IOPin: Runtime Profiling of Parallel I/O in HPC Systems

Seong Jo (Shawn) Kim[*], Seung Woo Son[+], Wei-keng Liao[+], Mahmut Kandemir*, Rajeev Thakur[#], and Alok Choudhary[+]

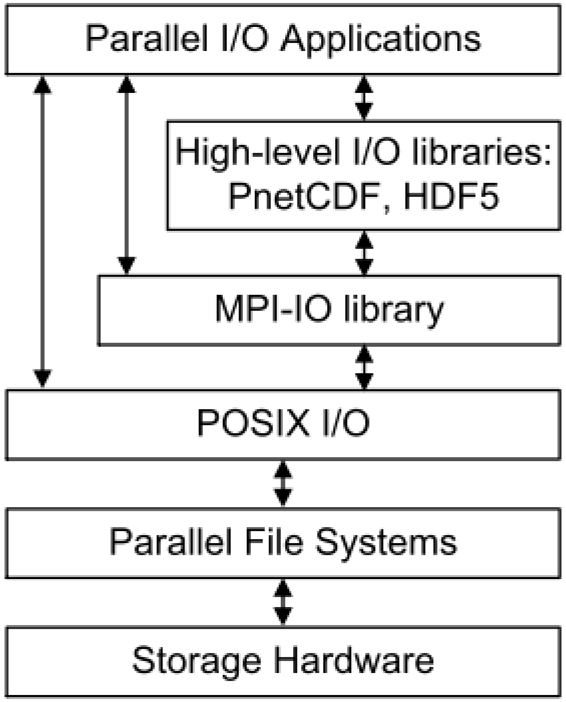
[*]: Pennsylvania State University
[+]: Northwestern University
[#]: Argonne National Laboratory

Outline

- Motivation
- Overview
- Background: IOPin
- Technical Details
- Evaluation
- Conclusion & Future Work

Motivation

- Users of HPC systems frequently find that limiting the perfor-mance of the applications is the storage system, not the CPU, memory, or network.
- I/O behavior is the key factor to determine the overall performance.
- Many I/O-intensive scientific applications use parallel I/O software stack to access files in high performance.
- Critically important is understanding how the parallel I/O system operates and the issues involved.
- Understand I/O behavior!!!



Motivation (cont'd)

- Manual instrumentation for understanding I/O behavior is extremely difficult and error-prone.
- Most parallel scientific applications are expected to run on large-scale systems with 100,000↑ processors to achieve better resolution.
- Collecting and analyzing the trace data from them is challenging and burdensome.

Our Approach

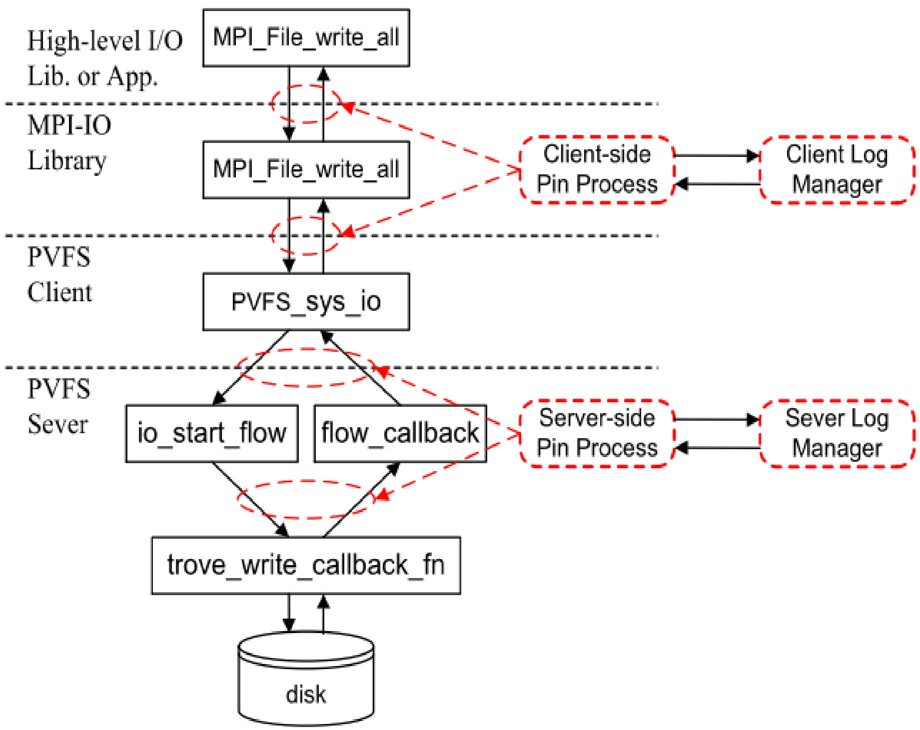
- IOPin – Dynamic performance and visualization tool
- We leverage a light-weight binary instrumentation using probe mode in Pin.
  - Language independent instrumentation for scientific applications written in C/C++ and Fortran
  - Neither source code modification nor recompilation of the application and the I/O stack components
- IOPin provides a hierarchical view for parallel I/O:
  - Associating MPI I/O call issued from the application with its sub-calls in the PVFS layer below
- It provides detailed I/O performance metrics for each I/O call: I/O latency at each layer, # of disk accesses, disk throughput
- Low overhead: ~ 7%

Background: Pin

- Pin is a software system that performs runtime binary instrumentation.
- Pin supports two modes of instrumentation, JIT mode and probe mode.
- JIT mode uses a just-in-time compiler to recompile the program code and insert instrumentation; while probe mode uses code trampolines (jump) for instrumentation.
- In JIT mode, the incurred overhead ranges from 38.7% to 78% of the total execution time with 32, 64, 128, and 256 processes.
- In probe mode, about 7%.

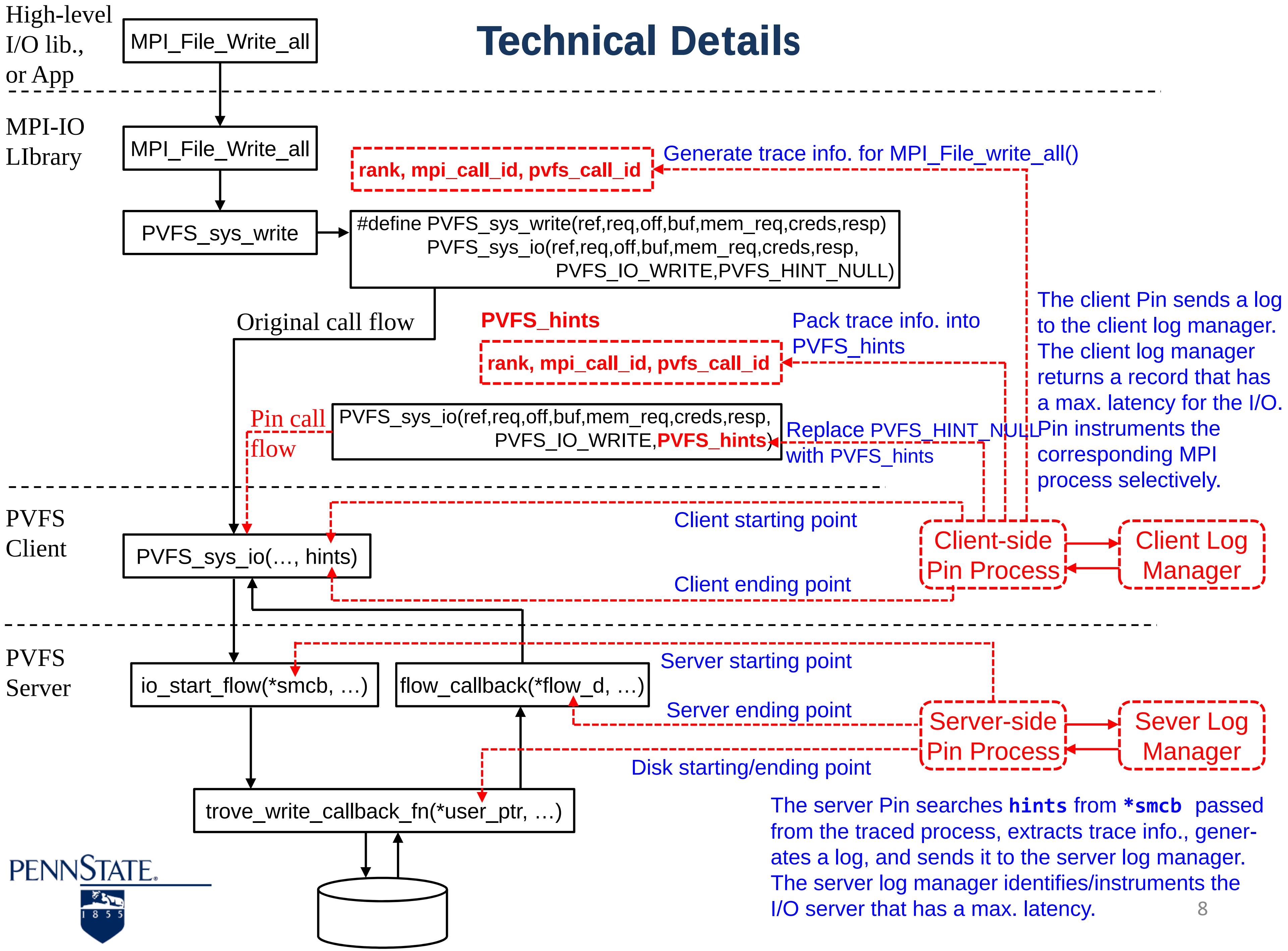
Overview: IOPin

- The pin process on the client creates two trace log info. for the MPI library and PVFS client.
  - rank, mpi_call_id, pvfs_call_id, I/O type (write/read), latency
- The pin process on the server produces a trace log info. with server_id, latency, processed bytes, # of disk accesses, and disk throughput.
- Each log info is sent to the log manager and the log manager identifies the process that has a max. latency.
- Pin process instruments the target process.

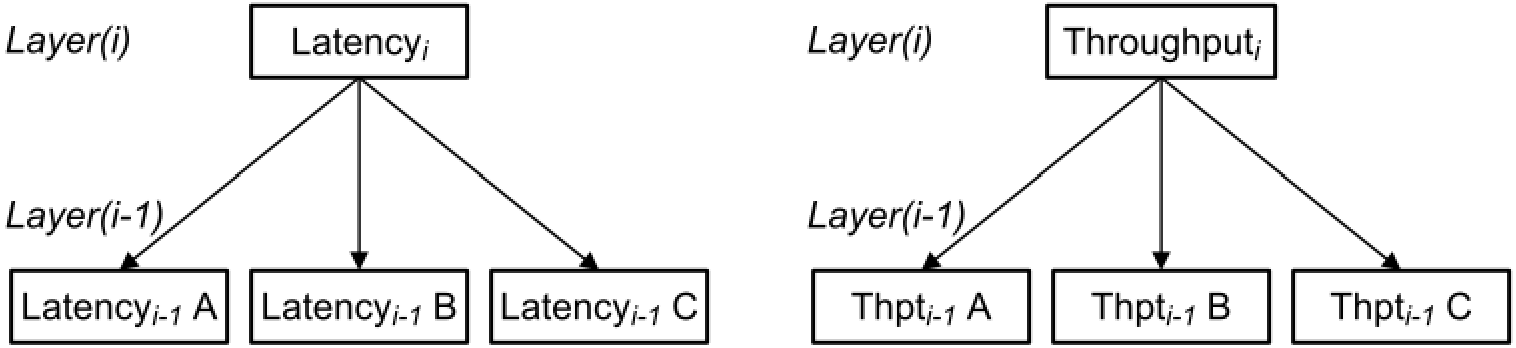


PENNSTATE
1855

Technical Details



Computation Methodology: Latency and Throughput



- For each I/O operation:
  - the I/O latency computed at each layer is the *maximum* of the I/O latencies from the layers below.
  - I/O throughput computed at any layer is the *sum* of the I/O throughput from the layers below

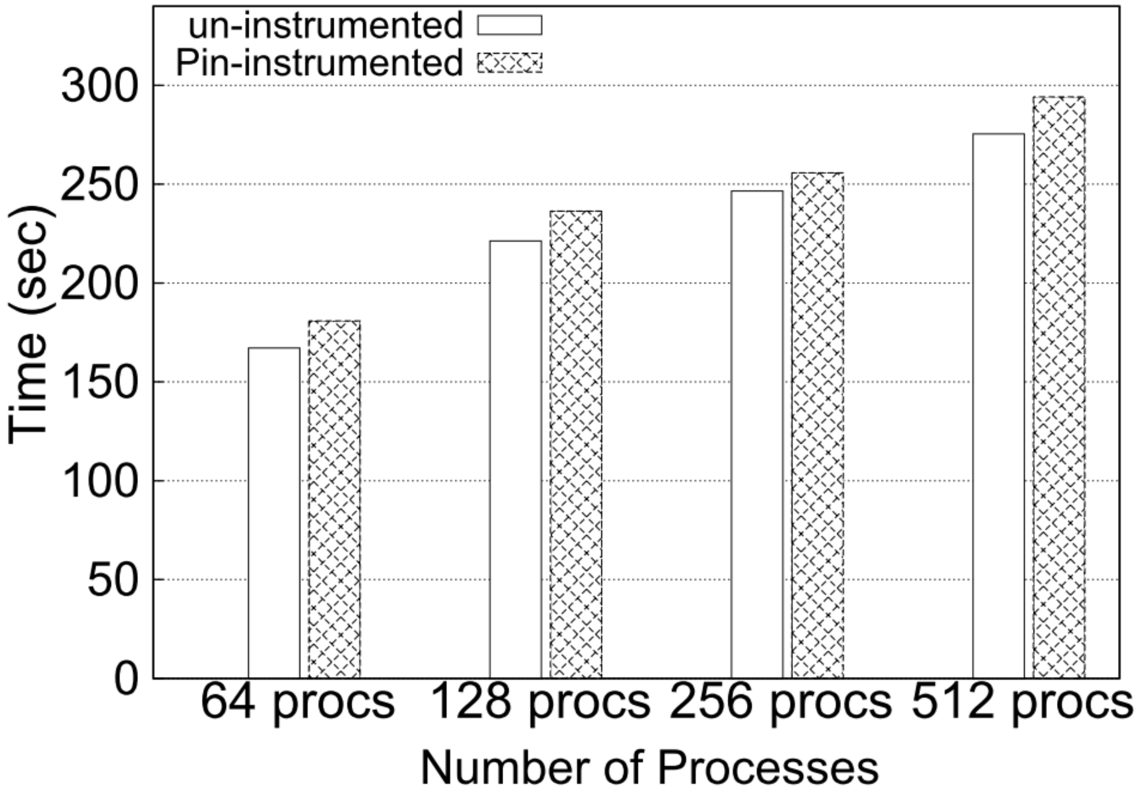
Evaluation

- Hardware:
  - Breadboard cluster at Argonne National Laboratory
  - 8 quad-core processors per node: support 32 MPI processes
  - 16 GB main memory
- I/O stack configuration:
  - Application: S3D I/O
  - PnetCDF (pnetcdf-1.2.0), mpich2-1.4, pvfs-2.8.2
- PVFS configuration:
  - 1 metadata server
  - 8 I/O servers
  - 256 MPI processes

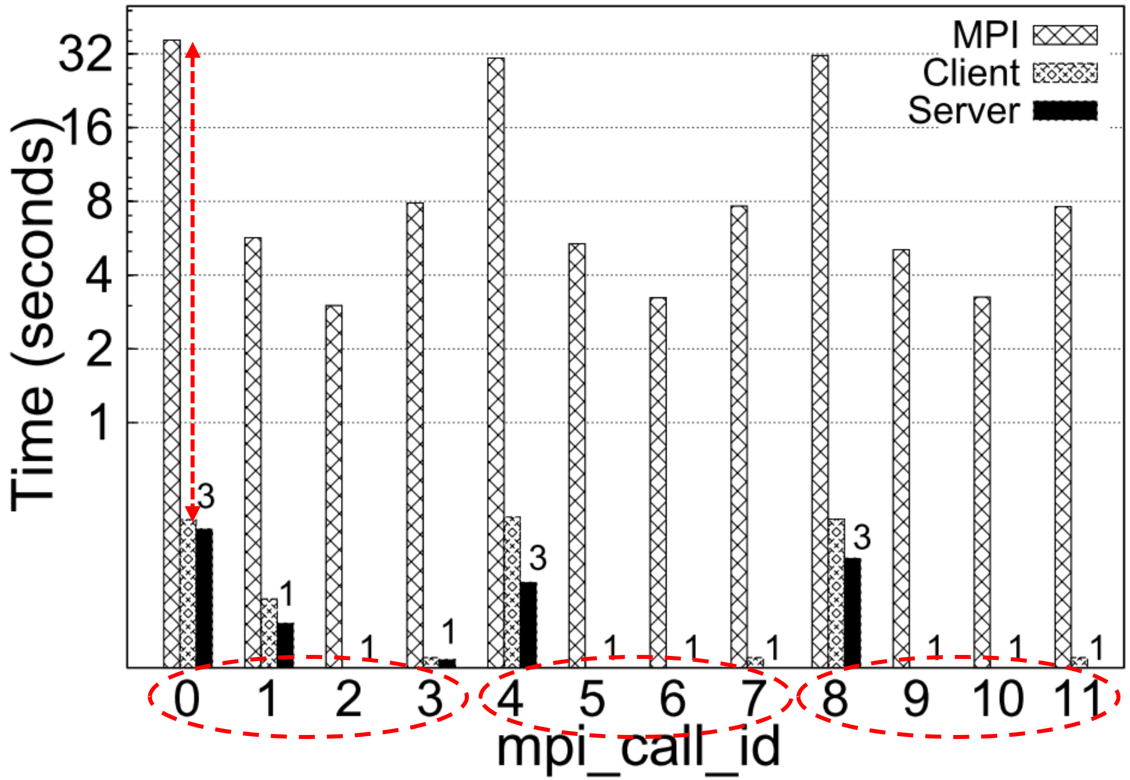
Evaluation:
S3D-IO

- S3D-IO
  - I/O kernel of S3D application
  - A parallel turbulent combustion application using a direct numerical simulation solver developed in SNL
- A checkpoint is performed at regular intervals.
  - At each checkpoint, four global arrays—representing the variables of mass, velocity, pressure, and temperature—are written to files.
- We maintain the block size of the partitioned X-Y-Z dimension as 200 * 200 * 200
- It generates three checkpoint files, 976.6MB each.

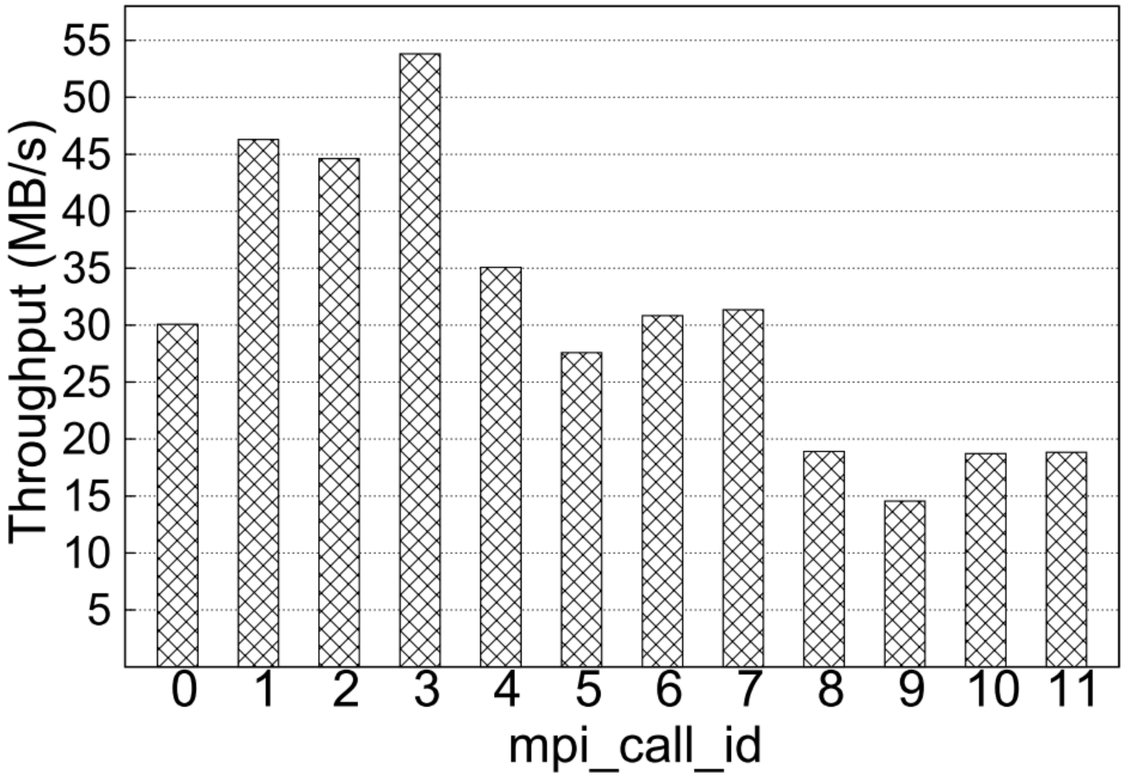
Evaluation:
Comparison of S3D I/O Execution Time



Evaluation:
Detailed Execution Time of S3D I/O



Evaluation:
I/O Throughput of S3D I/O



PENNSTATE
1855

Conclusion & Future Work

- Understanding I/O behavior is one of the most important steps for efficient execution of parallel scientific applications.
- IOPin provides dynamic instrumentation to understand I/O behavior without affecting the performance:
  - no source code modification and recompilation
  - a hierarchical view of the I/O call from the MPI lib. to the PVFS server
  - metrics: latency of each layer, # of fragmented I/O calls, # of disk accesses, I/O throughput
  - ~7% overhead
- Work is underway: (1) to test IOPin on a very large process counts, (2) to employ it for runtime I/O optimizations.

Questions?