Discovering Structure in Unstructured I/O

Jun He[1,2], John Bent[3], Aaron Torres[4],
Gary Grider[4], Garth Gibson[5],
Carlos Maltzahn[6], Xian-He Sun[1]

[1]Illinois Institute of Technology

[2]New Mexico Consortium

[3]EMC

[4]Los Alamos National Laboratory

[5]Carnegie Mellon University

[6]University of California Santa Cruz

November 12, 2012

New Mexico Consortium

NM State University / The University of New Mexico

Illinois Institute of Technology

EMC²

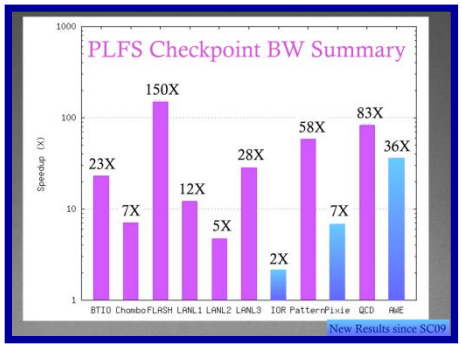
Los Alamos National Laboratory, Est. 1943

Carnegie Mellon

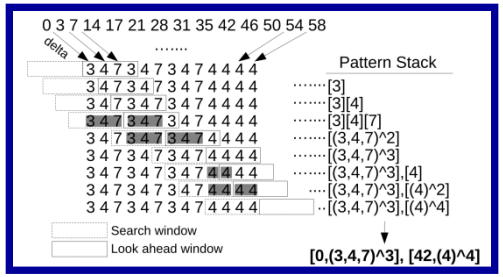
University of California Santa Cruz

Outline

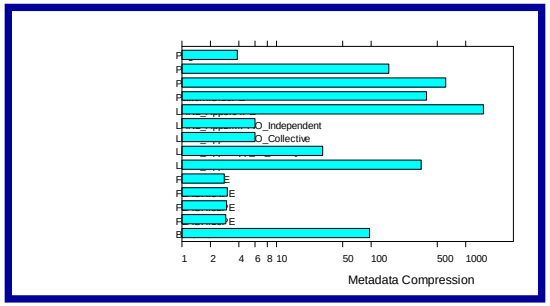
This presentation focuses on recognizing I/O patterns and representing them compactly.



PLFS (Parallel Log-structured File System) accelerates checkpointing significantly, but its internal metadata may grow too big.



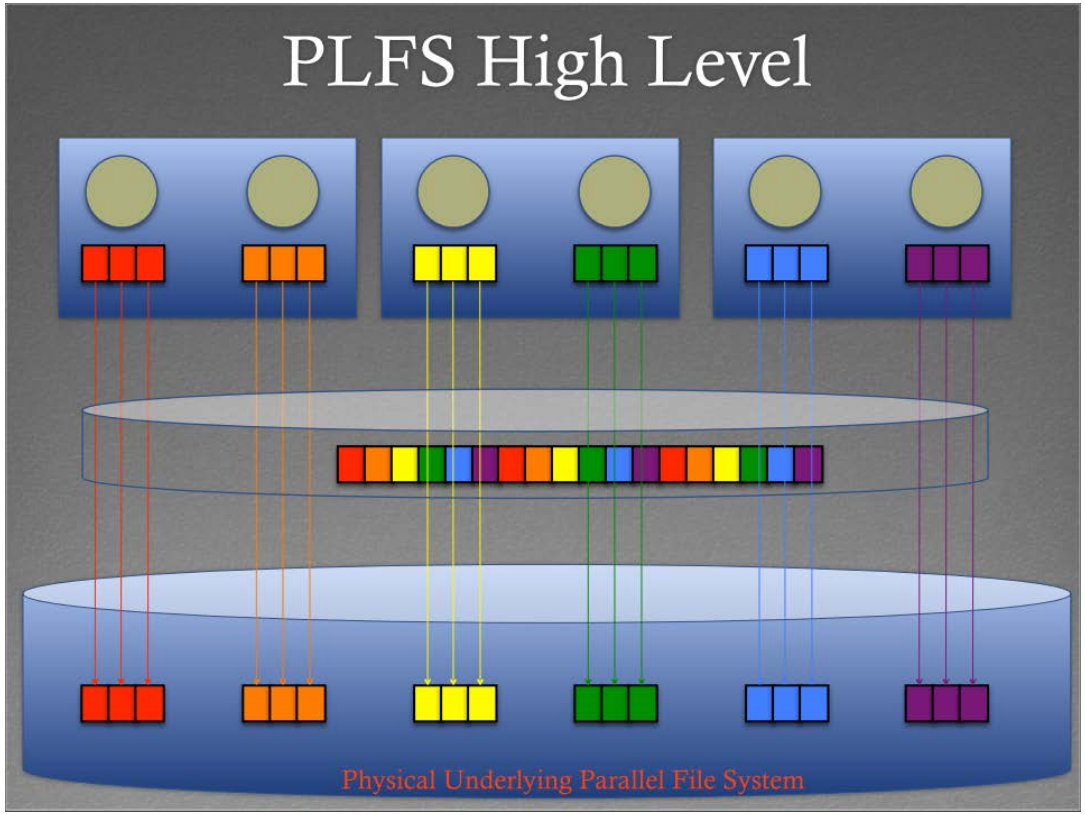
How to recognize I/O patterns and reduce PLFS metadata size.



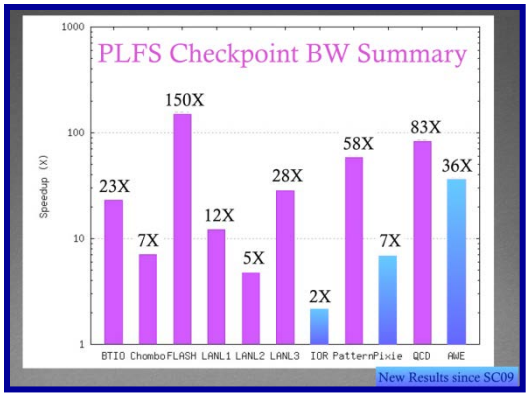
Metadata size is reduced significantly and R/W performance is improved.

Motivation

Checkpointing is the storage driver in supercomputers. PLFS can improve checkpointing significantly.

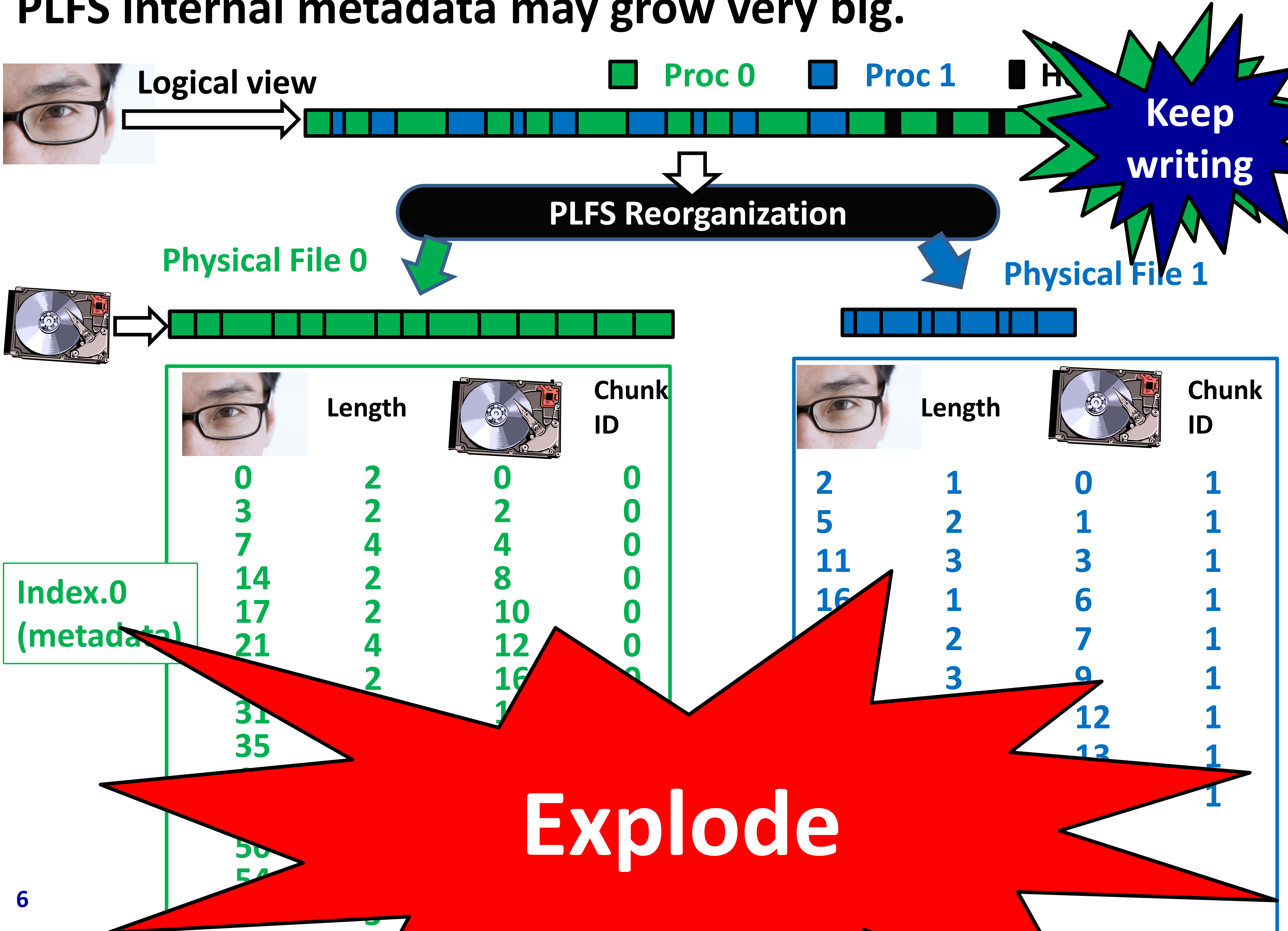


PLFS transparently transforms N-1 write to N-N write.

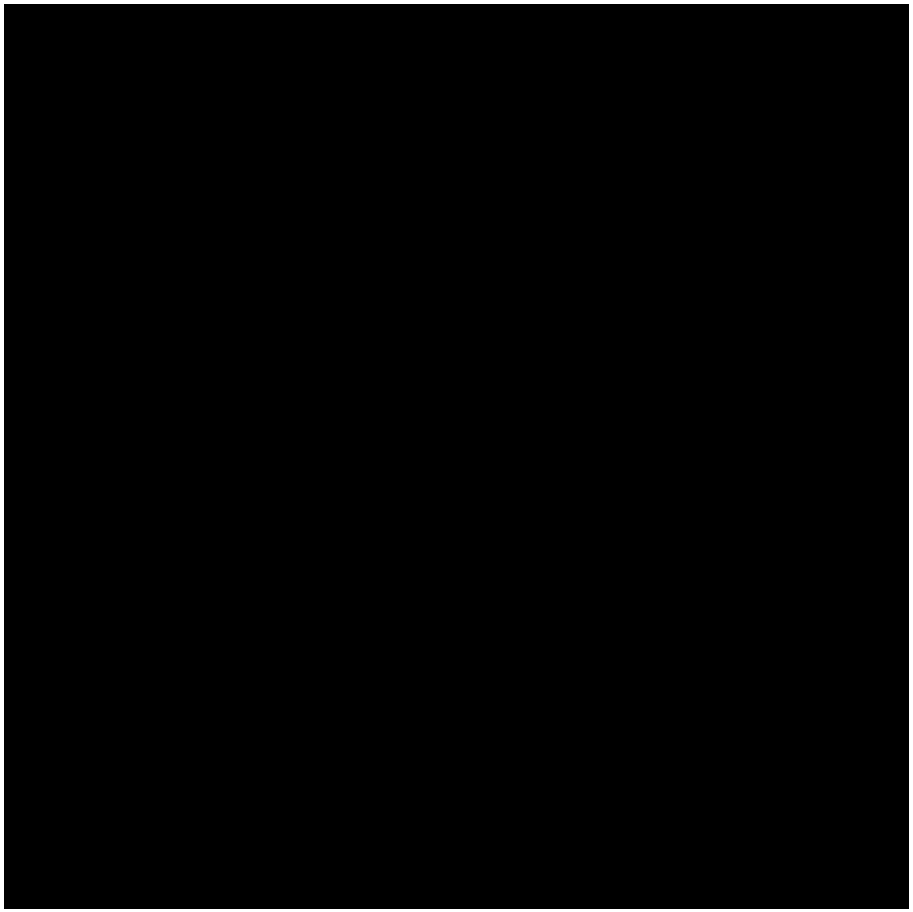


Up to several orders of magnitude improvement.

PLFS internal metadata may grow very big.



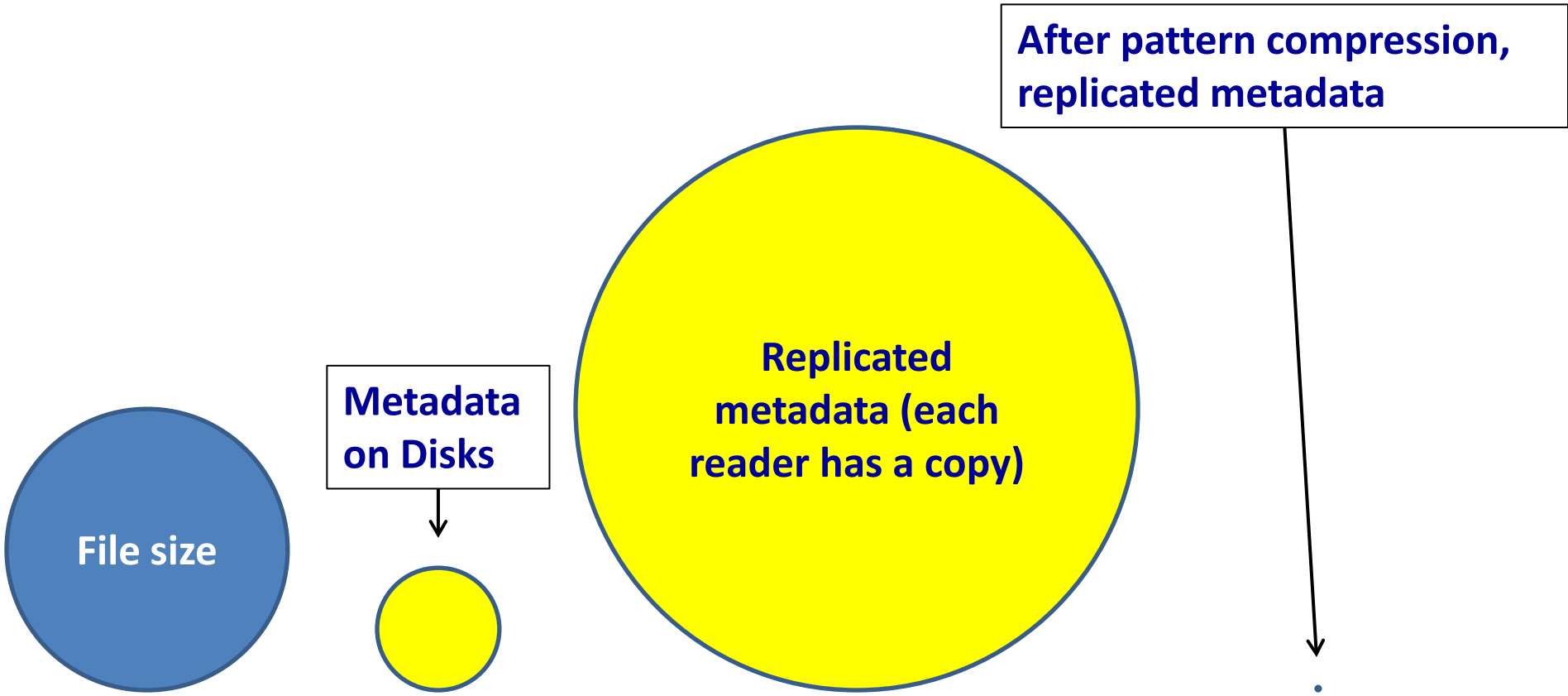
Applications' I/O has patterns and they can be represented compactly.



Pattern of LANL anonymous 3.

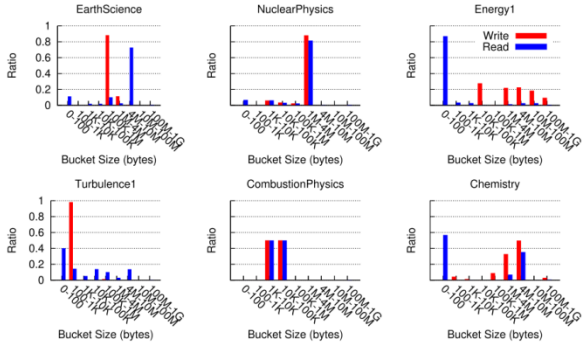
Colors indicate ranks.

Metadata of LANL anonymous 3 is big.



Related Work

Coarse-granularity patterns are not precise enough. Statistics methods are lossy.



From 1. Thanks to Phil Carns.

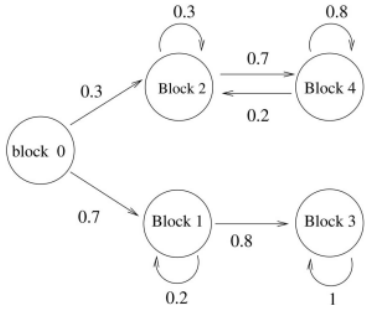


Fig. 1. Markov model file block example.

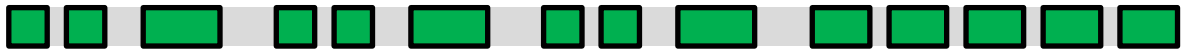
From 7.

1. **(DARSHAN) P. Carns, K. Harms, W. Allcock, C. Bacon, S. Lang, R. Latham, and R. Ross, "Understanding and improving computational science storage access through continuous characterization," ACM Transactions on Storage (TOS), vol. 7, no. 3, p. 8, 2011.**
2. B. Pasquale and G. Polyzos, "A static analysis of i/o characteristics of scientific applications in a production workload," in Proceedings of the 1993 ACM/IEEE conference on Supercomputing. ACM, 1993, pp. 388–397.
3. E. Smirni and D. Reed, "Lessons from characterizing the input/output behavior of parallel scientific applications," Performance Evaluation, vol. 33, no. 1, pp. 27–44, 1998.
4. S. Byna, Y. Chen, X. Sun, R. Thakur, and W. Gropp, "Parallel I/O prefetching using MPI file caching and I/O signatures," in Proceedings of the 2008 ACM/IEEE conference on Supercomputing. IEEE Press, 2008, p. 44.
5. J. He, H. Song, X. Sun, Y. Yin, and R. Thakur, "Pattern-aware file reorganization in mpi-io," in Proceedings of the sixth workshop on Parallel Data Storage. ACM, 2011, pp. 43–48.
6. T. Madhyastha and D. Reed, "Learning to classify parallel input/output access patterns," Parallel and Distributed Systems, IEEE Transactions on, vol. 13, no. 8, pp. 802–813, 2002.
7. **J. Oly and D. Reed, "Markov model prediction of i/o requests for scientific applications," in Proceedings of the 16th international conference on Supercomputing. ACM, 2002, pp. 147–155.**
8. N. Tran and D. Reed, "Automatic time series modeling for adaptive i/o prefetching," Parallel and Distributed Systems, IEEE Transactions on, vol. 15, no. 4, pp. 362–377, 2004.

Methods

Sliding window algorithm is effective in discovering pattern.

Logical file:



Logical offsets: 0 3 7 14 17 21 28 31 35 42 46 50 54 58

stride list: 3 4 7 3 4 7 3 4 7 4 4 4 4

Results

Patterns of real applications are explored, as well as benchmarks.

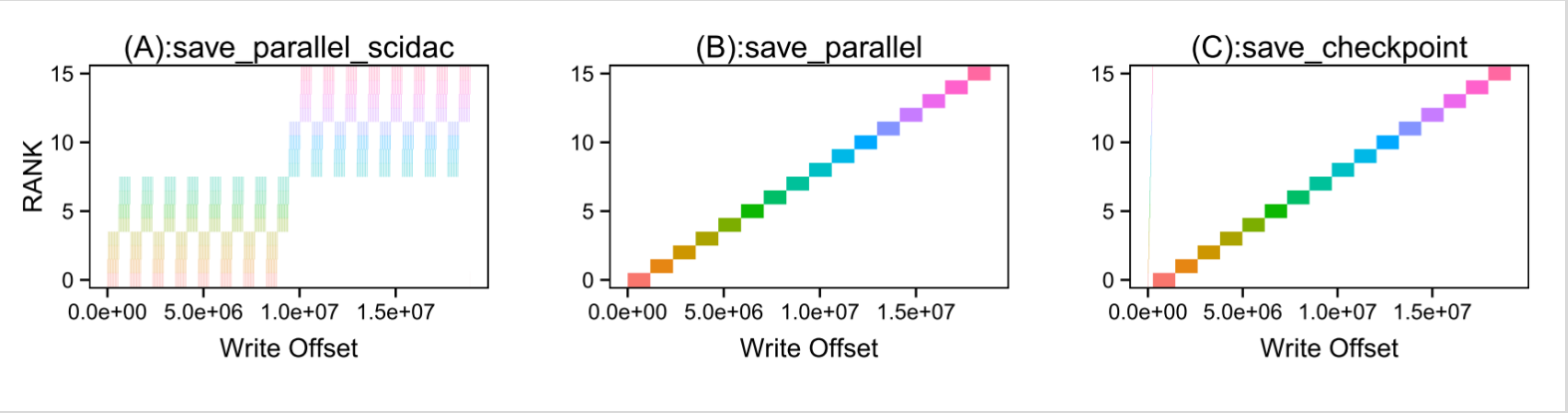
Applications explored:

- **LIVE RUN:**
  - **Pagoda (PNNL), MPI-Blast, MILC, Montage (NASA), ADIOS (ORNL), MADBench2 (LBL)**
- **TRACE REPLAY:**
  - **Alegra (SNL), S3D (SNL), LANL anonymous applications, FLASH, BTIO**

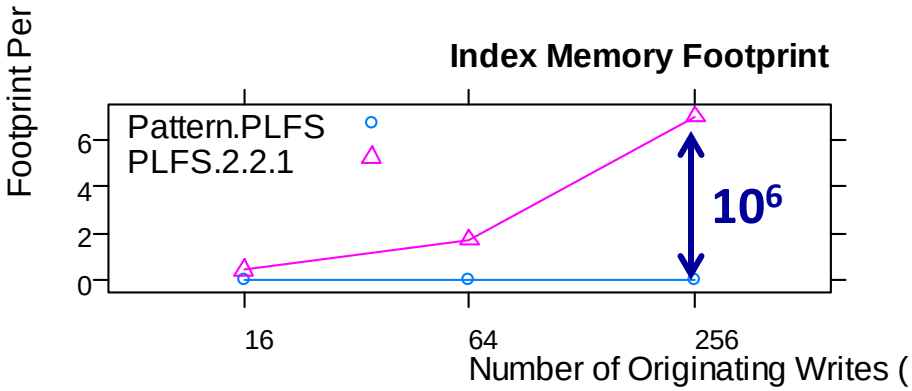
Benchmarks explored :

- **PATTERN-IO (NERSC), MPI-TILE-IO (ANL), FS-TEST (LANL)**

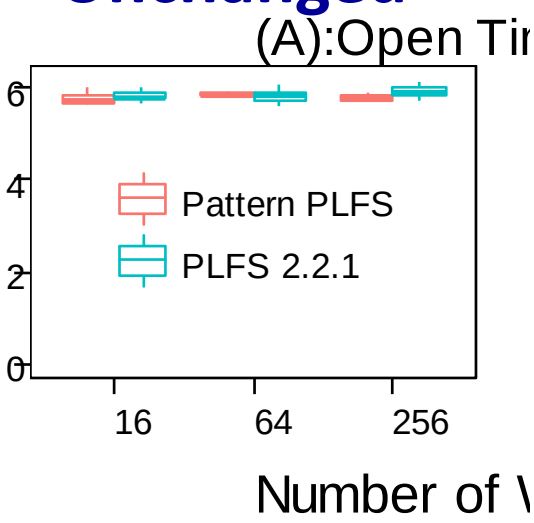
Example: write patterns of MILC (physics app).
In-memory index compression rates by Pattern PLFS (higher is better):
(A):37.0; (B):3.0;(C):3.6

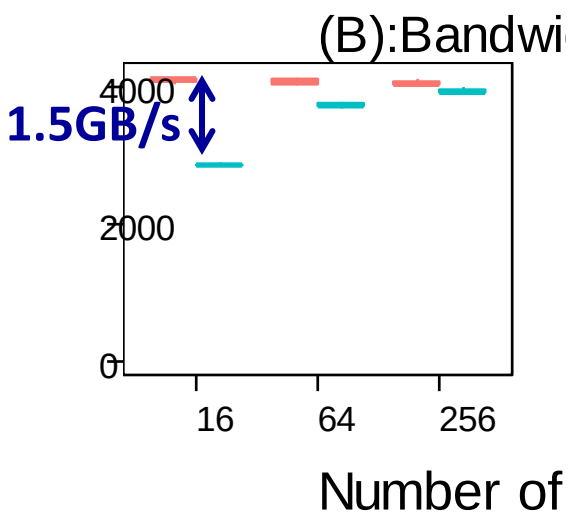


Write Performance Improvement

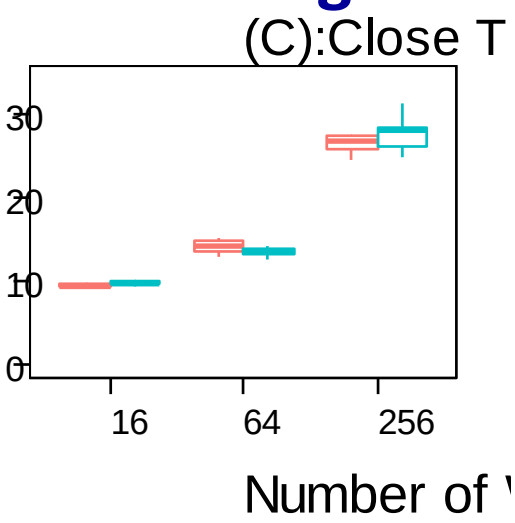


Unchanged



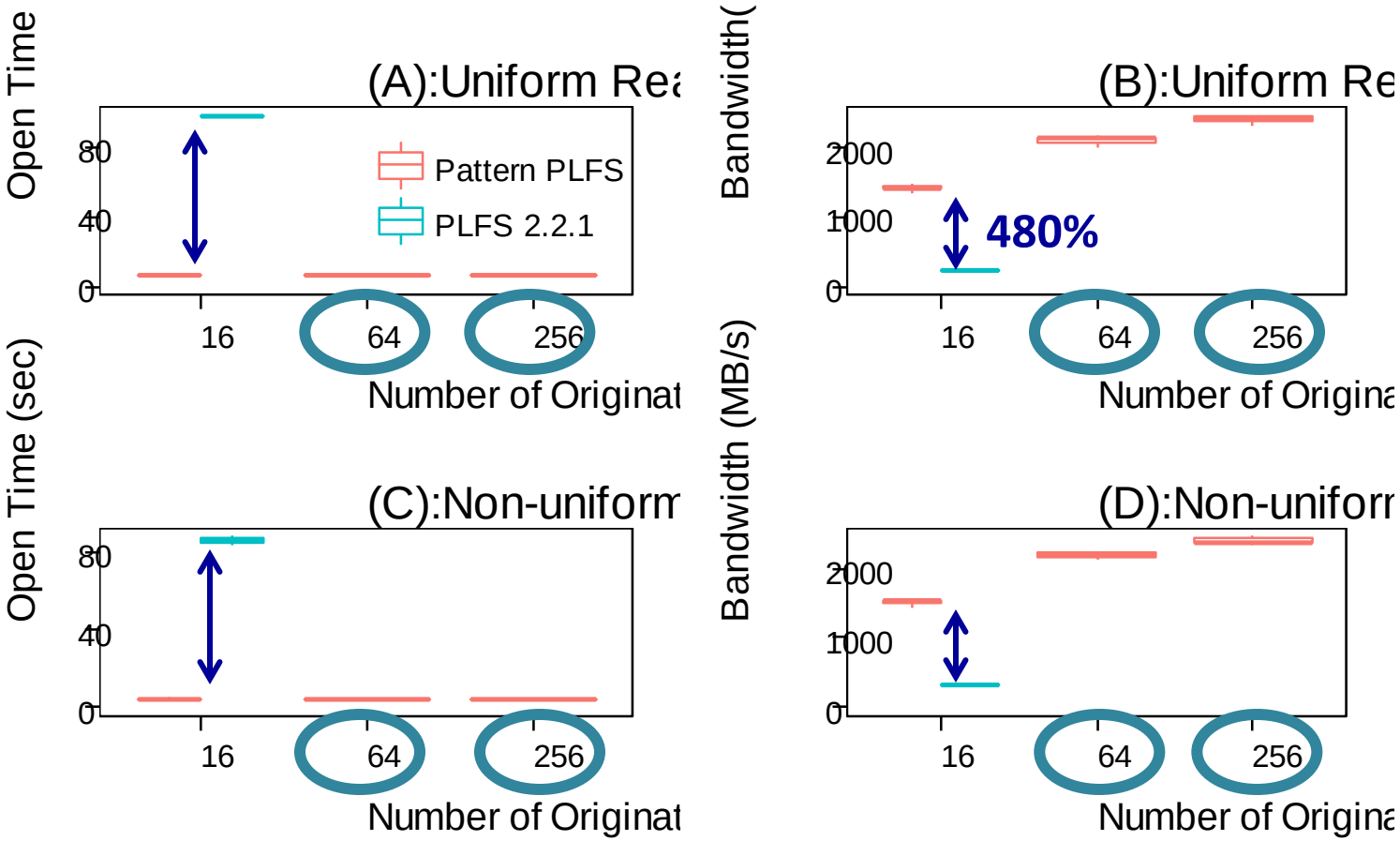


Unchanged



512 processes with write size of 4K.

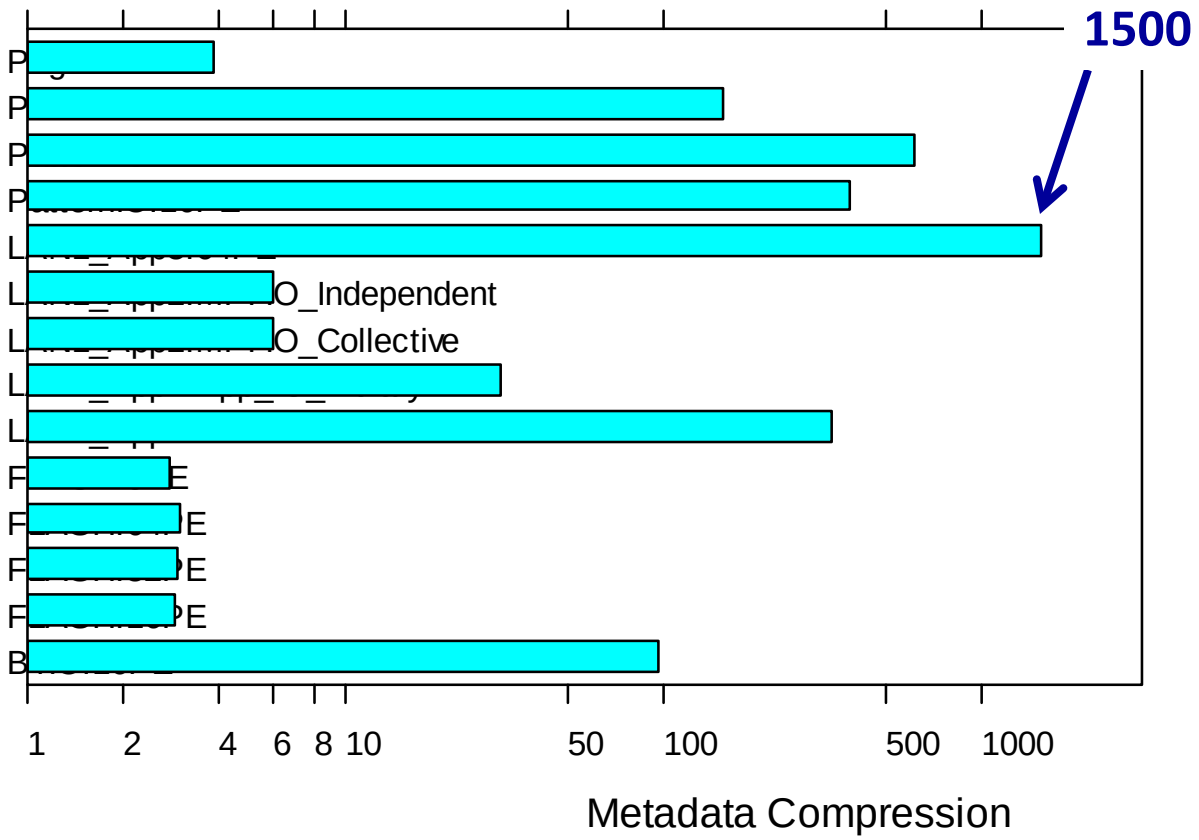
Read Performance Improvement



Uniform read: 512 processes

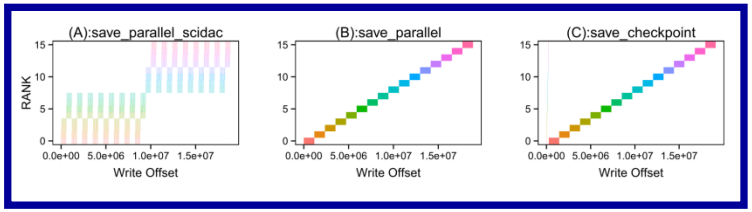
Non-uniform read: 256 processes

PLFS metadata can be reduced by up to several orders of magnitude.

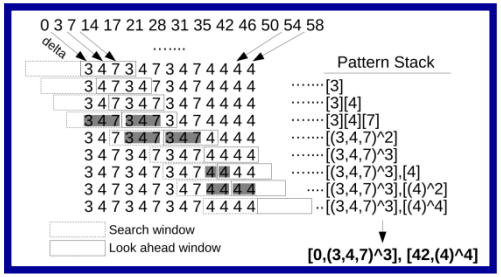


Conclusions & Future Work

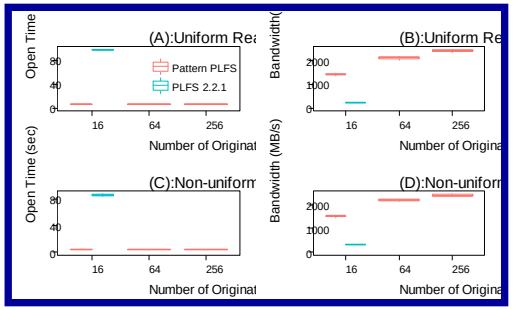
The proposed sliding window algorithm is effective on discovering structure and improving I/O performance.



Application patterns are studied.



I/O structure discovering algorithm and a compact structure representation are proposed.



Metadata is reduced and I/O performance is improved.

The proposed techniques have the potential for being applied in other systems.

Predictability & Compactness

Pre-fetching

Block pre-allocation

Data layout optimization

SciHadoop metadata compression

Acknowledgement

- Michael Lang (Los Alamos National Laboratory)
- Adam Manzanares (California State University)
- All the reviewers

This work was performed at the Ultrascale Systems Research Center (USRC) at Los Alamos National Laboratory, supported by the U.S. Department of Energy DE-FC02-06ER25750. The publication has been assigned the LANL identifier LA-UR-12-25954.

Q & A

Jun's email:
junnhe@gmail.com