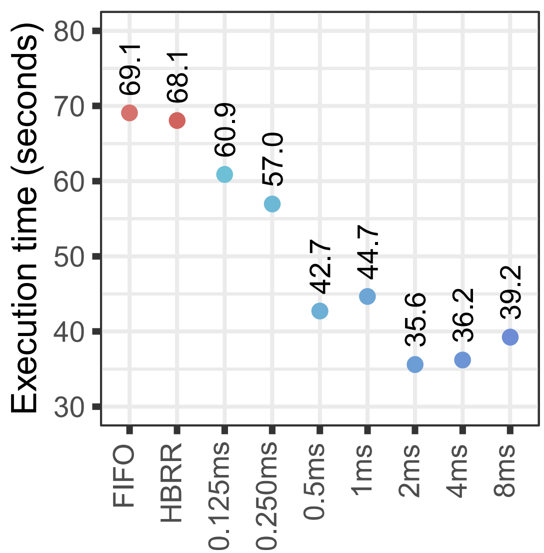
Application 2201660091 (job 15335183665324813784) running in the Intrepid supercomputer at Argonne National Laboratory (ANL)

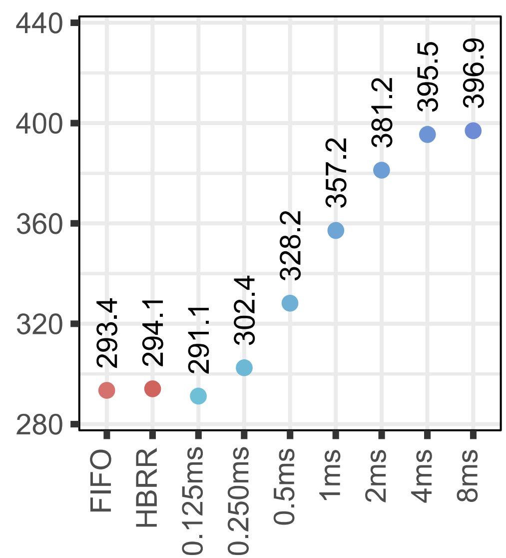
INTRODUCTION

TWINS Scheduler

- **Coordinate** the I/O nodes access to the shared PFS servers
  - Multiple request queues in each I/O node, one for each data server
  - **TIME WINDOW** dedicated to forward requests to a given data server
- Increase in performance by 48% over IOFSL's default schedulers
- The choice of window size is of **paramount** importance

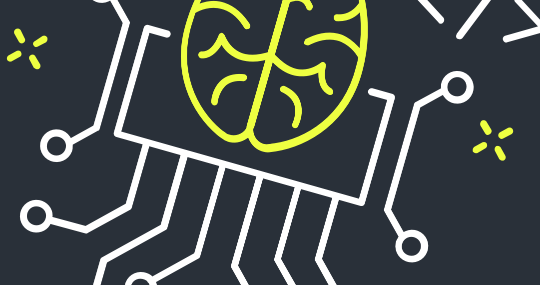
READ
8 I/O nodes





WRITE
2 I/O nodes

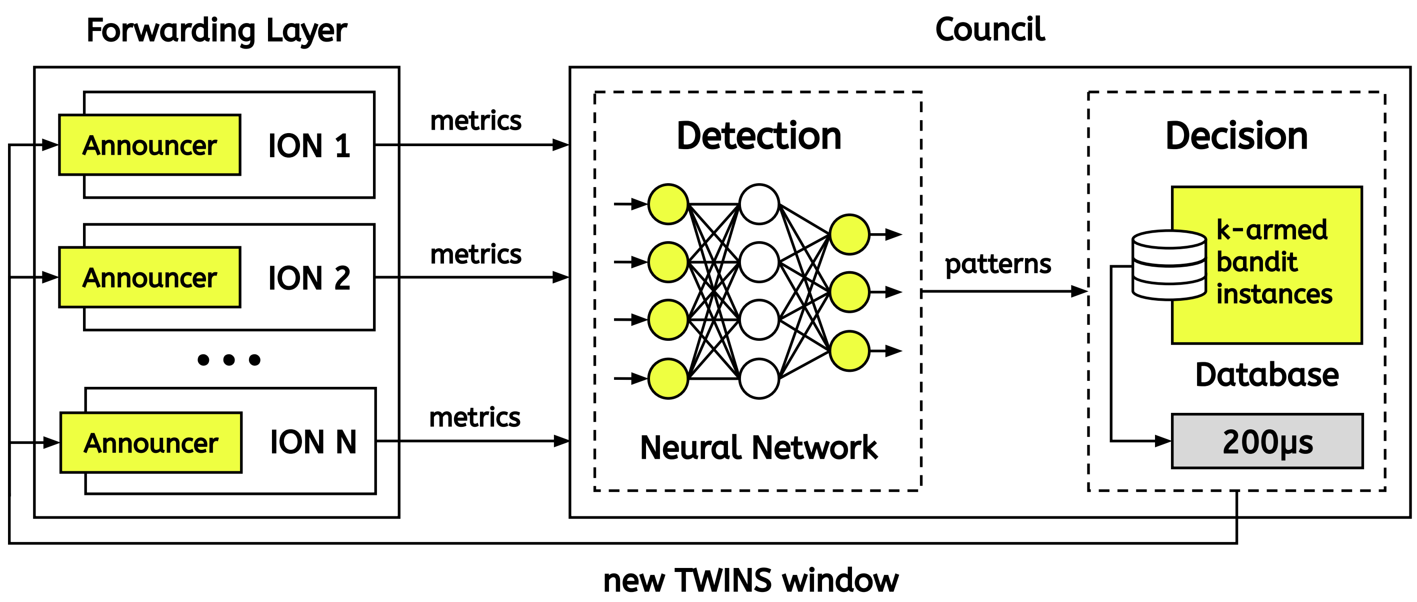
Reinforcement Learning



- Learn the best choice for different situations **while** they are observed
- Reinforcement Learning problem: **k-armed bandit**
  - At each step an agent takes one of the possible k actions and receives a reward
  - For TWINS each action represents a different time window duration
  - **Exploration** and **exploitation**
- **Contextual bandits**
  - Multiple “instances” of the k-armed bandit
  - One for each access pattern
  - ε-greedy at step *t* takes action *a* with probability (1-ε) or ε to random select an action
- Learning is **not limited** by the execution of the application

ADAPTIVE I/O FORWARDING

Proposed Solution

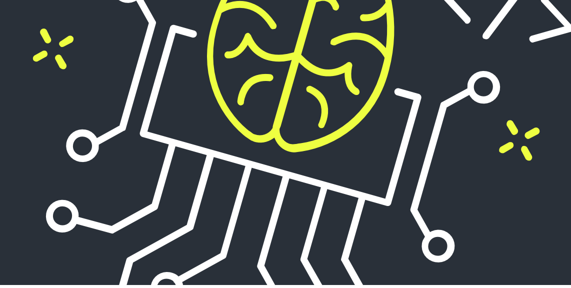


Experimental Setup

- 2 clusters from **Grid'5000** platform: **Grimoire** and **Grisou**
- 4 **PVFS** servers on Grimoire
- 1, 2, 4, and 8 **IOFSL** servers on Grisou
- 32 clients on Grisou
- **MPI-IO Test** benchmarking tool
  - Number of processes: 128, 256, and 512
  - File layout: shared file or file-per-process
  - Spatiality: contiguous or 1D-strided
  - Operation: read or write
  - Request size: 32KB or 256KB (smaller or larger than the PFS stripe size)
- **144** scenarios with **7 window sizes** = 1,008 experiments
- Metrics collected in each I/O node **every second** (>1 million observations)

ADAPTIVE I/O FORWARDING

Offline Evaluation

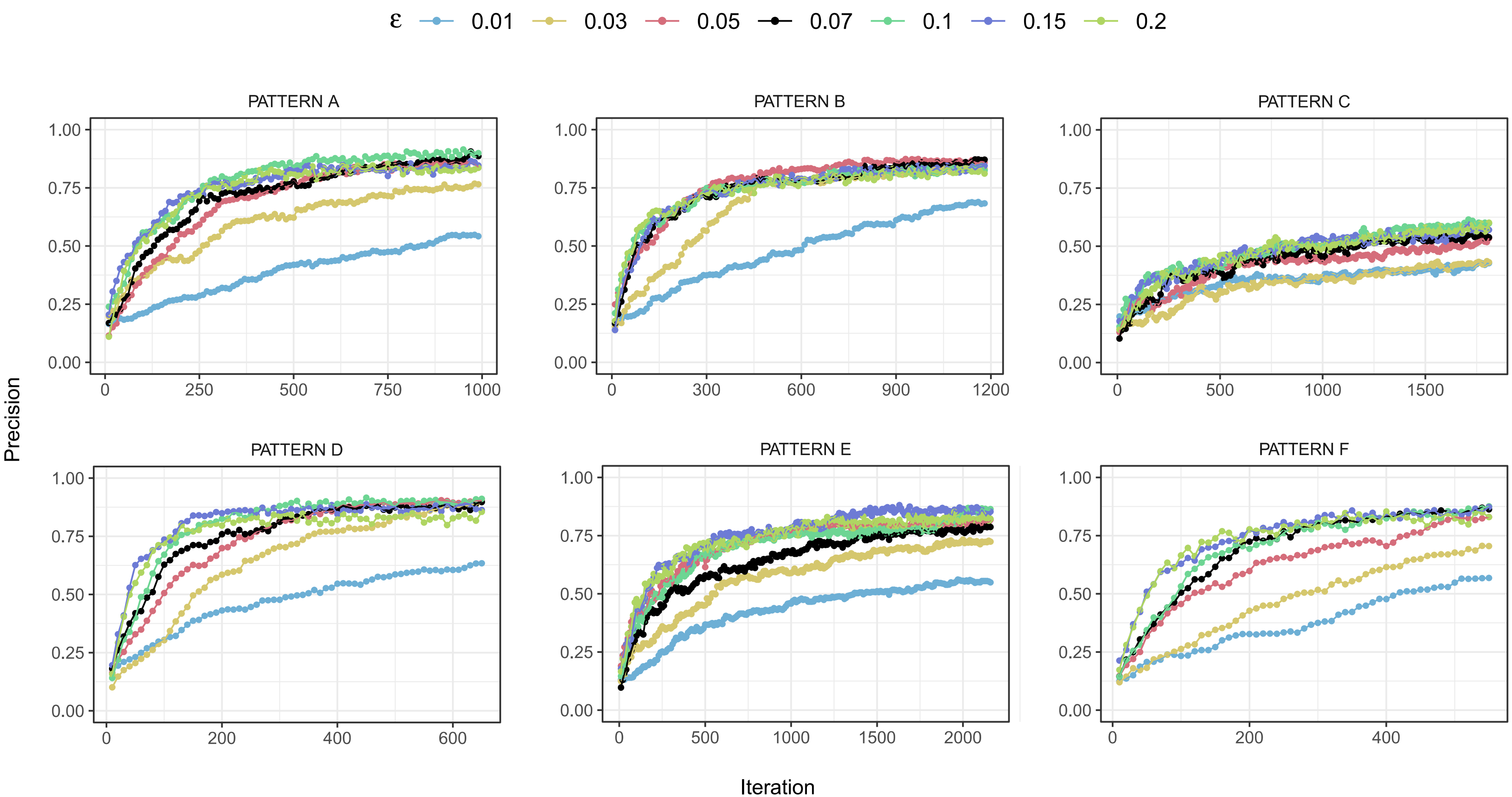


- Simulation of **ε-greedy** policy
- Assume perfect pattern detection
- Use previously collected real measurements
- **100** simulations

	I/O Nodes	Processes	File Layout	Request Spatiality	Request Size	Operation
A	8	128	Shared	1D-strided	32KB	read
B	2	128	Shared	Contiguous	32KB	write
C	8	512	Shared	Contiguous	32KB	read
D	1	128	Shared	1D-strided	32KB	write
E	1	128	Individual	Contiguous	32KB	write
F	4	128	Shared	1D-strided	32KB	read

ADAPTIVE I/O FORWARDING

Offline Evaluation



Offline Evaluation

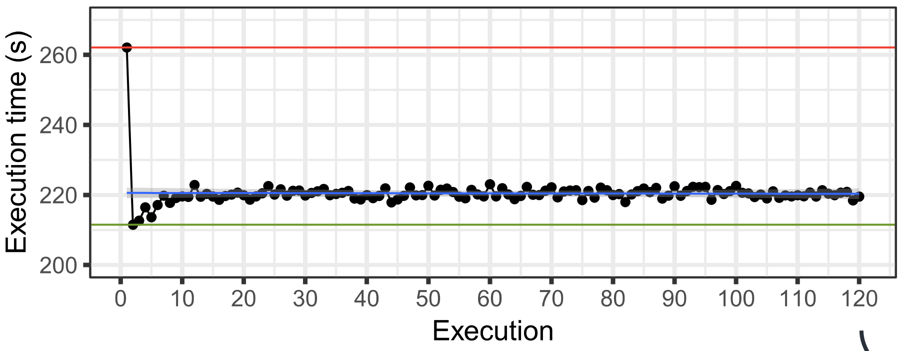
- The smaller the **ε** the slower the convergence
- **ε** of 0.15 only chooses the best action **85% of the time**

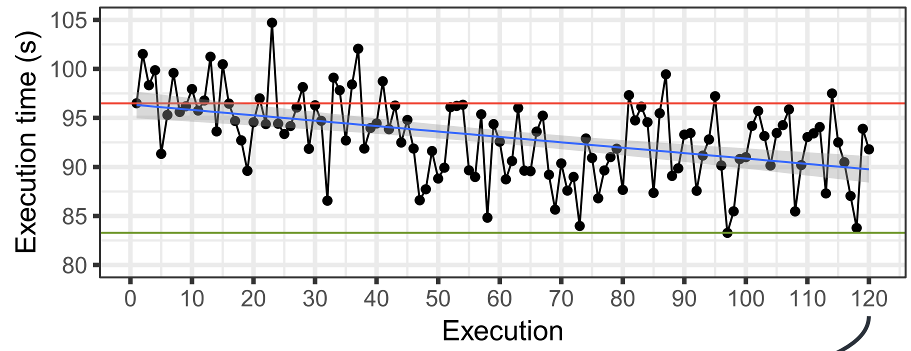
Pattern	**A**	**B**	**C**	**D**	**E**	**F**
Precision	0.88	0.88	0.49	0.87	0.59	0.59
Performance	0.99	0.96	0.96	0.97	0.98	0.92

- Easier to learn where there is a clear better choice
- Selecting a value similar to the best choice also yields performance

RESULTS

Live Adaptation



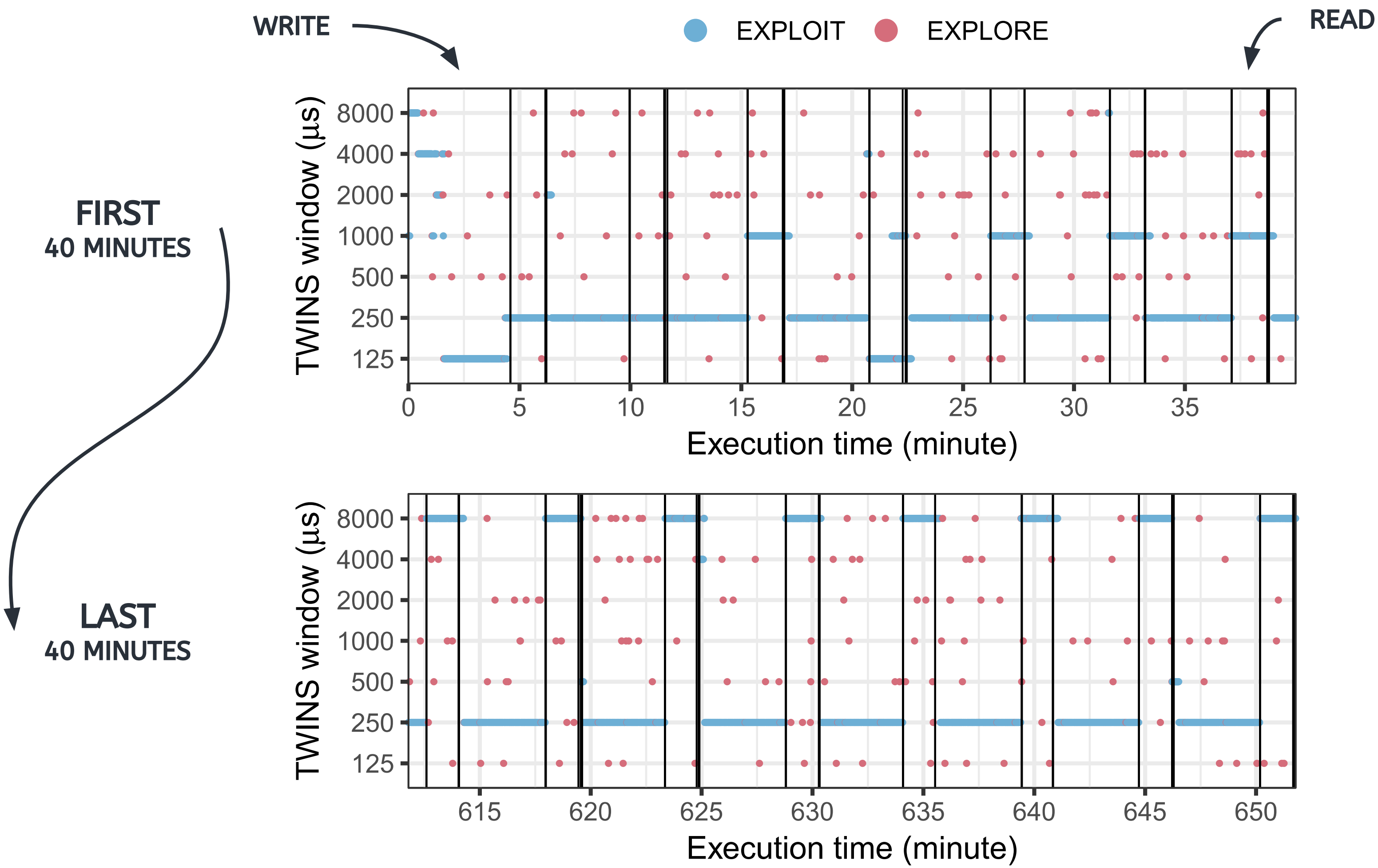


Scenario F
120 repetitions

- **WRITE**
  - In the first execution (first ~260s)
- **READ**
  - Phases are 60% shorter, thus less iterations
  - Delay of 1s to detect a phase has changed
  - Read time presents a higher variability which adds noise to the learning
  - Bad decision (during exploration) have bigger impact on reads

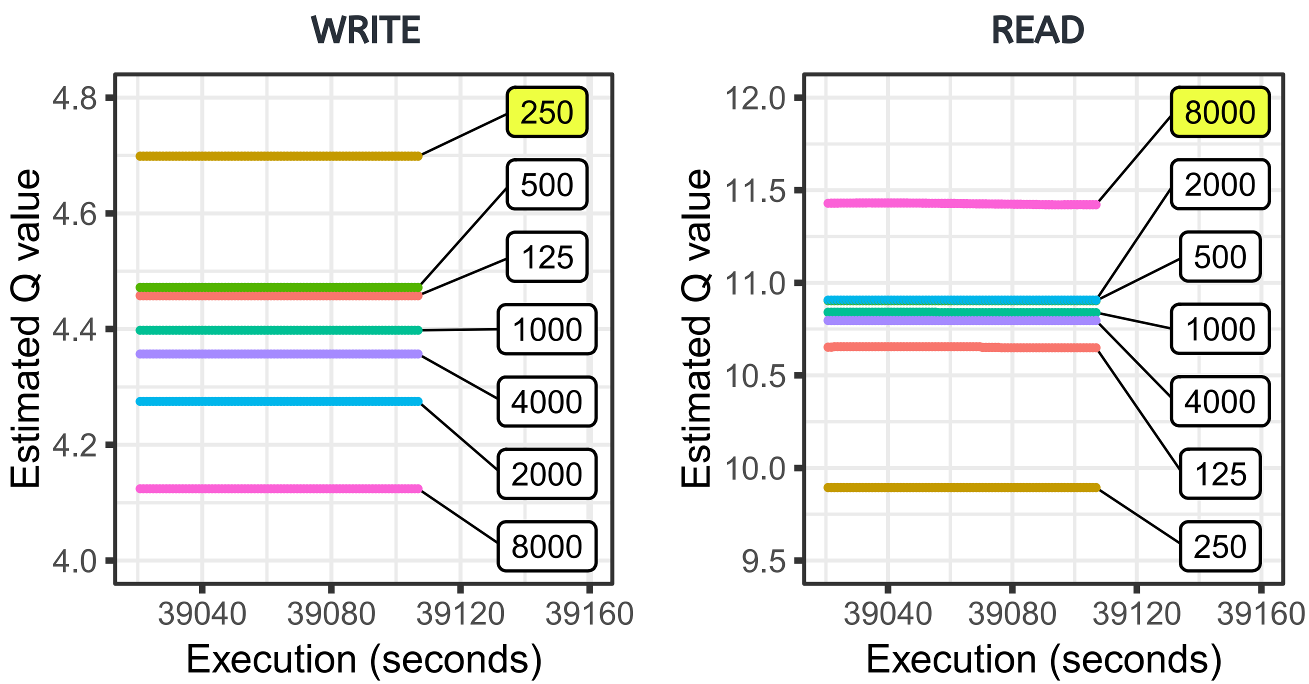
RESULTS

Live Adaptation



RESULTS

Estimated Q Values

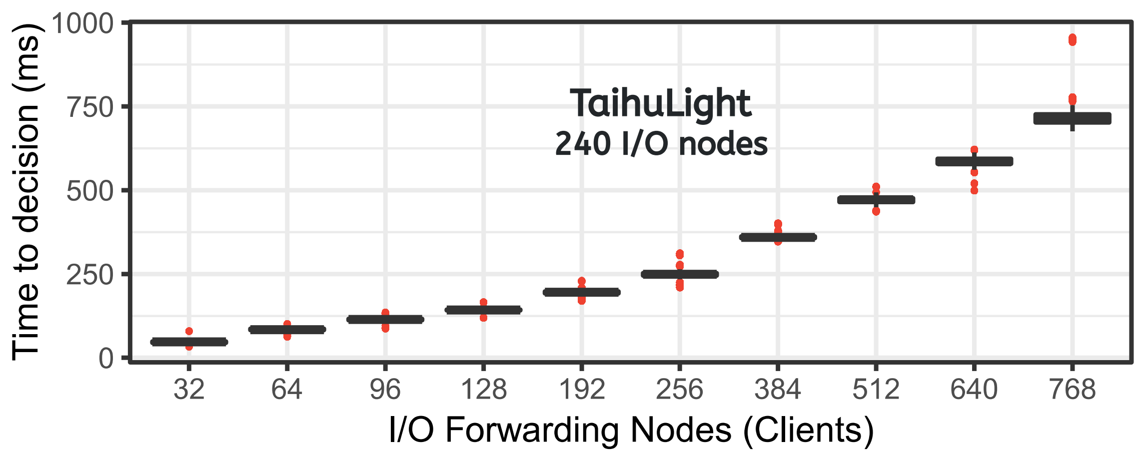


- The best choice is 250µs for WRITE phases and 8ms for READ phases
- **Improve performance for all applications that share the learned pattern**

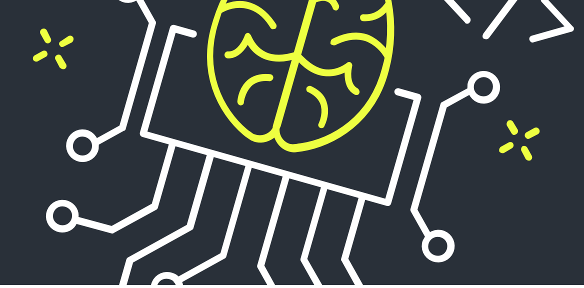
RESULTS

Overhead and Scalability

- Repeated the 144 experiments **ignoring** the decisions
- **Overhead** observed only for **65 scenarios (median < 2%)**
- Centralized Council's can handle a large of number of I/O nodes
  - Average of 60 executions for each number of clients reporting metrics



Conclusion



- Proposed an approach to **adapt the I/O forwarding layer**
- System can learn the best choice during **runtime**
  - Remove the burden from the user to find the tuned parameter
  - Re-use learned information for all applications that share similar patterns
- TWINS rely on the correct window size
  - Depends on the system configuration and application's access pattern
- **~88% precision** reaching **~99%** of the **performance** of the best option
- Solution is **not** specific to **TWINS**

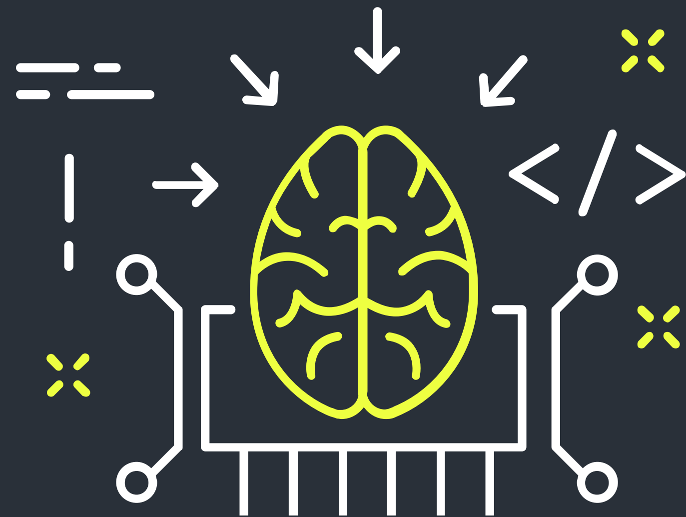
jeanbez.gitlab.io/adaptative-io-scheduling

https://doi.org/10.1016/j.future.2020.05.005

ACKNOWLEDGMENTS

This study was financed in part by the **Coordenação de Aperfeiçoamento de Pessoal de Nível Superior** - Brasil (CAPES) - Finance Code 001. It has also received support from the **Conselho Nacional de Desenvolvimento Científico e Tecnológico** (CNPq), Brazil; the **LICIA** International Laboratory; NCSA-Inria-ANL-BSC-JSC-Riken **Joint-Laboratory on Extreme Scale Computing** (JLESC); the **Spanish Ministry of Science and Innovation** under the TIN2015−65316 grant; and the **Generalitat de Catalunya** under contract 2014−SGR−1051. This research received funding from the **European Union's Horizon 2020** research and innovation programme under the **Marie Skłodowska-Curie** grant agreement No. 800144. Experiments presented in this paper were carried out using the **Grid'5000** testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (www.grid5000.fr).

This project is partially funded by the European Union.



A Reinforcement Learning Strategy to Tune Request Scheduling at the I/O Forwarding Layer

Jean Luca Bez, Francieli Zanon Boito, Ramon Nou,
Alberto Miranda, Toni Cortes, and Philippe O. A. Navaux

jean.bez@inf.ufrgs.br

HPC-IODC HPC I/O in the Data Center Workshop 2020

UFRGS
UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

inria
informatiques mathématiques