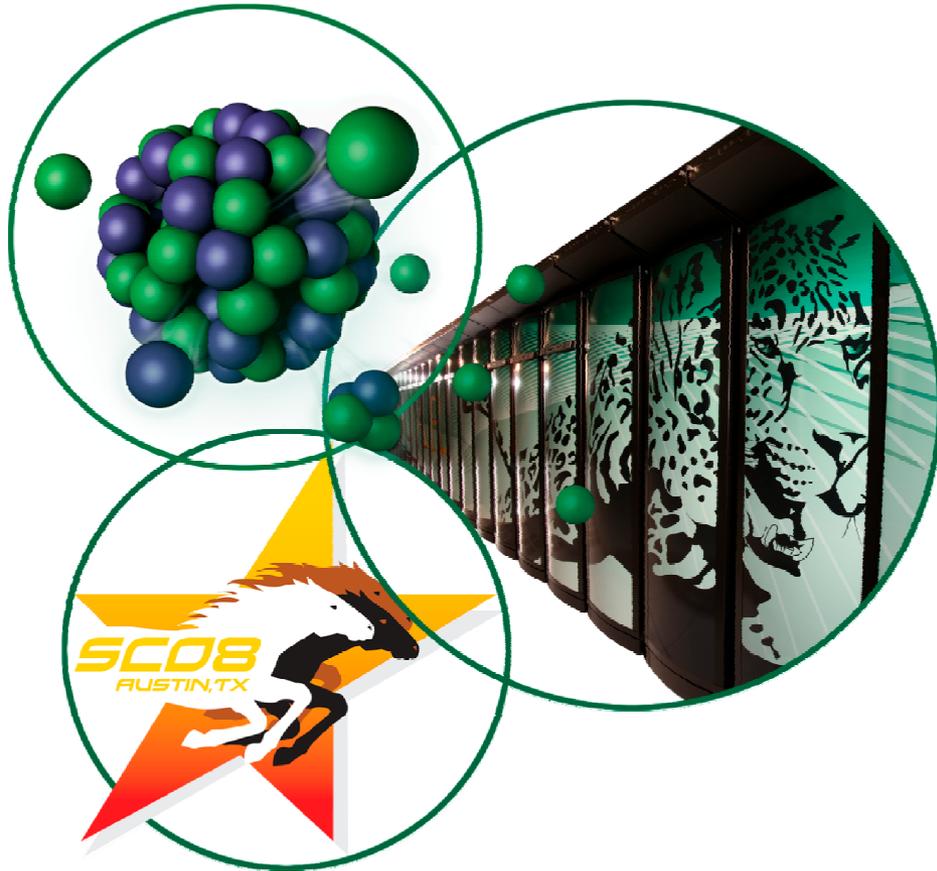


Understanding and Optimizing Data Input/Output of Large-Scale Scientific Applications

Presented by

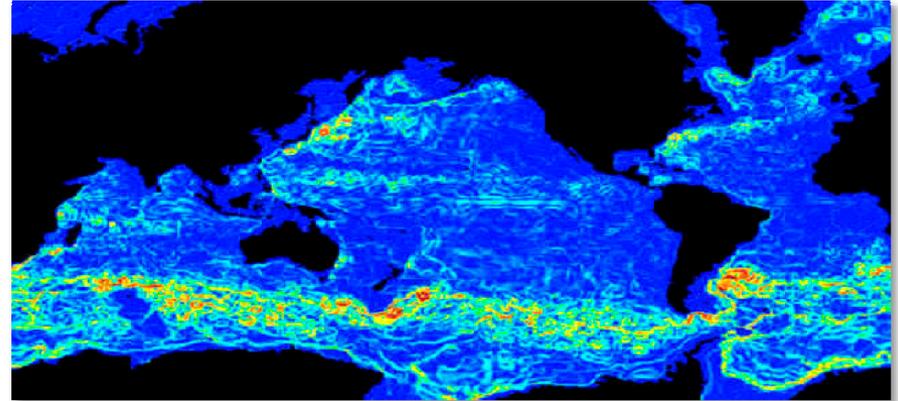
Jeffrey S. Vetter (Leader)
Weikuan Yu
Philip C. Roth

Future Technologies Group
Computer Science and Mathematics Division

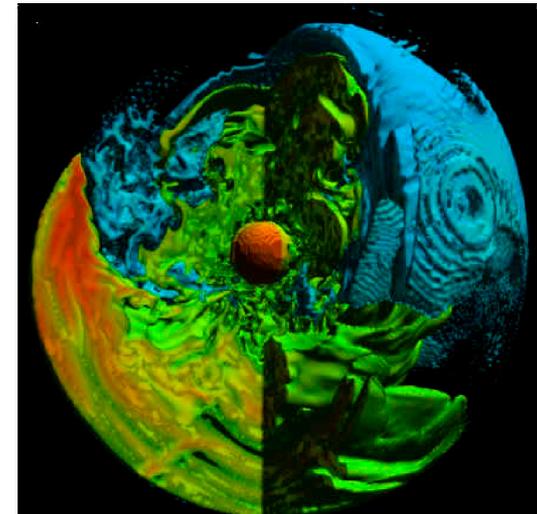
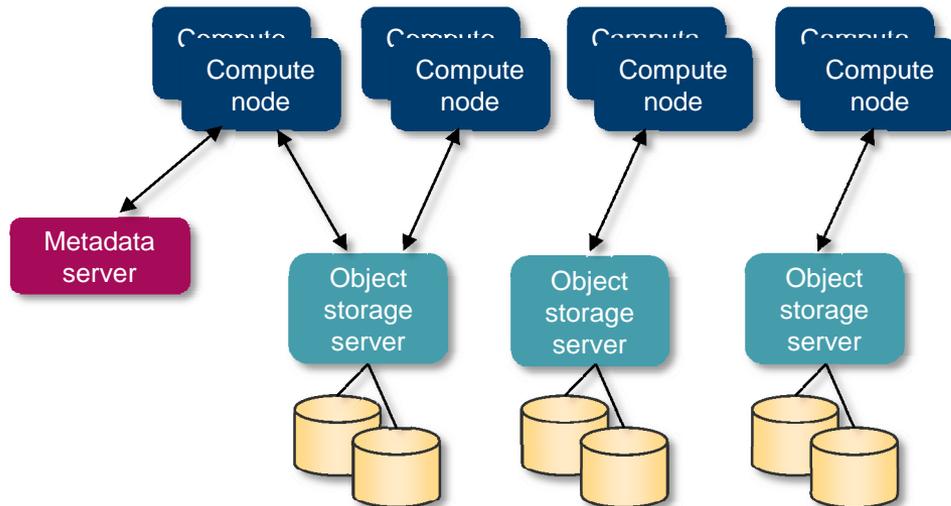


I/O for large-scale scientific computing

- Reading input and restart files
- Writing checkpoint files
- Writing movie, history files
- Gaps of understanding across domains; efficiency is low

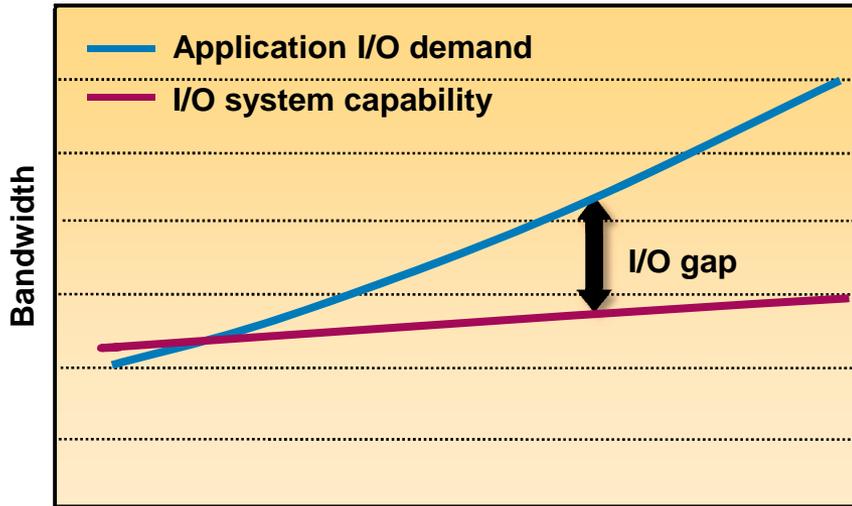


SciDAC climate studies visualization at ORNL

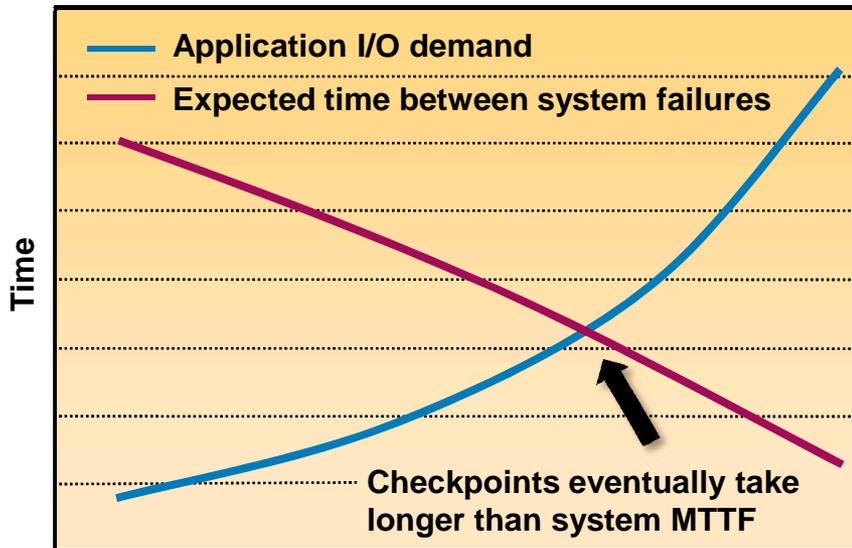


SciDAC astrophysics simulation visualization at ORNL

The I/O gap



Widening gap between application I/O demands and system I/O capability

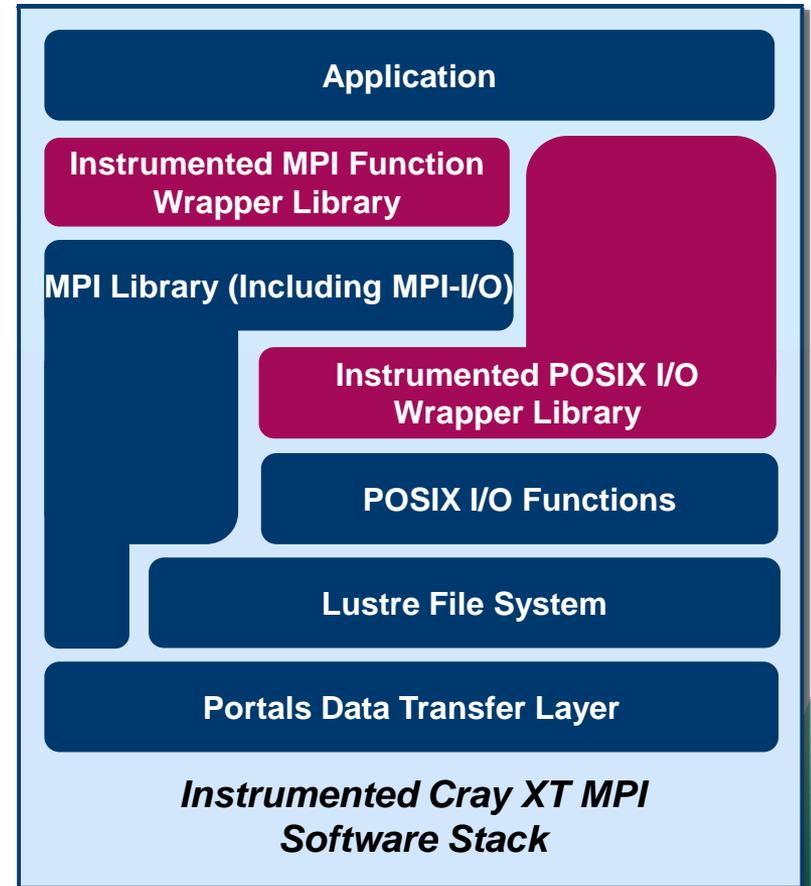


Gap may grow too large for existing techniques (e.g., checkpointing) to be viable because of decreases in system reliability as systems get larger

System Size

Insight into I/O behavior

- Performance data collection infrastructure for Cray XT
- Gathers detailed I/O request data without changes to application source code
- Useful for
 - Characterizing application I/O
 - Driving storage system simulations
 - Deciding how and where to optimize I/O



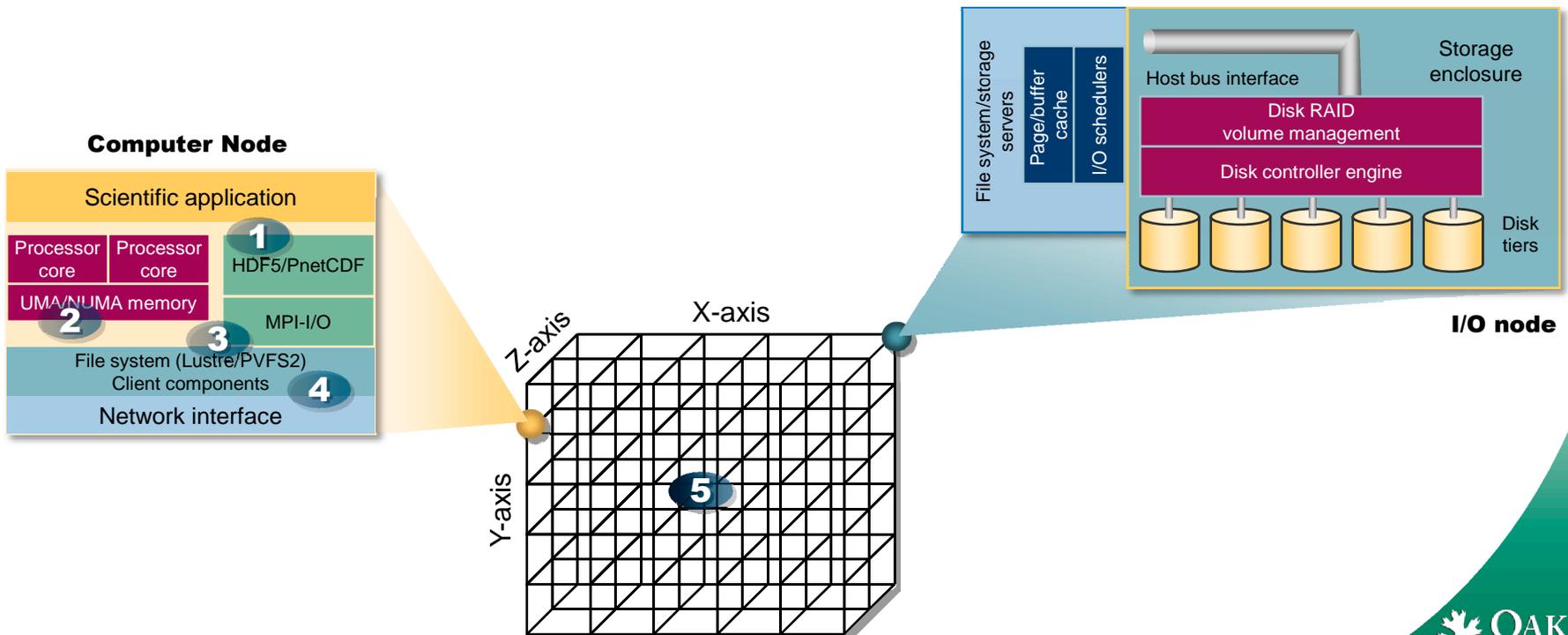
Optimization through parallel I/O libraries

- **Advantages from parallel I/O libraries**

- Interfacing application, runtime, and operating system
- Ease of solution deployment

- **Challenges approachable via libraries**

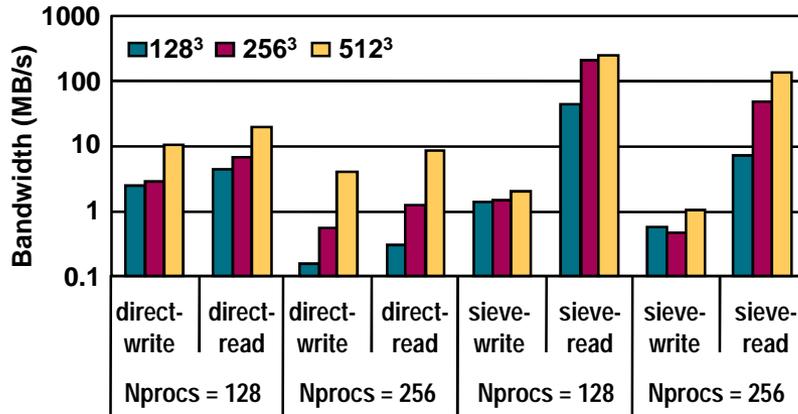
1. Application hints and data manipulation
2. Processor/memory architecture
3. Parallel I/O protocol processing overhead
4. File-system-specific techniques
5. Network topology and status



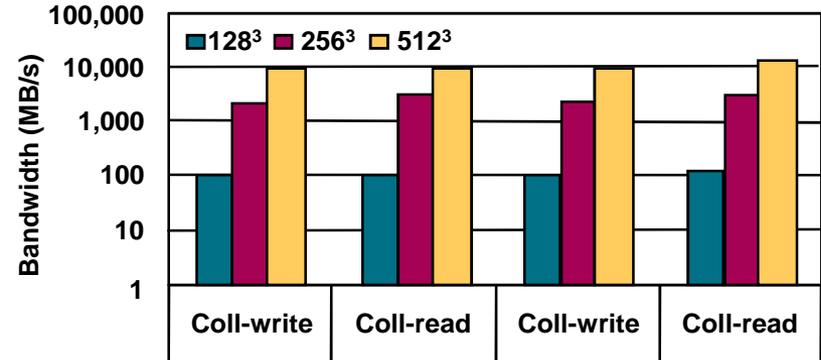
Performance and optimization I/O on Jaguar

Yu, Vetter, Oral, IPDPS 208

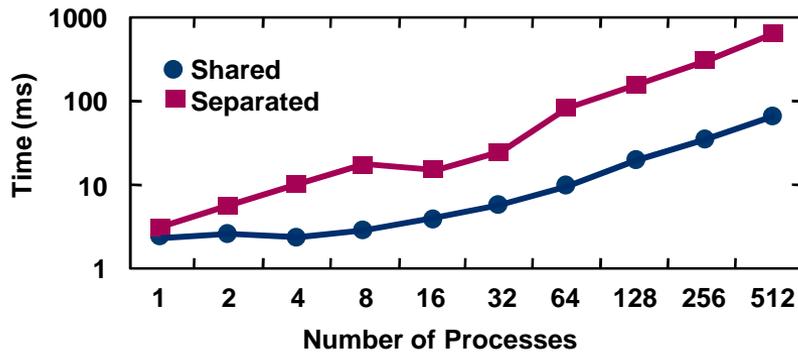
Data Sieving



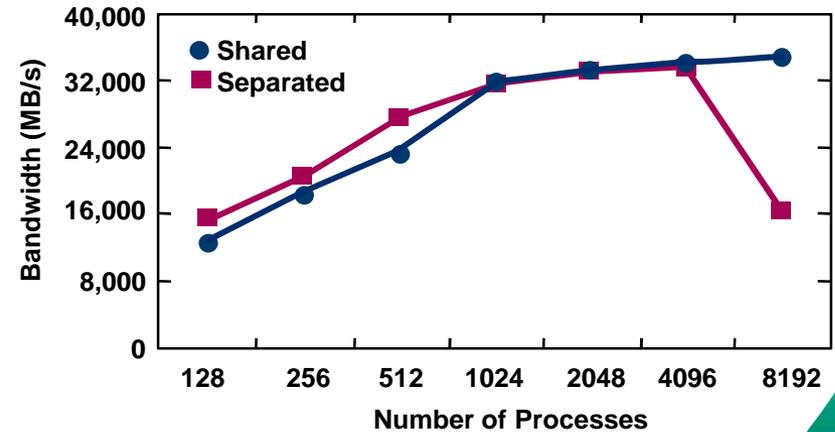
Collective I/O



Parallel Open



S3D Effective Bandwidth



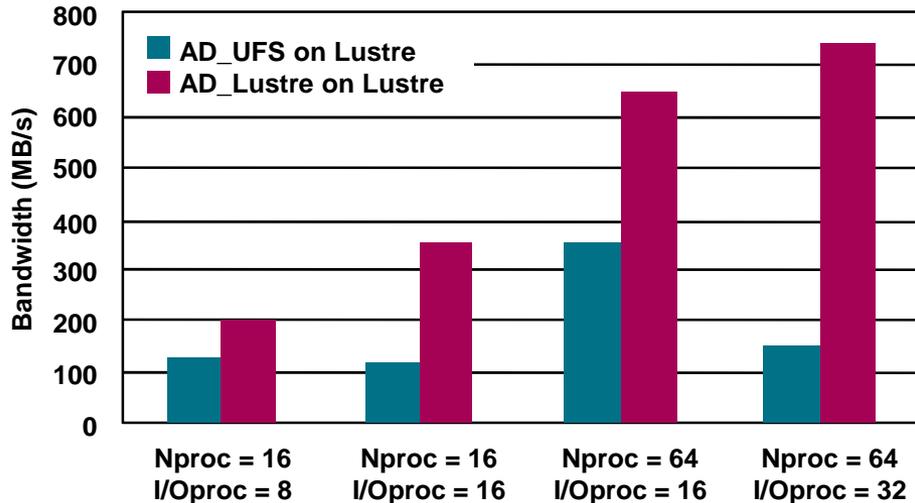
Opportunistic and adaptive MPI-I/O for Lustre (OPAL)

Yu, Vetter, Canon, SNAPI 2007

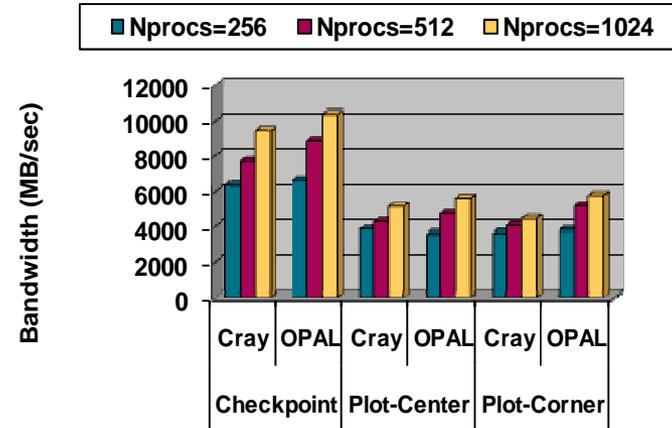
An open-source implementation of MPI-I/O
(Cray XT and Linux clusters)

- Overcome the restriction of a proprietary MPI-I/O stack
- Improved data-sieving implementation; arbitrary striping specification
- Lustre stripe-aligned file domain partitioning
- Release via MVAPICH-1.0 and MPICH2-1.0.7

NAS BT-I/O Class B with MVAPICH/Lustre

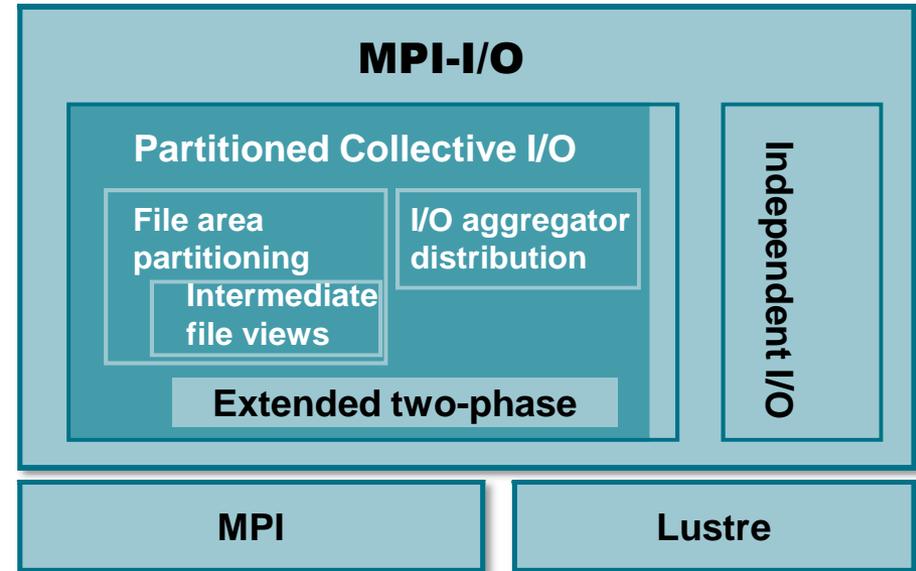


Flash I/O with OPAL

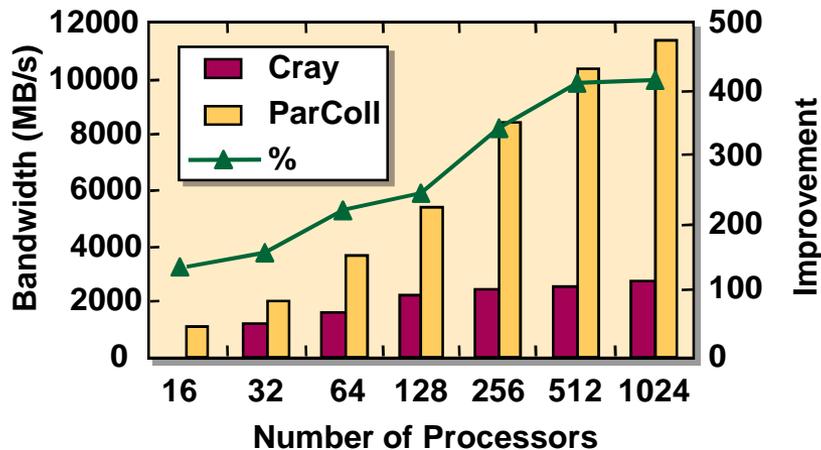


Partitioned collective I/O (ParColl)— Yu, Vetter, ICPP 2008

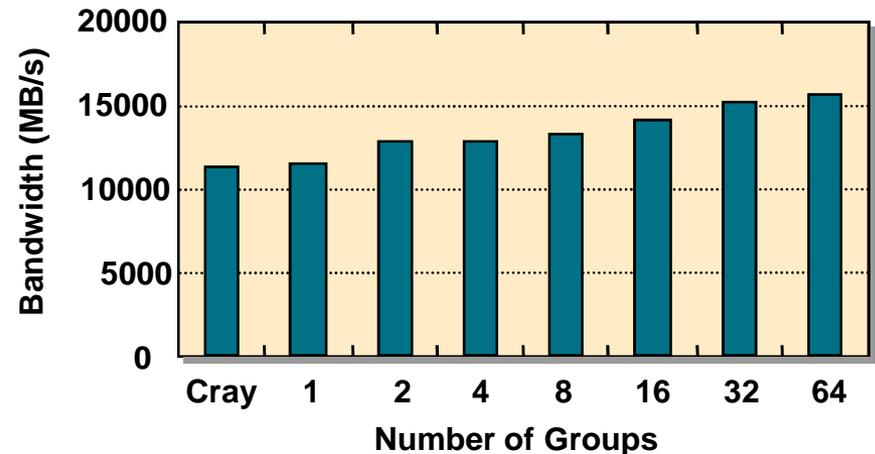
- Collective I/O is good for small I/O request aggregation
- But global synchronization within is a barrier to scalability
- ParColl partitions global processes, I/O aggregators, and the shared global file appropriately for scalable I/O aggregation



MPI-Tile-I/O with ParColl



Flash I/O with ParColl



Contacts

Jeffrey S. Vetter

Leader

Future Technologies Group

Computer Science and Mathematics Division

(865) 356-1649

vetter@ornl.gov

Weikuan Yu

(865) 574-7990

wyu@ornl.gov

Philip C. Roth

(865) 241-1543

rothpc@ornl.gov

