DataStager: Scalable Data Staging Services for Petascale Applications

Hasan Abbasi
Matthew Wolf
Karsten Schwan
Fang Zheng

College of Computing,
Georgia Institute of Technology

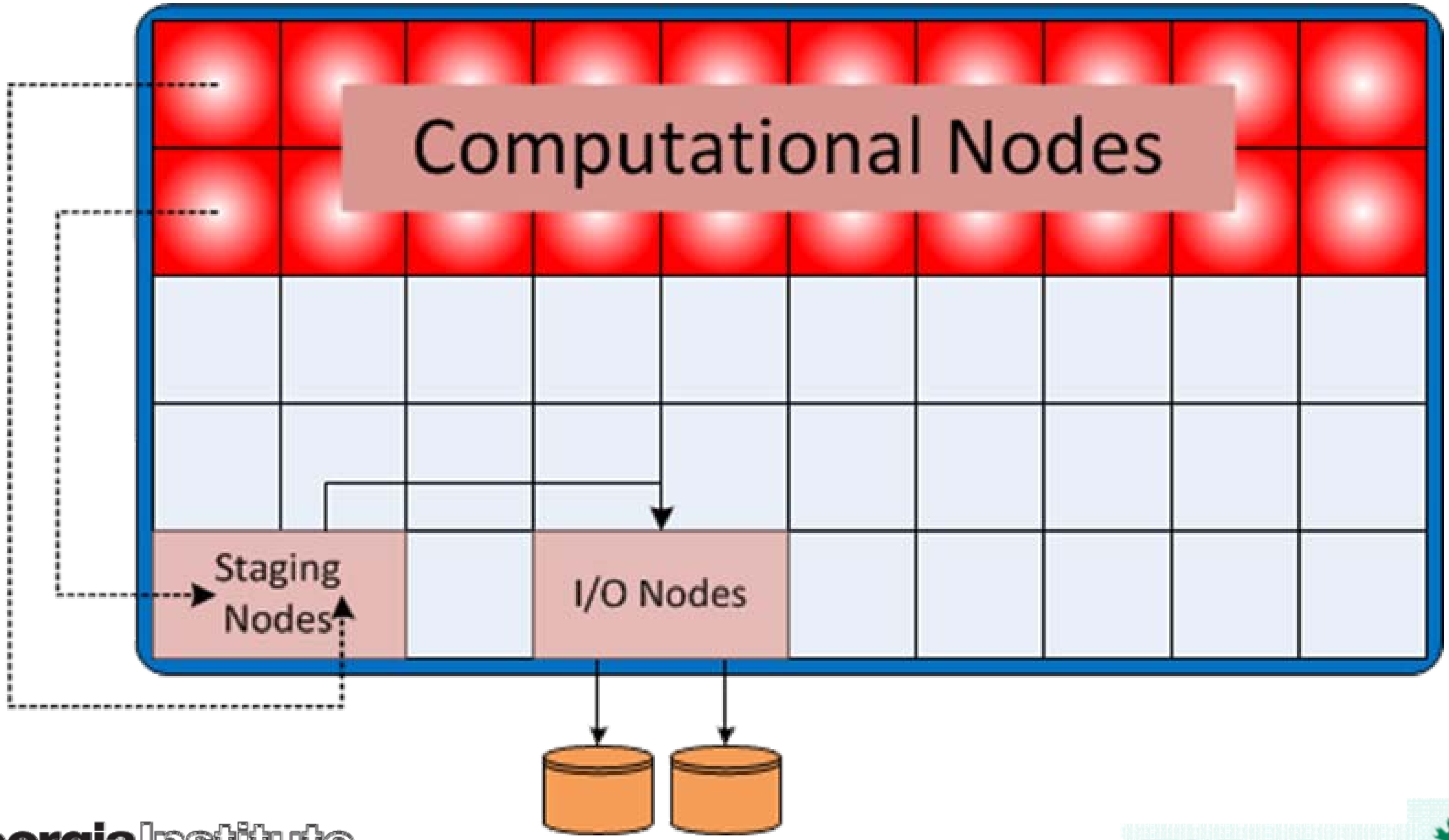
Scott Klasky

Oak Ridge National Laboratory

Motivation

- Increases in computational capacity
- Corresponding increase in data size
- I/O performance becoming a bottleneck at scale
- Information extraction increasing in importance
- Motivating application GTC

Data Staging Overview



Outline

- Data Staging Overview
- Advantages and Design highlights
- DataStager Architecture
- Scalable data movement
- Managing data transfers
  - Scheduling policies
- Evaluation
- Conclusion and the future

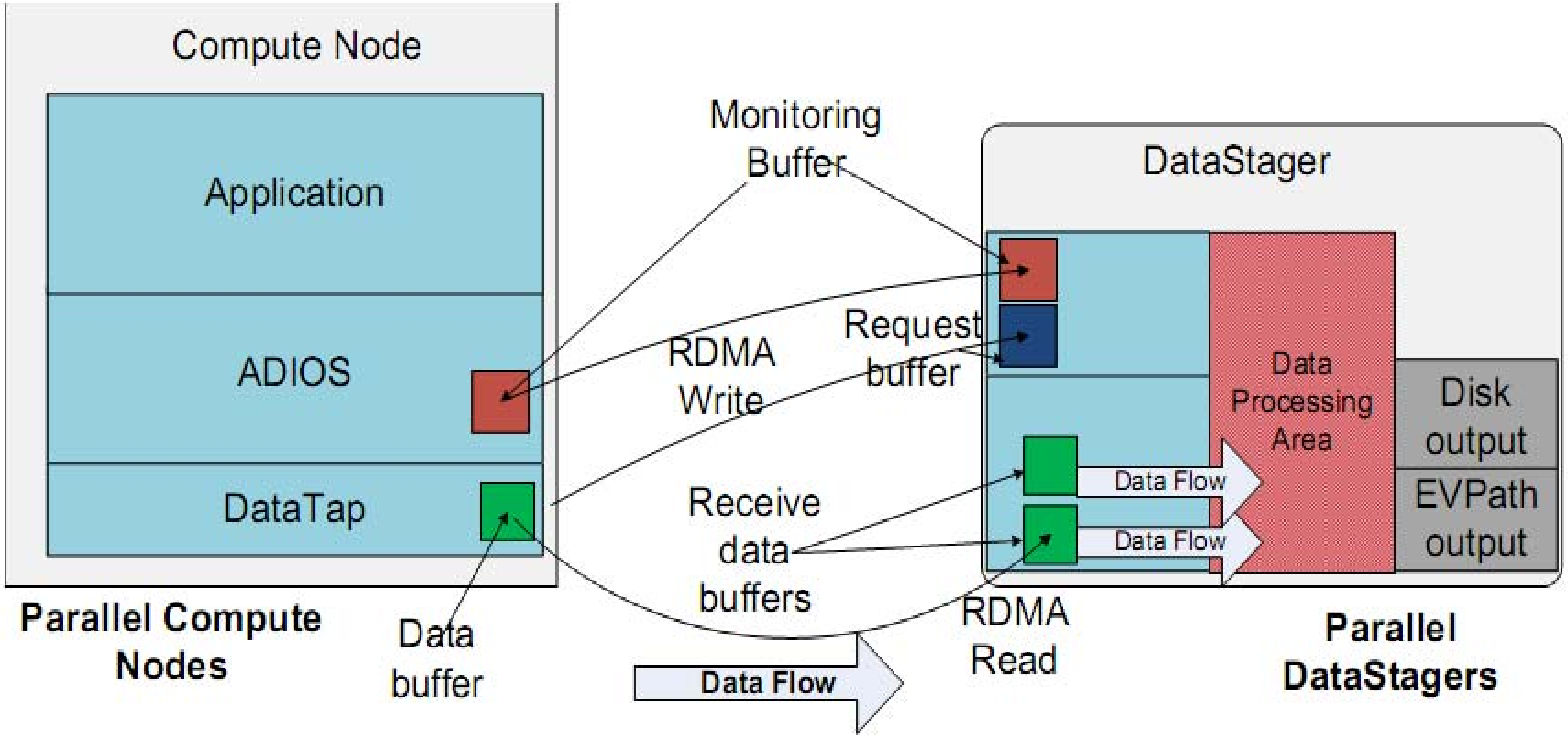
DataStaging Advantages

- Reduces performance linkage between I/O subsystem and application
- Enables optimizations based on dynamic number of writers
- High bandwidth data extraction from application

Design Highlights

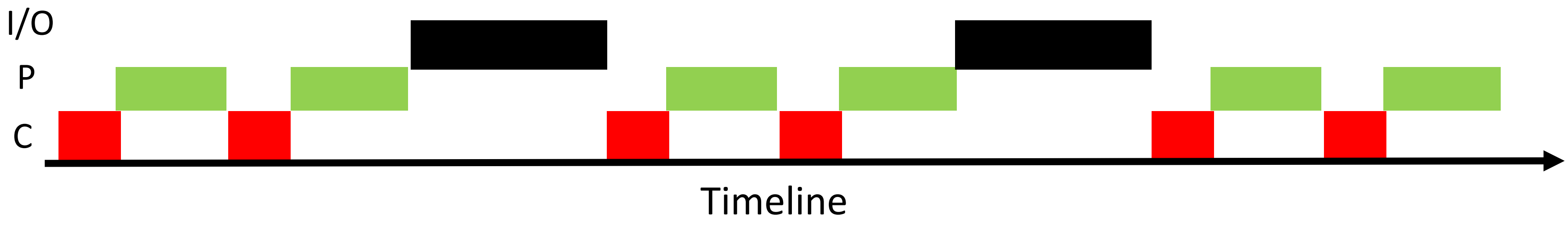
- Server directed asynchronous I/O
- ADIOS compatibility
- Structured Data Transport
- Minimal Runtime impact
- Utilize available RDMA transports
  - Implementation on Cray XT and Infiniband

DataStager Architecture



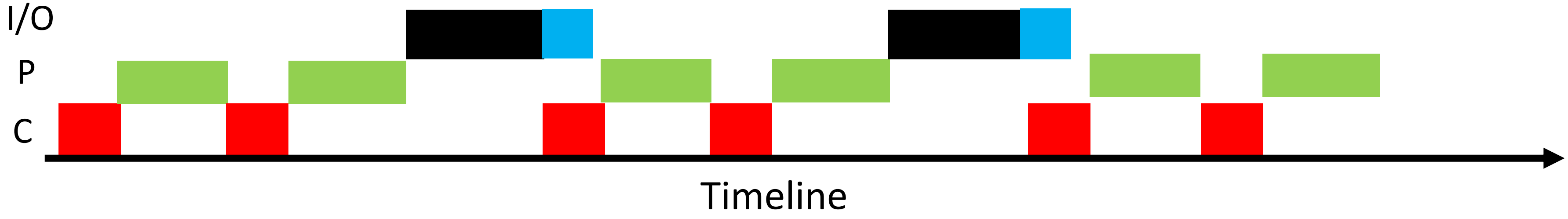
Data Movement

- Ideal synchronous (POSIX) output
- No buffering of data
- Transfer completes when function returns
- Application runtime is greatly impacted by actual output bandwidth
- Requires disabling file system buffering and has a huge performance impact



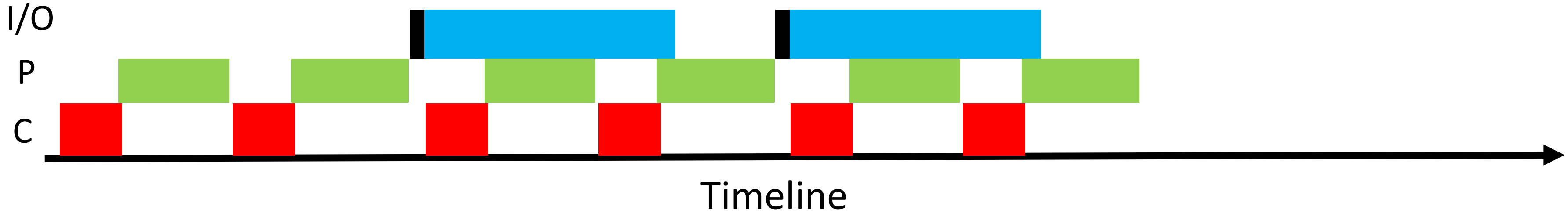
Data Movement

- Actual synchronous (POSIX) output
- Output data is partially buffered
- High throughput background transfer as Lustre drains the buffer
- Application runtime is impacted by the interference from the background data transfer



Data Movement

- Asynchronous data output (DataStager)
- All the data is buffered
- Blocking time is only the time taken to buffer data
- Data is transferred in the background concurrently with application execution
- Application runtime is impacted by the interference from the background data transfer



Management of Transfers

- Artifact of server directed I/O
  - DataTap issues a request to DataStager
  - Requests from all client nodes are queued
  - DataStager checks available buffer space on staging node
- Scheduling of data movement to reduce overhead
  - Requests are serviced in order of scheduling policy
  - Multiple requests are serviced simultaneously

Schedulers

- Constant drain scheduler (CD)
  - Transfer data as fast possible
  - Highest throughput
  - Impact on application runtime from interference

Schedulers

- State aware congestion avoidance (PA)
  - For applications with regular patterns
  - Partition application runtime into *phases*
  - Only transfers data in *computation* phase
  - Avoids interference with intra-application MPI communication

Schedulers

- Attribute aware in-order scheduler
  - Uses data attributes to order movement
  - Creates ordered data stream

Schedulers

- Rate limiting scheduler (Con_X)
  - Limits concurrent transfers to X
  - Applies to irregular applications where state aware scheduler cannot accurately predict phase
  - Reduces interference by limiting bandwidth usage

Schedulers

- Multiple schedulers can be used together
- Rate limiting + state aware (PA_Con_X)
  - Can work across range of application behaviors
  - Further reduce impact from background I/O

Phase detection

- Uses ADIOS user specified hints

  adios_start_calculation

  adios_end_calculation

- DataStager maintains application timing information

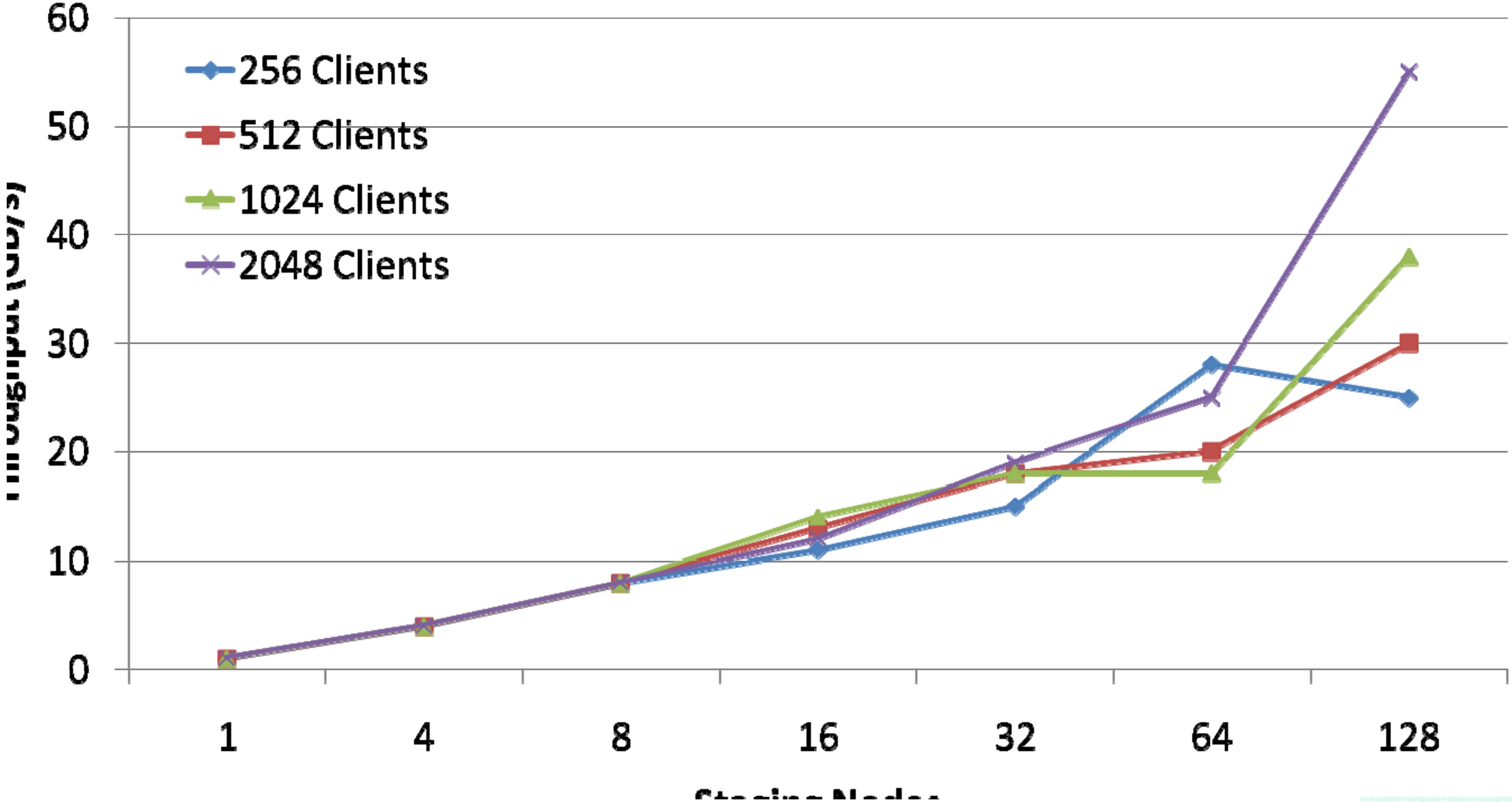
Evaluation

- Full application benchmark
  - Measure real application overhead
  - Not bandwidth
- Gyrokinetic Turbulence Code
  - Particle-in-cell
  - Scalable
  - We use *weak scaling* – 180 MB output/process
- All benchmarks on Jaguar at ORNL
  - Quad core, 2 GB/core
  - Seastar2 interconnect

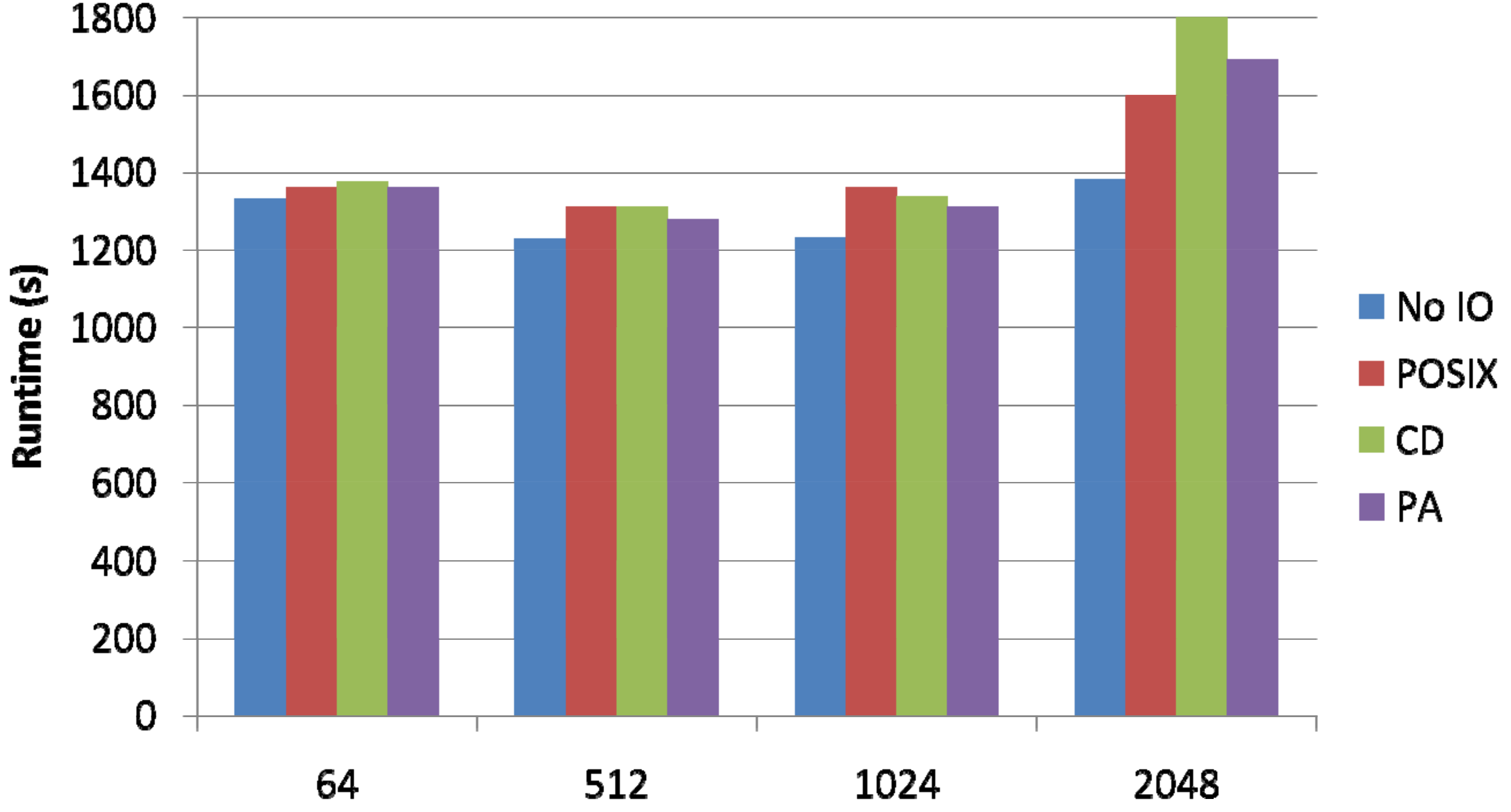
Abbreviations

- Continuous drain scheduler – CD
- State aware scheduler – PA
- Concurrency limiting – Con_X
  - X is 1 or 4
- Combination Scheduler
  - PA_Con_X

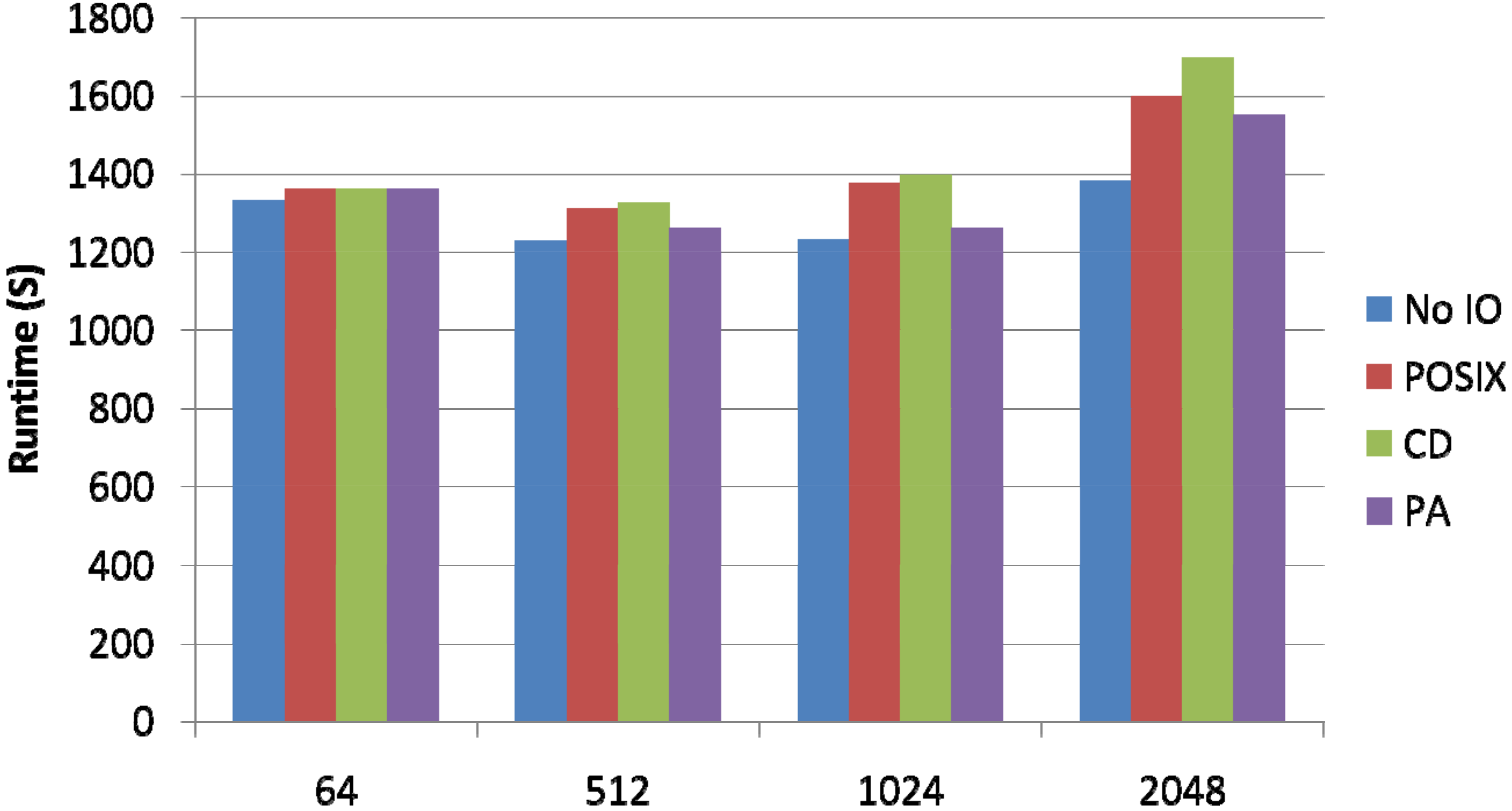
DataStager Maximum Throughput



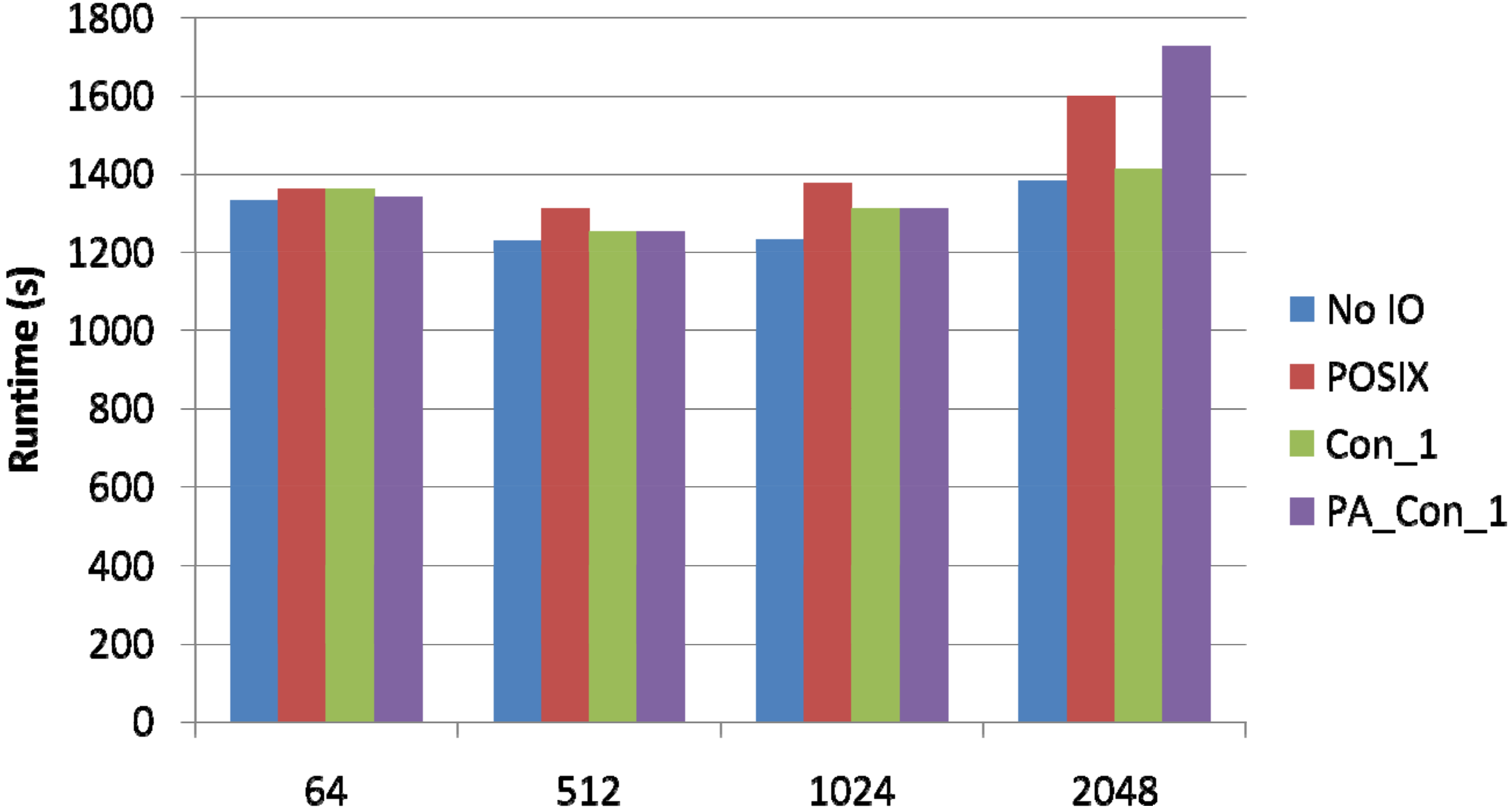
4 staging nodes



16 staging nodes



Advanced Scheduling (4)

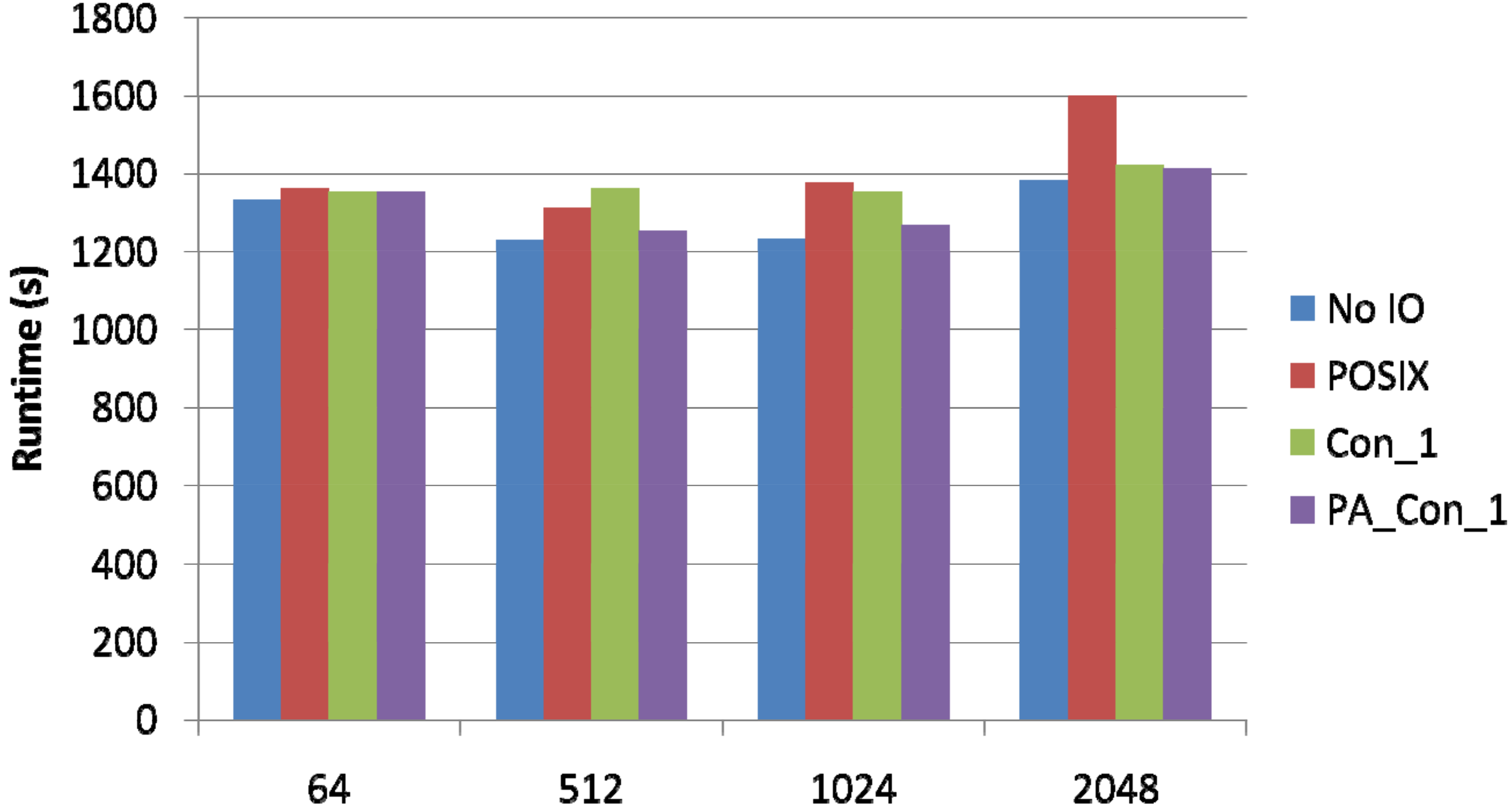


Georgia Institute of Technology

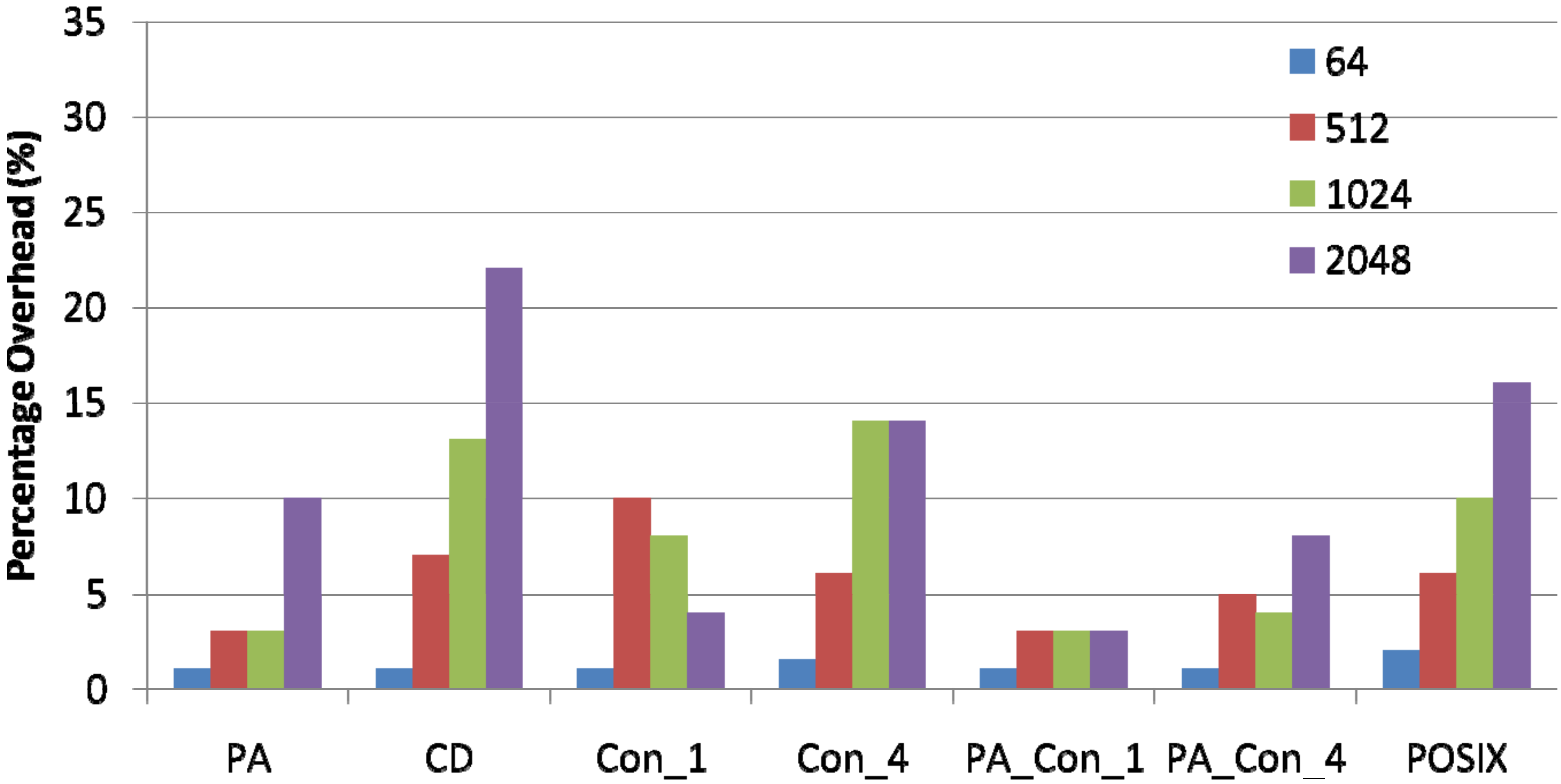
Office of Science U.S. DEPARTMENT OF ENERGY

OAK RIDGE National Laboratory

Advanced Scheduling (16)



Runtime Overhead comparison for all evaluated scheduling mechanism 16 Stagers

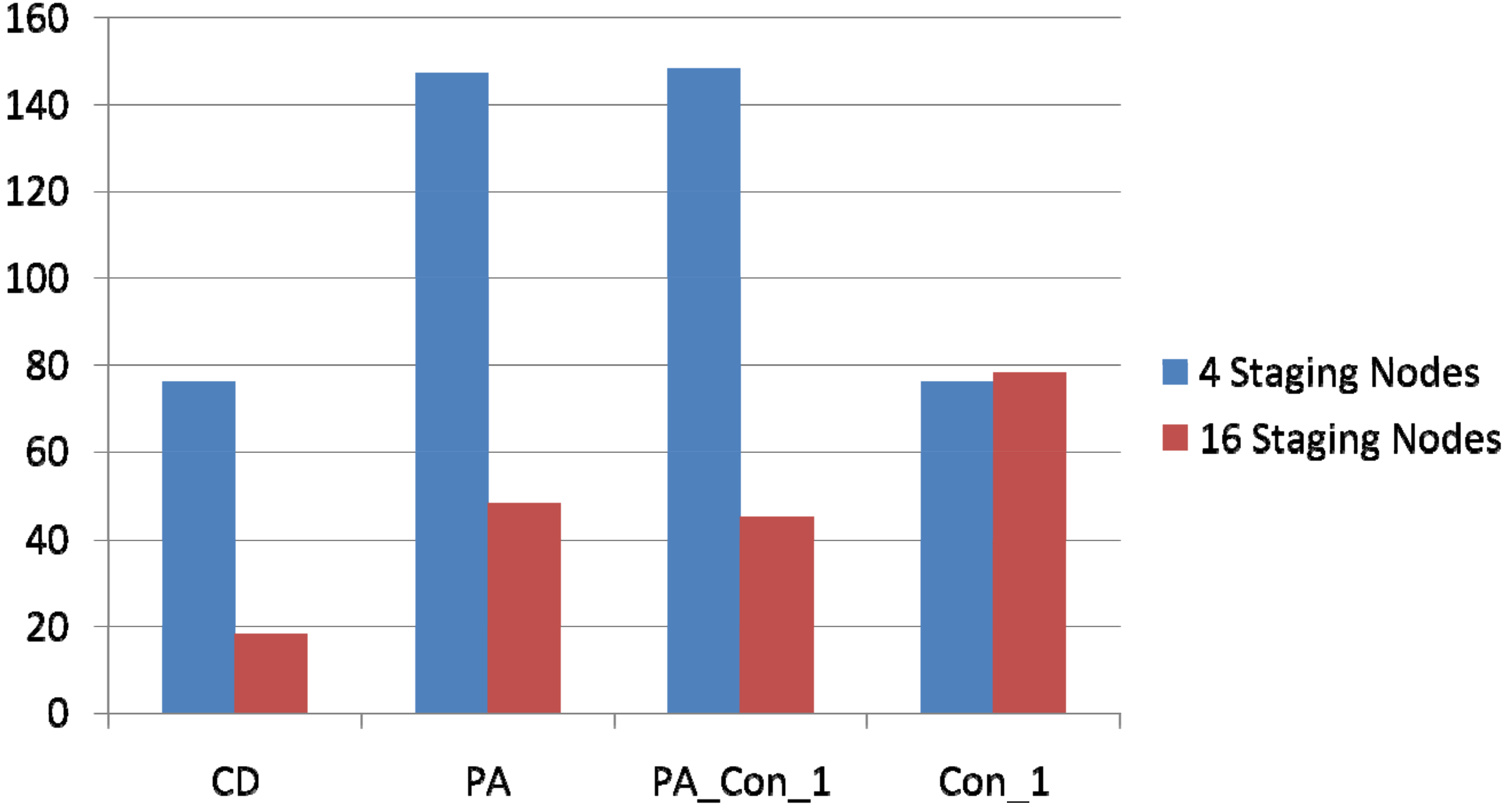


Georgia Institute of Technology

Office of Science U.S. DEPARTMENT OF ENERGY

OAK RIDGE National Laboratory

Completion Time



Georgia Institute of Technology

Office of Science
U.S. DEPARTMENT OF ENERGY

OAK RIDGE
National Laboratory

Conclusions

- Scalable data movement with shared resources requires us to manage the transfers
- Estimating an application's phase enables a low cost mechanism for avoiding interference
- Scheduling properly can greatly reduce the impact of I/O

Future Work

- DataStager is the first component of the larger Data Service approach
- Using staging area for information processing
  - Visualization
  - Data output
  - Application coupling
  - Data analytics
- E2e management of the data pipeline