RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

université de BORDEAUX

LaBRI

BORDEAUX INP

cea

cnrs

Inria

FRANCE 2030
PROGRAMME DE RECHERCHE
NUMÉRIQUE POUR L'EXASCALE

Towards I/O Resource Management for HPC

Francieli Boito

Associate Professor @ University of Bordeaux, France

Researcher @ LaBRI & Inria (TADaaM team)

Motivation

Parallel I/O is an important bottleneck for HPC

ASCI Red
3.2 TFlops
1GiB/s

~840k
times

El Capitan
2.7 EFlops

~10k
times

Aurora
10TiB/s

25 years ago

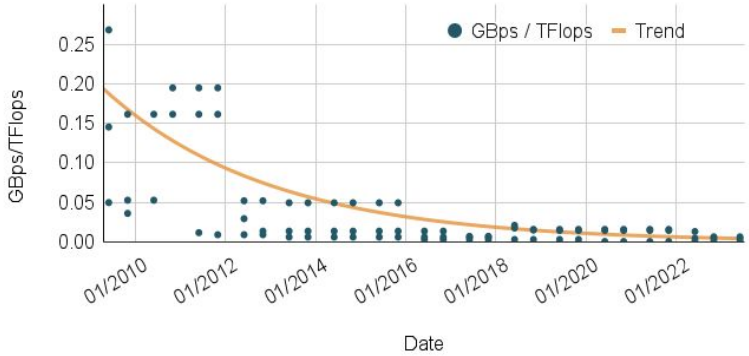
now

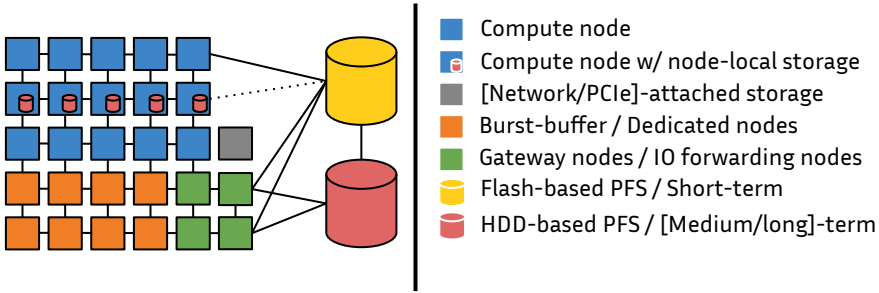
Context: parallel I/O for HPC

Processing becomes faster, the amount of data grows

- At NERSC, data volume x41 in 10 years [Lockwood et al., Storage 2020: A Vision for the Future of HPC Storage]
- Emergence of data-intensive workloads (e.g. AI)

I/O speed is a bottleneck for an increasing number of applications

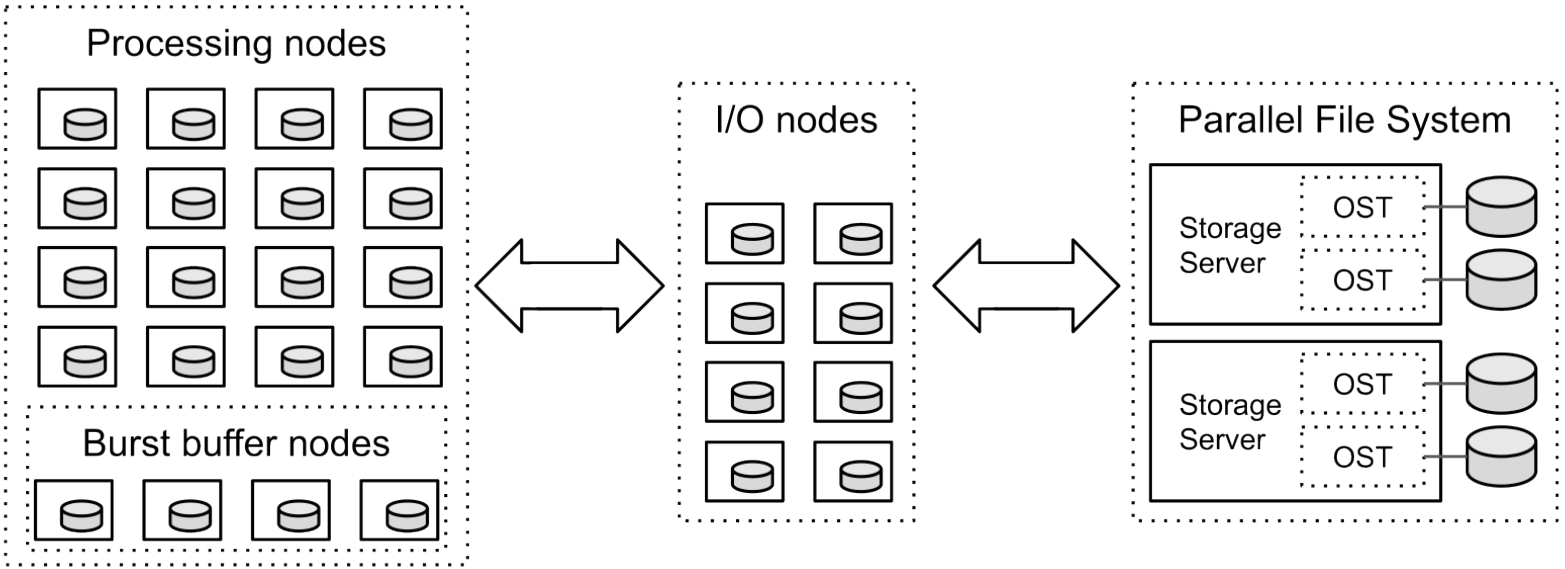




Trend in storage technologies available on extreme-scale systems

(figures by François Tessier)

Parallel I/O

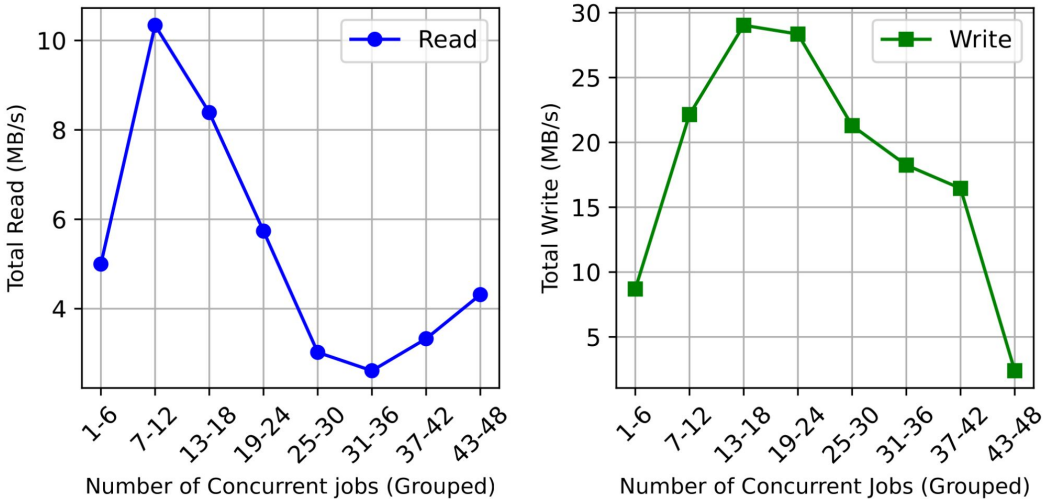


"I/O resource": OSTs, I/O nodes, burst buffers, etc.

"stripe count": the number of OSTs used for a file

I/O resource management

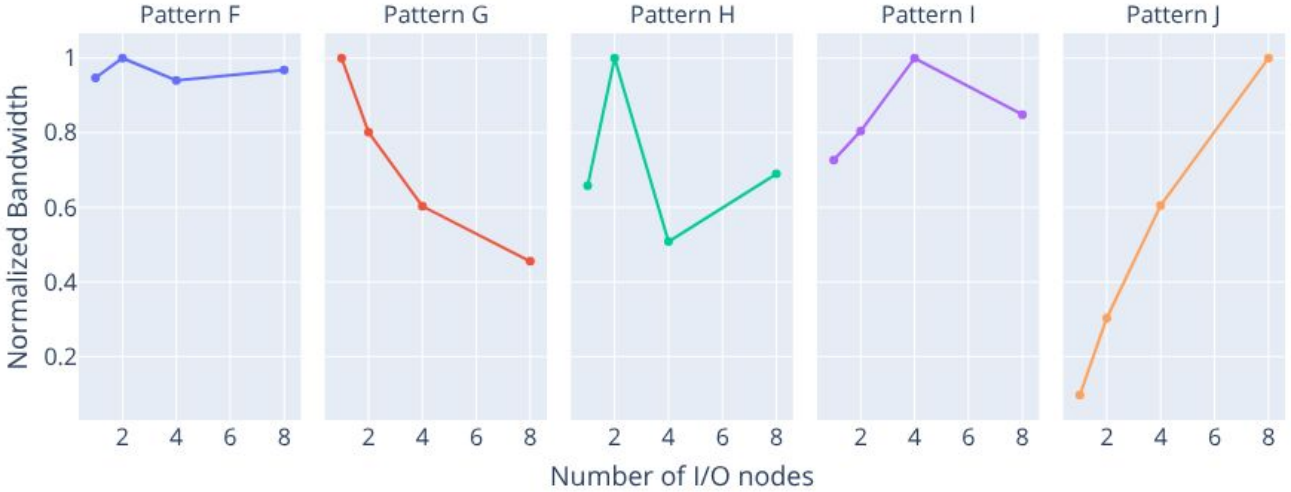
- The usual case:
  - Compute resources are allocated exclusively for jobs
  - I/O resources are NOT arbitrated
- Interference from concurrent jobs **harms performance, leads to high variability, wastes compute resources**



Average bandwidth in SDumont for different degrees of concurrency. [Boito et al. 2025]

How to do I/O resource management?

It depends on application characteristics! (and on the system)



[J.L. Bez, F. Boito, A. Miranda, R. Nou, T. Cortes, P.O.A. Navaux, Towards On-Demand I/O Forwarding in HPC Platforms, PDSW 2020]

"access pattern": the way I/O is performed - how many files, offsets, transfer sizes, etc.

I/O resource management

- Systems use a default, “general-purpose” allocation
  - stripe count of 1, number of I/O nodes that depends on the number of compute nodes, etc.
- Not enough information to do better
  - of application characteristics
  - of their impact
- **Optimization opportunities are lost!**

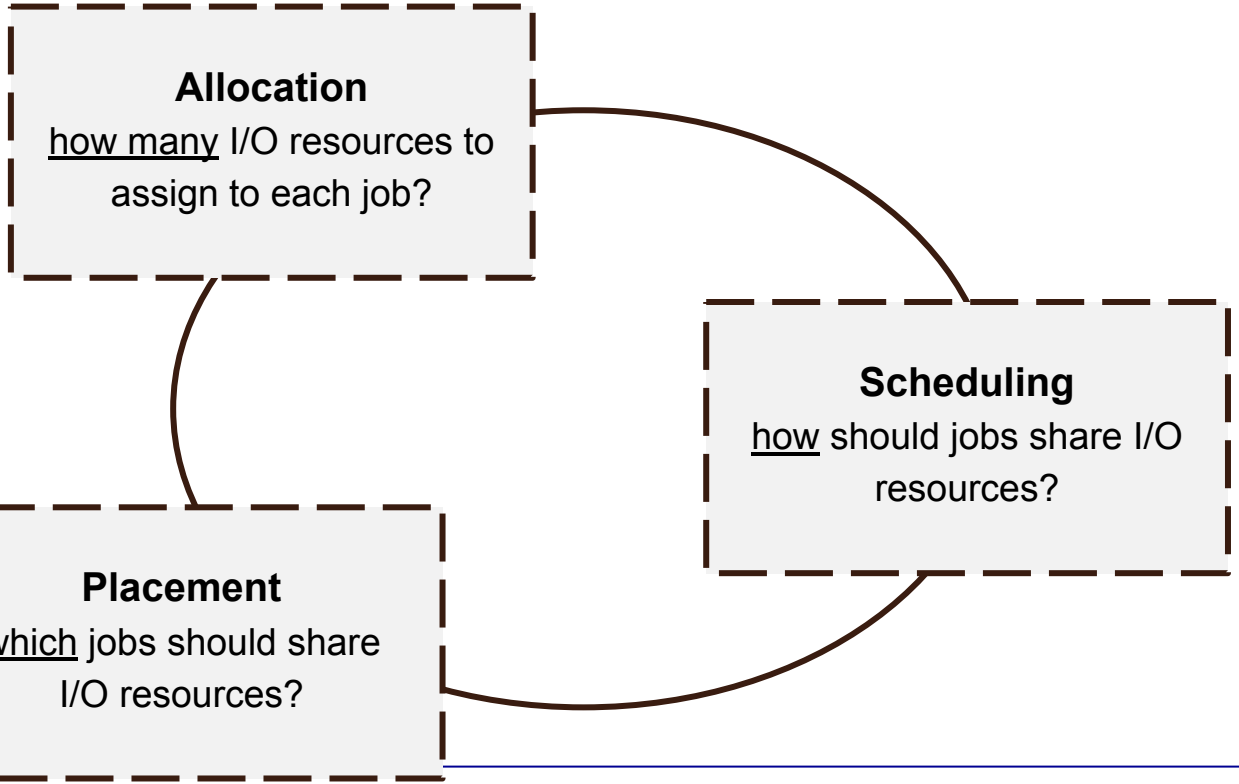
I/O resource management

- Systems use a default, “general-purpose” allocation
  - stripe count of 1, number of I/O nodes that depends on the number of compute nodes, etc.
- Not enough information to do better
  - of application characteristics
  - of their impact
- **Optimization opport**

We want:

- **to manage I/O resources**
- **to benefit ALL applications**
- **(by adapting to their characteristics)**
- **and mitigating contention**

I/O resource management



1. The I/O scheduling problem

how to share resources?

"IO-Sets … for I/O bandwidth management"

Francieli Boito, Guillaume Pallez, Luan Teylo, Nicolas Vidal (TPDS 2023)

ADMIRE
malleable data solutions for HPC

- I/O scheduling to mitigate interference
  - control all accesses to the shared I/O resource
  - decide what applications can do I/O and when
- Our focus: periodic I/O
  - common in checkpointing, numerical simulations, etc.

IO-Sets

A set-based method for I/O scheduling

- At the start of an I/O phase, the application is assigned to a set S_i
  - **according to its I/O periodicity**
  - each set S_i is assigned a priority p_i
- Only one application per set is allowed to do I/O
  - Exclusive access within each set
  - Sharing between sets
- The available bandwidth is shared among sets according to their priorities
  - most frequent I/O - shorter I/O phases - gets higher priority

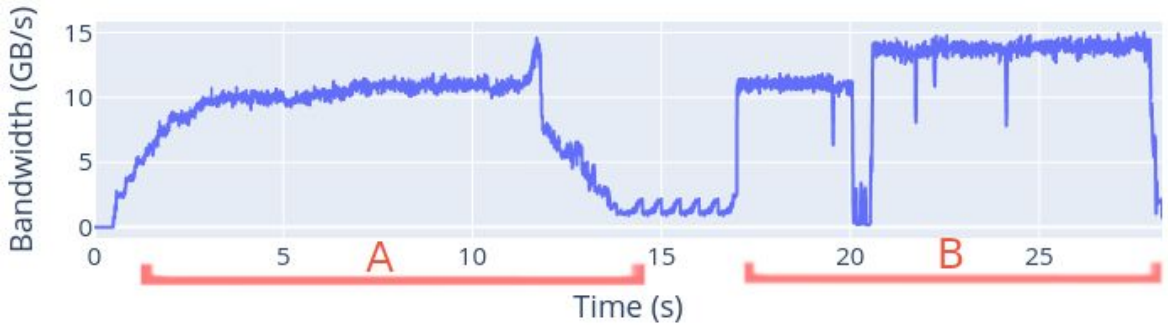
Capturing Periodic I/O Using Frequency Techniques

Ahmad Tarraf, Alexis Bandet, Francieli Boito, Guillaume Pallez, Felix Wolf (IPDPS 2024)

- A first step: **the time between the start of consecutive I/O phases**
- It is actually much harder than it sounds…
  - an I/O phase = multiple I/O requests
  - not all I/O is interesting

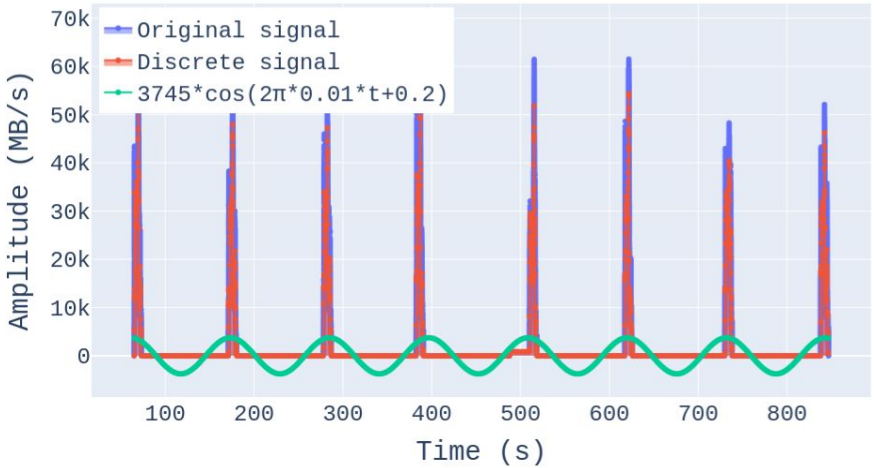
TECHNISCHE UNIVERSITÄT DARMSTADT

ADMIRE
malleable data solutions for HPC

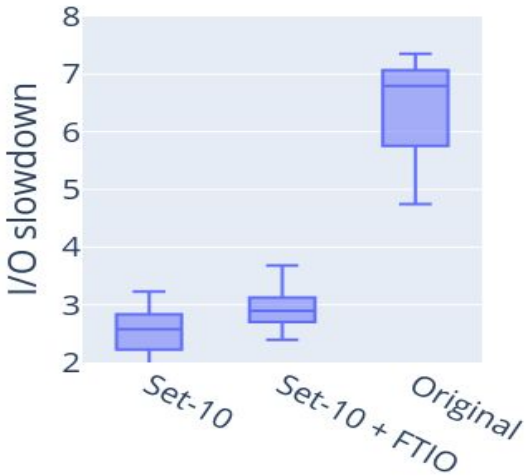


FTIO: frequency techniques for I/O

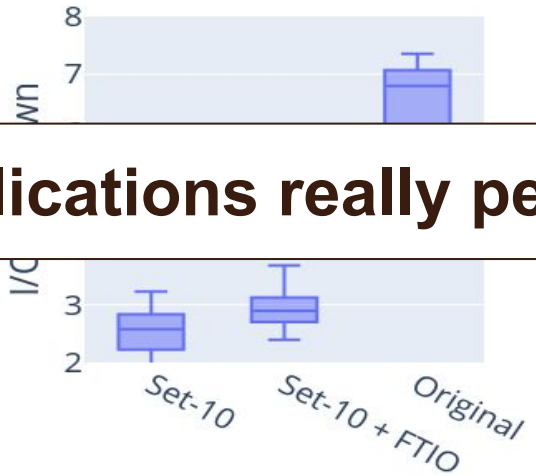
- Treat I/O bandwidth over time as a signal
  - Apply **discrete Fourier transform (DFT)** + outlier detection (e.g. z-score) to find the dominant frequency(ies)
- It can be done **online**, on a time window of recent activity
- Open source and freely available: https://github.com/tuda-parallel/FTIO



IOR on 9216 ranks



The lower, the better



But are applications really periodic?

The lower, the better

"A Deep Look Into the Temporal I/O Behavior…"

Francieli Boito, Luan Teylo, Mihail Popov, Théo Jolivel, François Tessier, Jakob Luettgau, Julien Monniot, Ahmad Tarraf, André Carneiro, Carla Osthoff (IPDPS 2025)

Machine	Year(s)	Scale	Country	PFS	#Traces
PlaFRIM	2022-2024	small	France	BeeGFS	16,474
SDumont	2020	medium	Brazil	Lustre	65,714
Intrepid	2013	large	USA	GPFS + PVFS	316,399
Blue Waters	2019	large	USA	Lustre	42,066

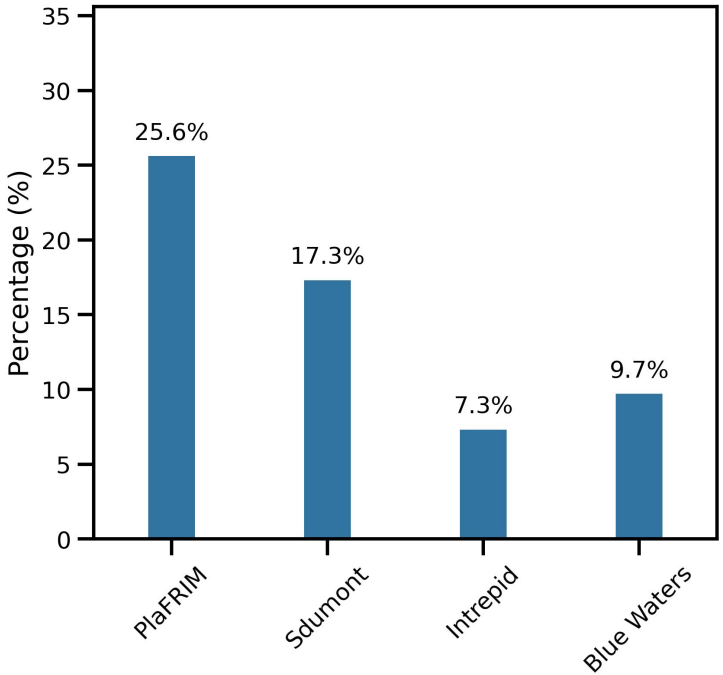
File system traces (time series)

Darshan traces

different trace granularities

Are applications periodic?

- Less than 26% of the jobs were found to be periodic
  - Partially due to trace granularity (lower for Darshan traces)
- But they account for over a third of write time and of written data
- Among the 10% largest and longest jobs, up to 50% are periodic
- No particular frequency (or range) is more common than the others

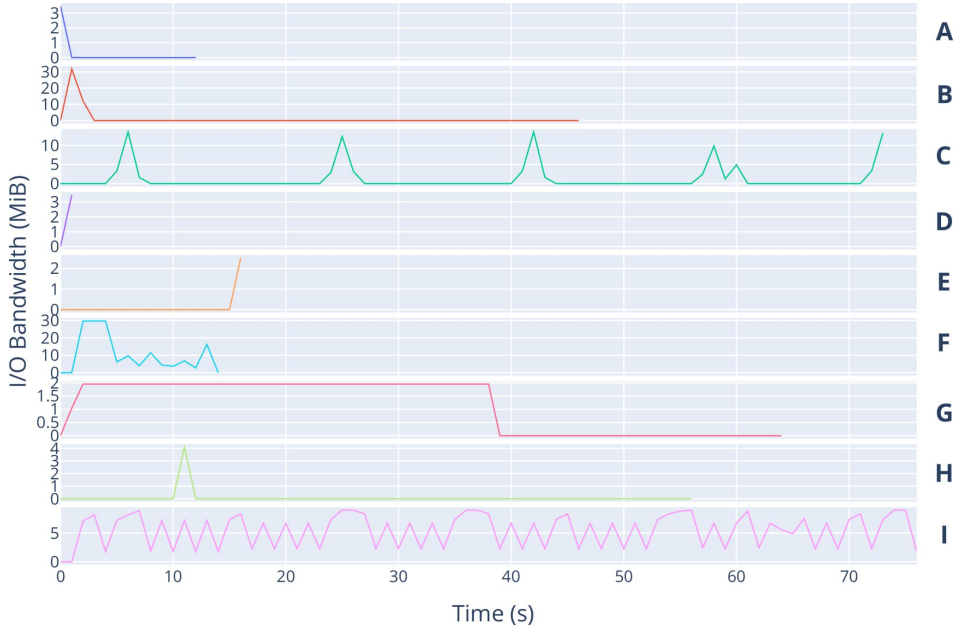


Short jobs were filtered out

Ongoing work: I/O scheduling in practice

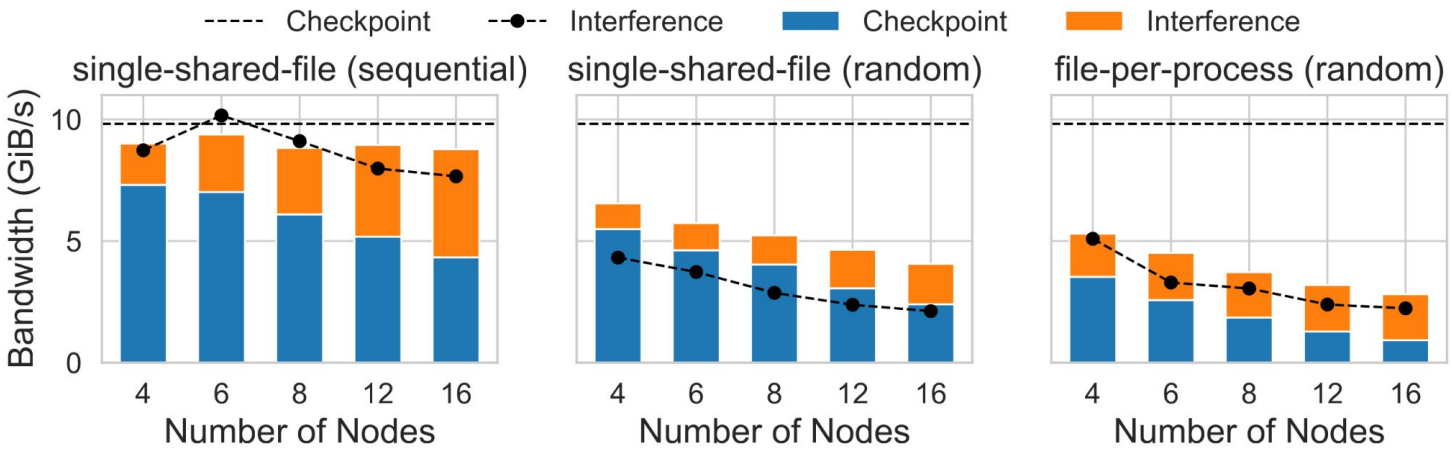
Classes of temporal I/O behavior

- Separated the jobs by duration and peak bandwidth
- K-means to cluster by behavior into each group, then manual labels
- Found only a small number of clusters, most jobs are in one of them



Ongoing work: I/O scheduling in practice

The plot thickens! Interference depends on application characteristics too!



[M. Trochon, J.-T. Acquaviva, F. Boito, B. Goglin, F. Tessier, L. Teylo, On the Impact of Interference from Concurrent Jobs on Checkpointing Performance, pre-print, 2025]

2. The allocation problem

how many I/O resources?

“Scheduling Distributed I/O Resources in HPC Systems”

Alexis Bandet, Francieli Boito, Guillaume Pallez (Euro-Par 2024)

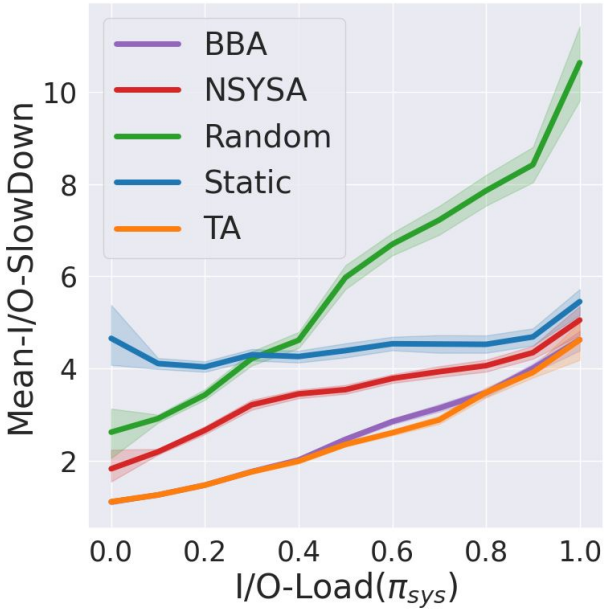
- **Allocation algorithms:**
  - Baselines: **Random**, **Static**, **MCKP** [Bez et al. 2021]
  - **BBA**: for each job, number of I/O resources to maximize its performance
  - **NSYSA**: for each job, number that minimizes I/O stress on the system

$$\texttt{I/O-Stress}(j,n) = \frac{n \cdot T\text{io}^{j}_{\text{cf}}(n)}{T^{j}_{\text{cpu}} + T\text{io}^{j}_{\text{cf}}(n)}$$

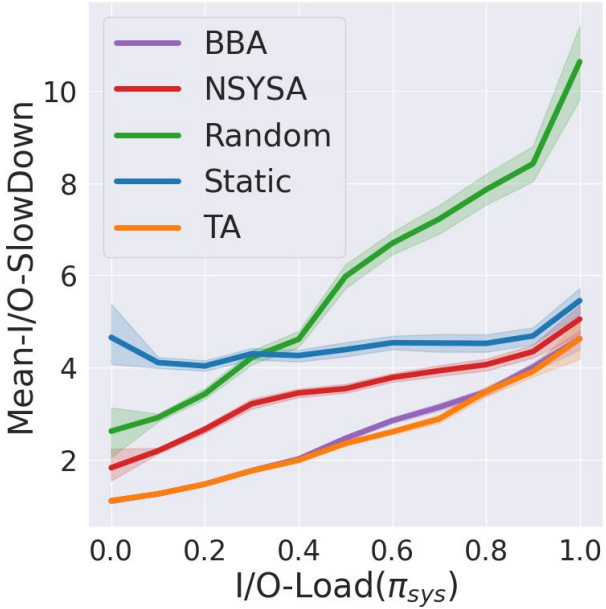
  - **TA**: start with NSYSA solution, greedily move towards BBA while

$$\sum_j \texttt{I/O-Stress}(j, \pi(j)) < 1$$

Results


BBA
NSYSA
Random
Static
TA
Mean-I/O-SlowDown
2
4
6
8
10
0.0
0.2
0.4
0.6
0.8
1.0
I/O-Load(π_{sys})

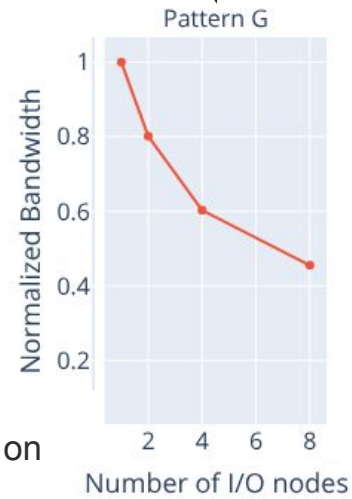
Results



- **MCKP** requires the b(n) function
- **NSYSA** and **TA** require the b(n) function and time spent on I/O
- **BBA requires the argmax b(n)**

- **When is the allocation done?**
  - once per execution (e.g. some I/O nodes) could rely on information from previous executions
  - at execution time (e.g. OSTs) - need to rely on partial, recent information

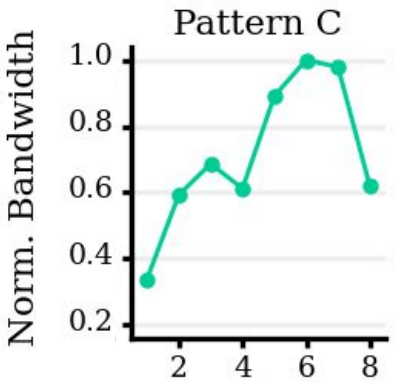
the b(n) function

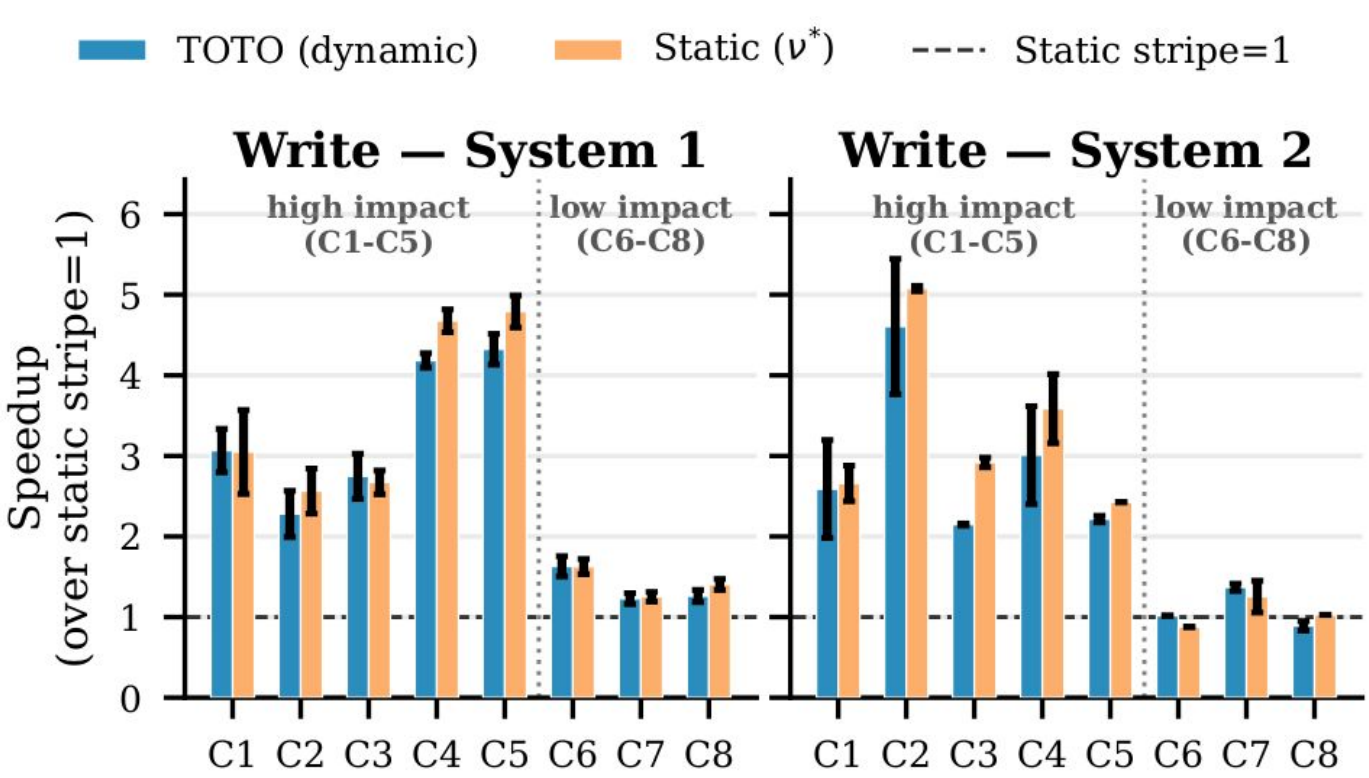


“TOTO: Transparent I/O Tuning for HPC Applications”

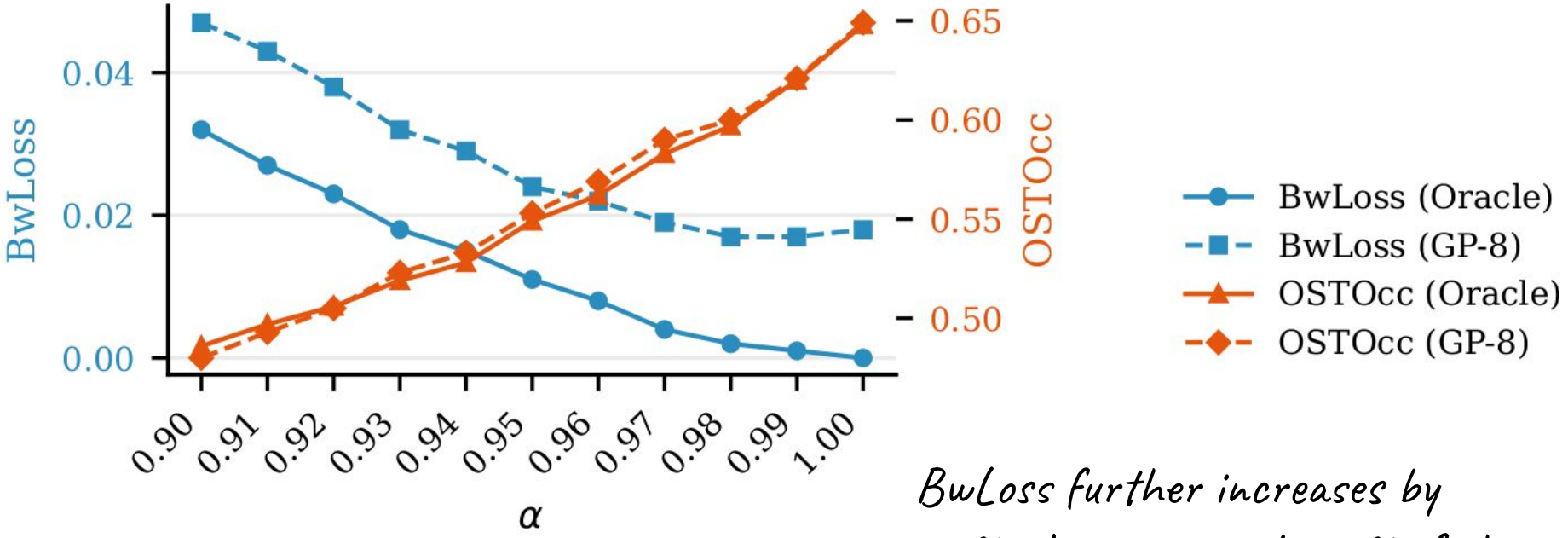
Francieli Boito, Luan Teylo, Mihail Popov, Laora Aimi, Alexis Bandet, Laércio Lima Pilla, Guillaume Pallez (under review, 2026)

- **New allocation algorithm: BBA***
  - The smallest number of I/O resources that reaches at least α of the best performance
- **Decision tree based models to find such number**
  - From request size, number of nodes and processes, number of files, spatiality
- **<u>Local</u> decisions (in the context of each application)**
  - May impose higher load on the system
  - Scheduling and placement problems!





Overhead < 8% for all cases



BwLoss further increases by ~4% when using only 10% of the data for training

5. Conclusion and Perspectives

Conclusions & Perspectives

- When sharing I/O resources, information about **periodicity** can help mitigate contention
  - but not all jobs are periodic! And application characteristics impact interference
  - Classifying applications according to their temporal behaviors
  - Ongoing work: improved I/O scheduling
- **Behavior-aware I/O resource allocation** can improve performance and utilization
  - Can be done with somewhat limited information about applications
  - TOTO: lightweight, low overhead, trained in a few hours, ~5x faster I/O

RÉPUBLIQUE FRANÇAISE
Liberté Égalité Fraternité

université de BORDEAUX

LaBRI

BORDEAUX INP

cea

cnrs

Inria

FRANCE 2030
PROGRAMME DE RECHERCHE
NUMÉRIQUE POUR L'EXASCALE

Towards I/O Resource Management for HPC

Francieli Boito

Associate Professor @ University of Bordeaux, France

Researcher @ LaBRI & Inria (TADaaM team)