Decoupled Execution Paradigm for Data-Intensive High-End Computing

Project Investigators

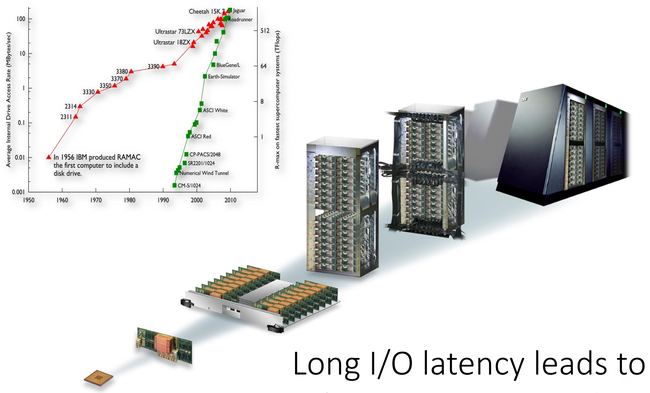
Xian-He Sun — Illinois Institute of Technology
Yong Chen — Texas Tech University
William Gropp — University of Illinois at Urbana-Champaign
Rajeev Thakur — Argonne National Laboratory

Graduate Students

Yanlong Yin — Illinois Institute of Technology
Chao Chen — Texas Tech University
Hassan Eslami — University of Illinois at Urbana-Champaign

I/O Bottleneck

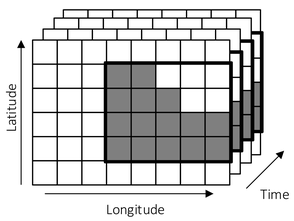
Significant gap between computing and I/O.

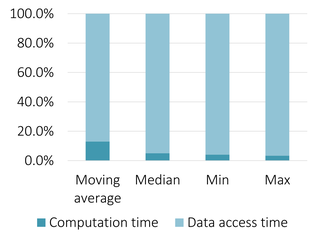


Long I/O latency leads to performance degradation.

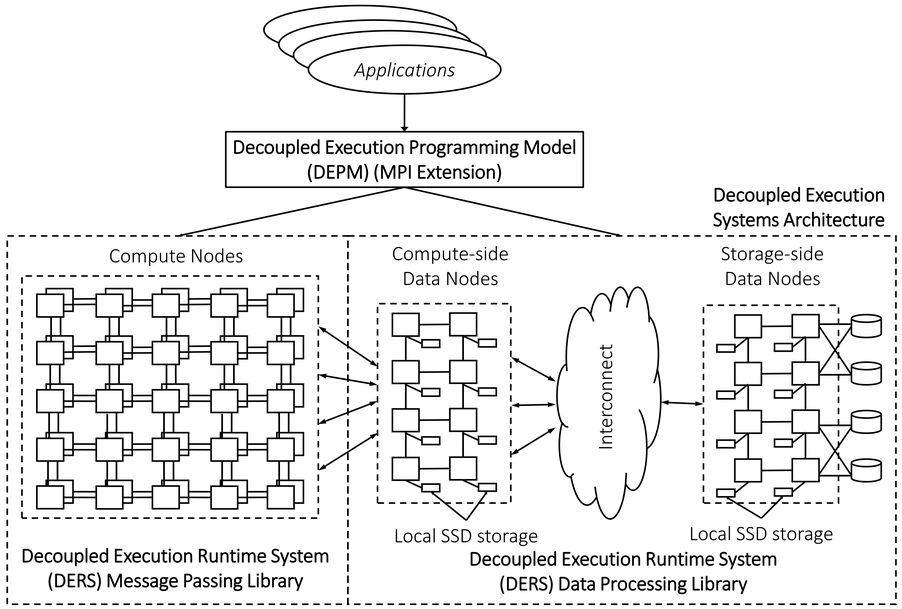
Motivating Example

- Data commonly represented by a multi-dimensional array-based data model
- Read required data from storage servers to compute nodes
- Perform computations on desired data with specified conditions, and then write back
- With clear data retrieval and processing phases and computing and simulation phases
- Data access and movement often dominate execution time for data-intensive HEC apps





System Architecture



- Compute-side data nodes reduce the size of computing generated data before sending it to storage nodes
- Storage-side data nodes reduce the size of data retrieved from storage before sent
- Data nodes conduct decoupled data-intensive operations and optimizations to reduce the data size and movement
- Compute nodes take care of computation-intensive operations collectively

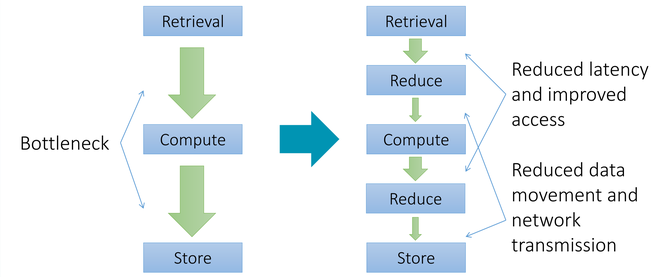
Programming Model

- To determine operations to be passed to data nodes
- Designed as an MPI extension, allowing users to specify operations conducted on data nodes
- Results sent back to compute nodes for further processing

Runtime System

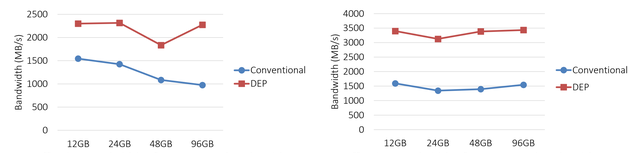
- Relies on two libraries, message passing library and data processing library
- Message passing library focuses on the memory abstraction and provides support for computation-intensive operations
  - Leverage the existing MPI library for this purpose
- Data processing library focuses on the I/O abstraction and provides support for data-intensive operations

Execution Paradigm Comparison



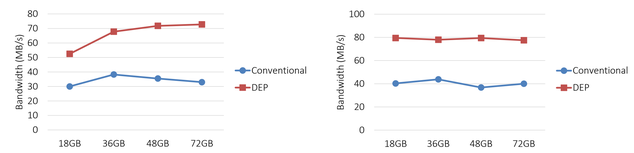
Experimental Results

- Two application kernels
  - Kernel calculation of the CESM that computes the moving average of selected area of specified data
  - Flow routing and flow accumulation calculations in geographic information system

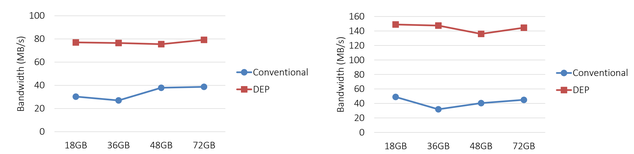


Effective Bandwidth of CESM Kernel Code with Different Data Sets on 96 Nodes, with 24 storage-side data nodes

Effective Bandwidth of CESM Kernel Code with Different Data Sets on 96 Nodes, with 48 storage-side data nodes



Effective Bandwidth of Flow Routing and Accumulation Code with Different Data Sets on 24 Nodes
Left: with 6 storage-side data nodes. Right: with 12 storage-side data nodes.



Effective Bandwidth of Flow Routing and Accumulation Code with Different Data Sets on 48 Nodes
Left: with 12 storage-side data nodes. Right: with 24 storage-side data nodes.

ACKNOWLEDGMENT: This research is sponsored in part by the U.S. National Science Foundation under grants CNS-1162540, CNS-1162488, CNS-1161507 and by the NSF grant CCF-0937877.

ILLINOIS INSTITUTE OF TECHNOLOGY · TEXAS TECH UNIVERSITY · ILLINOIS · Argonne NATIONAL LABORATORY · NSF

http://discl.cs.ttu.edu/dep