# Collective Framework and Performance Optimization to Open MPI for Cray XT 5 platforms

Cray Users Group 2011

Joshua S. Ladd, Manjunath Gorentla Venkata, Pavel Shamis, Richard L. Graham
Computer Science & Mathematic Division
Oak Ridge National Laboratory

# Collectives are Critical for HPC Application Performance

- A large percentage of application execution time is spent in the global synchronization operations (collectives)

- Moving towards exascale systems (million processor cores), the time spent in collectives only increases

- Performance and scalability of HPC applications requires efficient and scalable collective operations

# Weakness in current Open MPI implementation

Open MPI lacks support for

- Customized collective implementation for arbitrary communication hierarchies
- Concurrent progress of collectives on different communication hierarchies
- Nonblocking collectives
- Taking advantage of capabilities of recent network interfaces (example offload capabilities)
- Efficient point-to-point message protocol for Cray XT platforms

# Cheetah : A Framework for Scalable Hierarchical Collectives

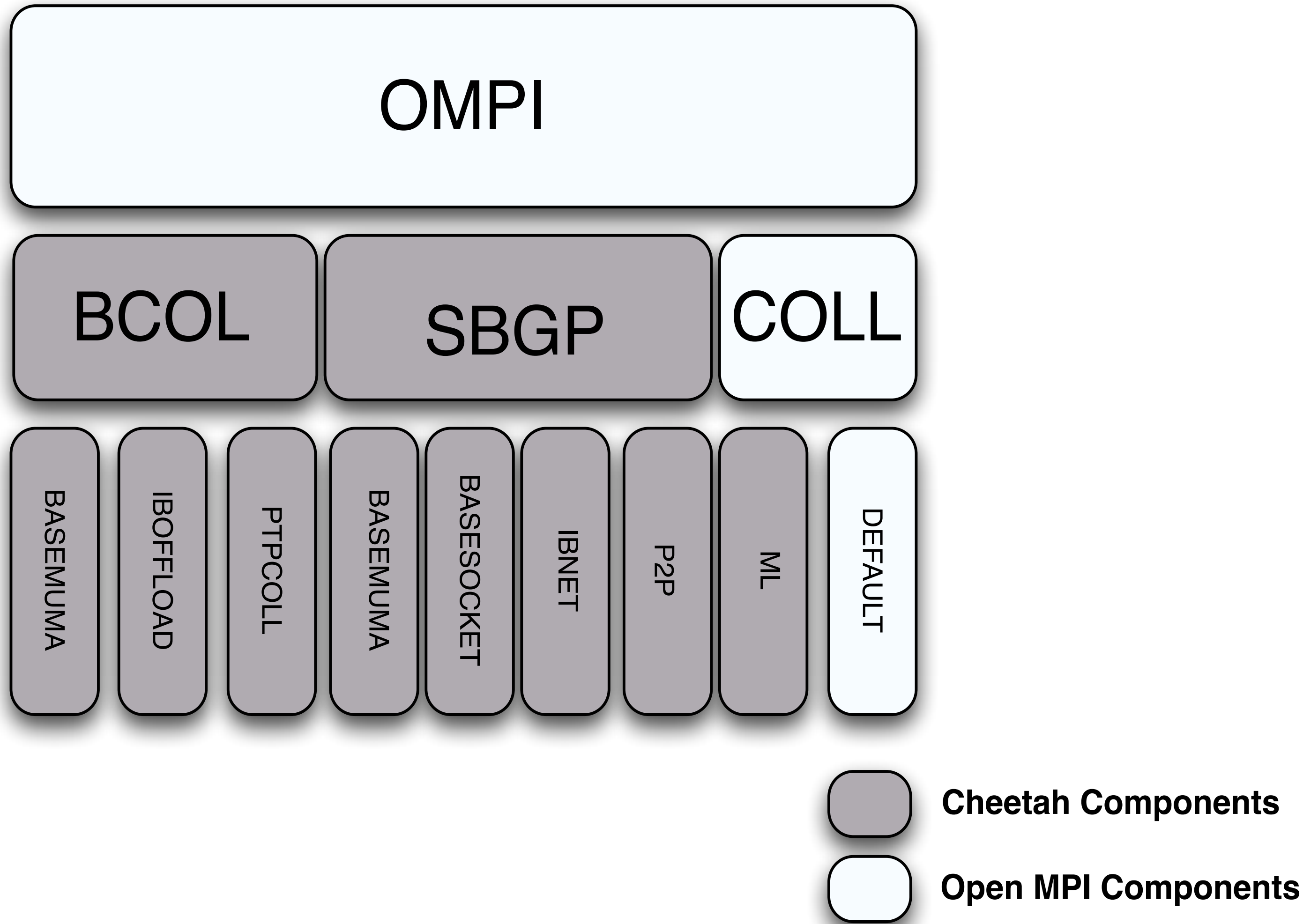
## Goals of the framework

- Provide building blocks for implementing collectives for arbitrary communication hierarchy
- Support collectives tailored to the communication hierarchy
- Support both blocking and nonblocking collectives efficiently
- Enable building collectives customized for the hardware architecture

# Cheetah Framework : Design principles

- Collective operation is split into collective primitives over different communication hierarchies
- Collective primitives over the different hierarchies are allowed to progress concurrently
- Decouple the topology of a collective operation from the implementation, enabling the reusability of primitives
- Design decisions are driven by nonblocking collective design, blocking collectives are a special case of nonblocking ones
- Use Open MPI component architecture

# Cheetah is Implemented as a Part of Open MPI



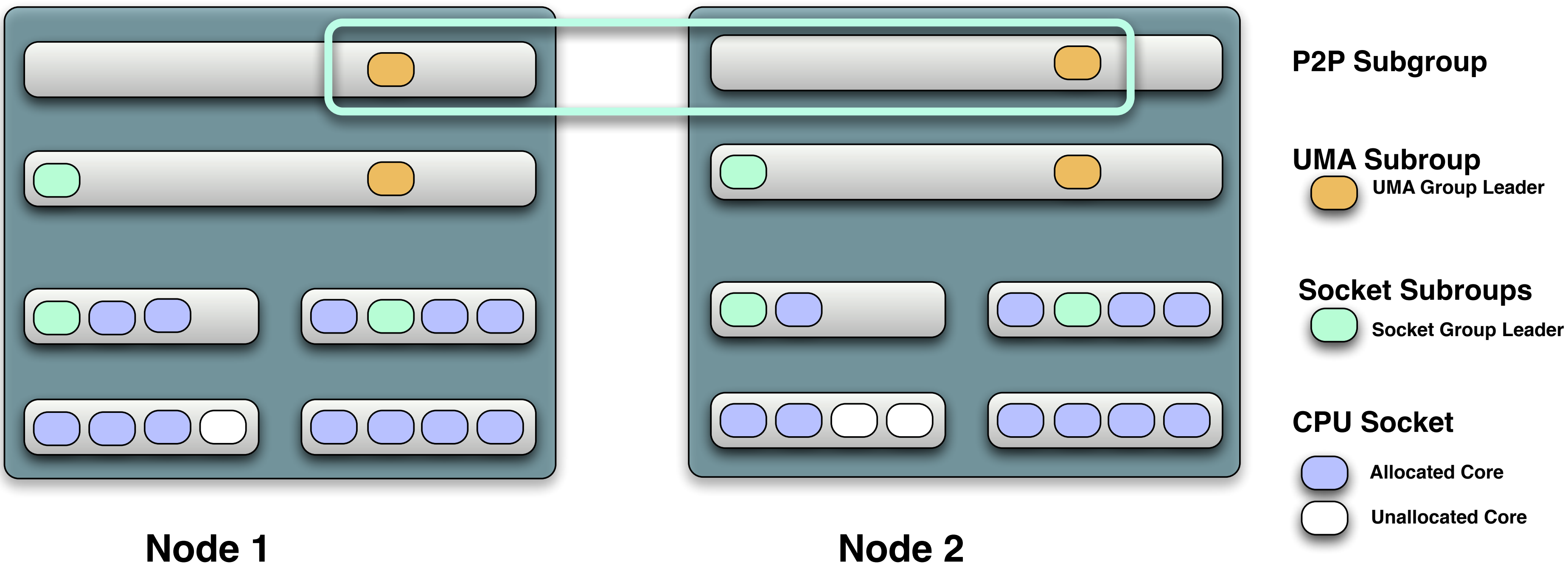
# Cheetah Components and its Functions

- Base Collectives (BCOL) – Implements basic collective primitives
- Subgrouping (SBGP) – Provides rules for grouping the processes
- Multilevel (ML) – Coordinates collective primitive execution, manages data and control buffers, and maps MPI semantics to BCOL primitives
- Schedule – Defines the collective primitives that are part of collective operation
- Progress Engine – Responsible for starting, progressing and completing the collective primitives

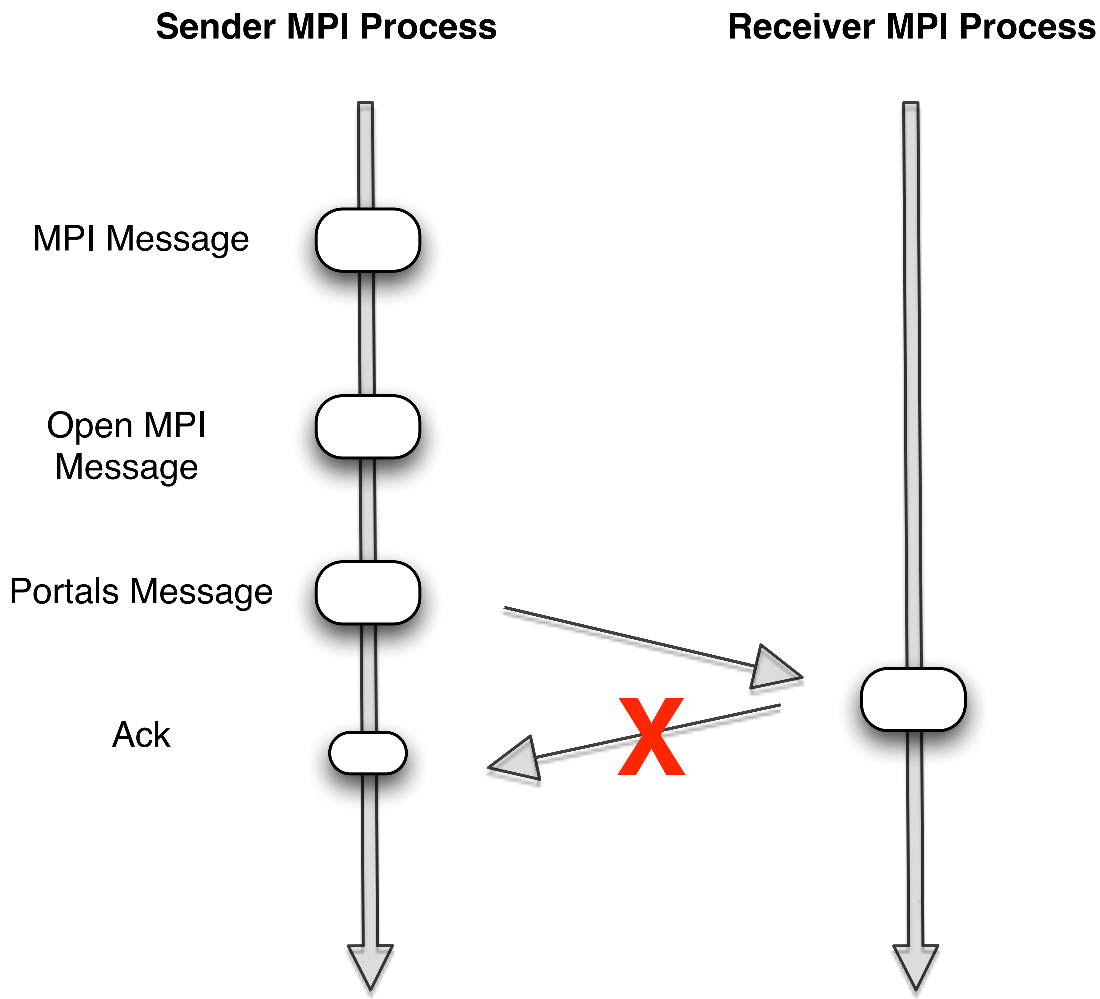
# BCOL Component – Base collective primitives

- **Provides collective primitives that are optimized for certain communication hierarchies**
  - **BASESMUMA: Shared memory**
  - **P2P: SeaStar 2+, Ethernet, InfiniBand**
  - **IBNET: ConnectX-2**
- **A collective operation is implemented as a combination of these primitives**
  - **Example, *n* level Barrier can be a combination of Fanin ( first *n-1* levels), Barrier (n$^{th}$ level) and Fanout ( first *n-1* levels)**

# SBGP Component – Group the Processes Based on the Communication Hierarchy



# Open MPI portals BTL optimization




**Portal acknowledgment is not required for Cray XT 5 platforms as they use Basic End to End Protocol (BEER) for message transfer**

# Experimental Setup

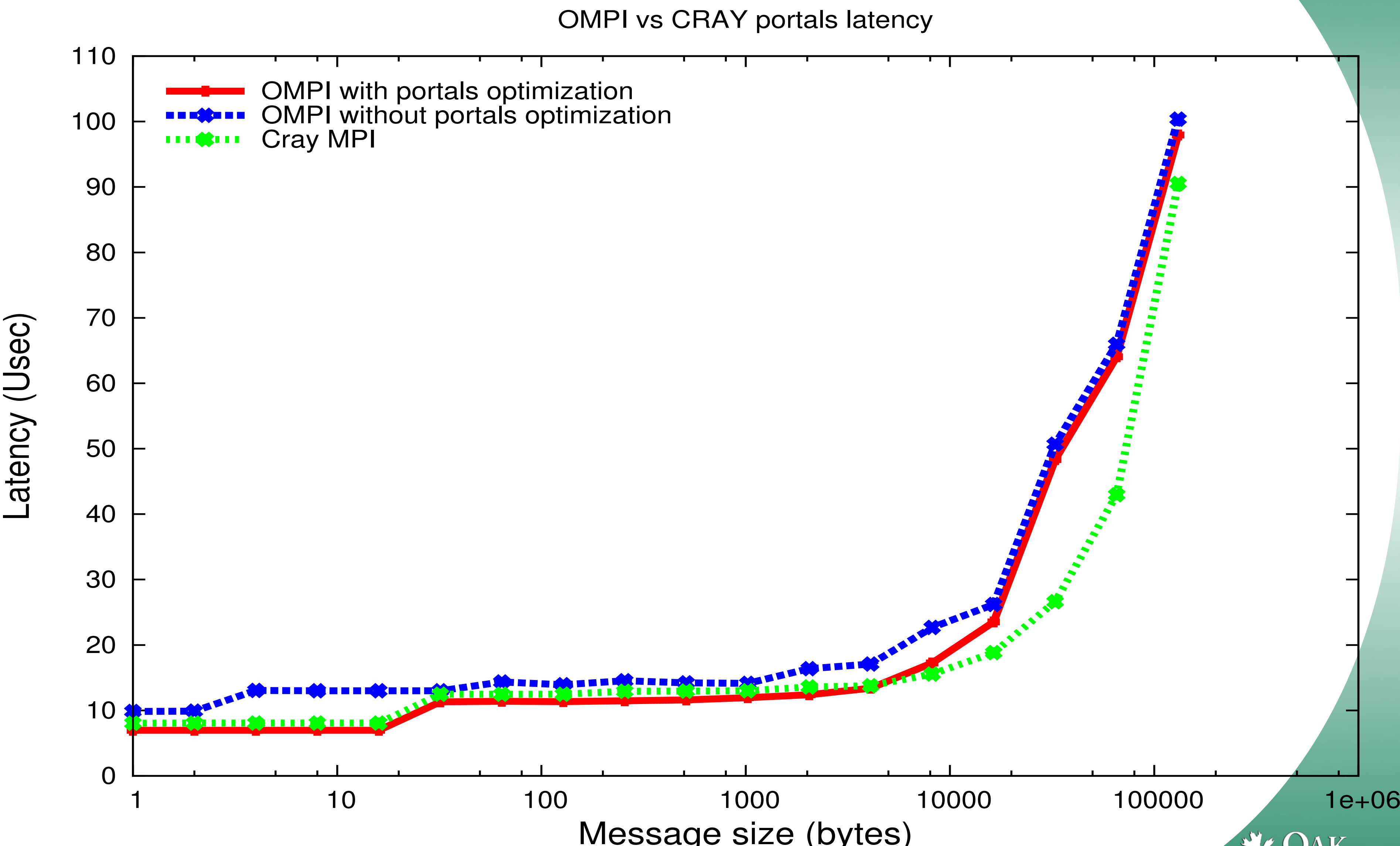
- **Hardware :**

  **Jaguar**

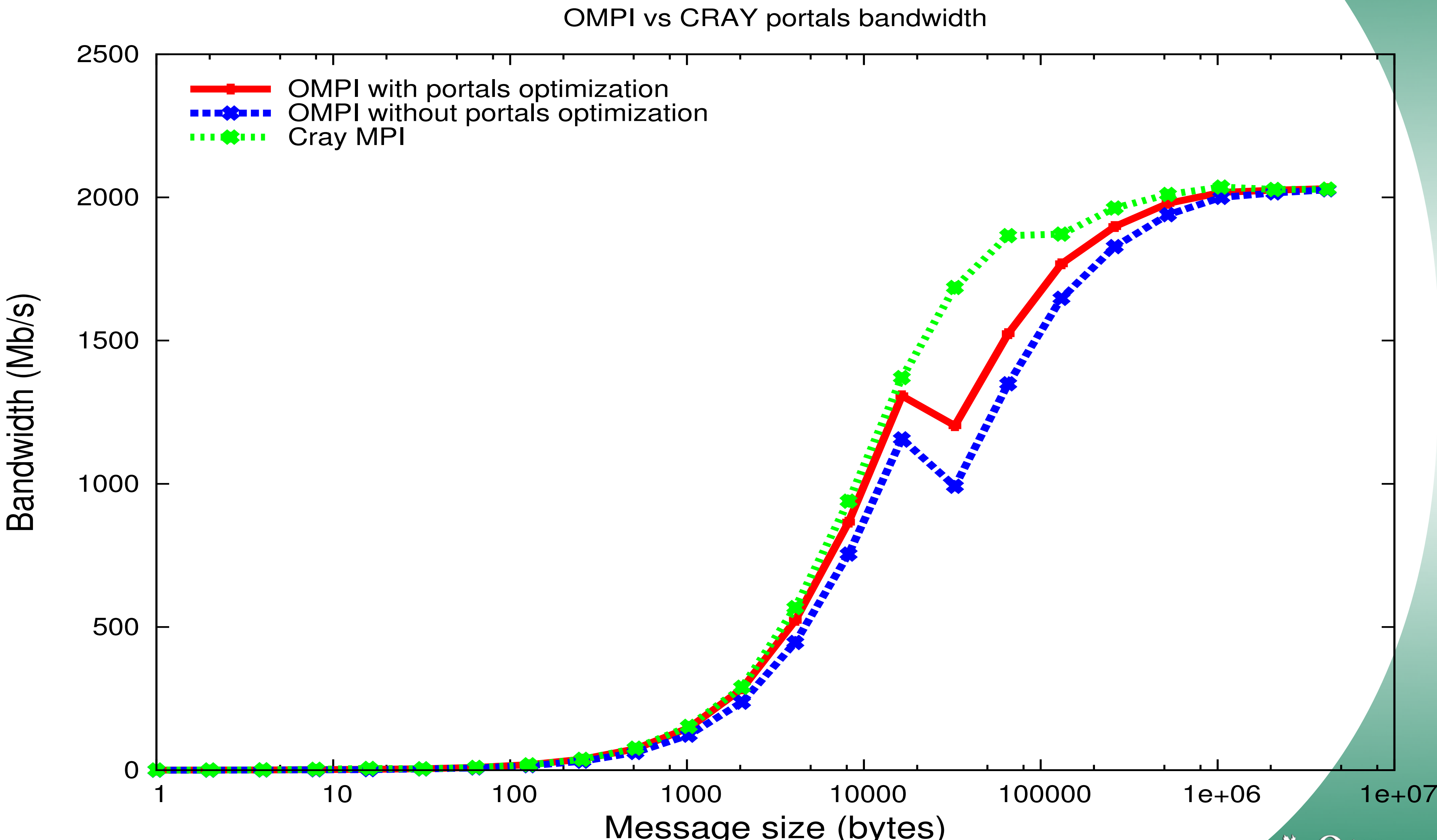
  - 18,688 Compute Nodes
  - 2.6 GHz AMD Opteron (Istanbul)
  - SeaStar 2+ Routers connected in a 3D torus topology

- **Benchmarks :**
  - Point-to-Point : OSU Latency and Bandwidth
  - Collectives :
    - Broadcast in a tight loop
    - Barrier in a tight loop

# 1 Byte Open MPI P2P Latency is 15% better than Cray MPI

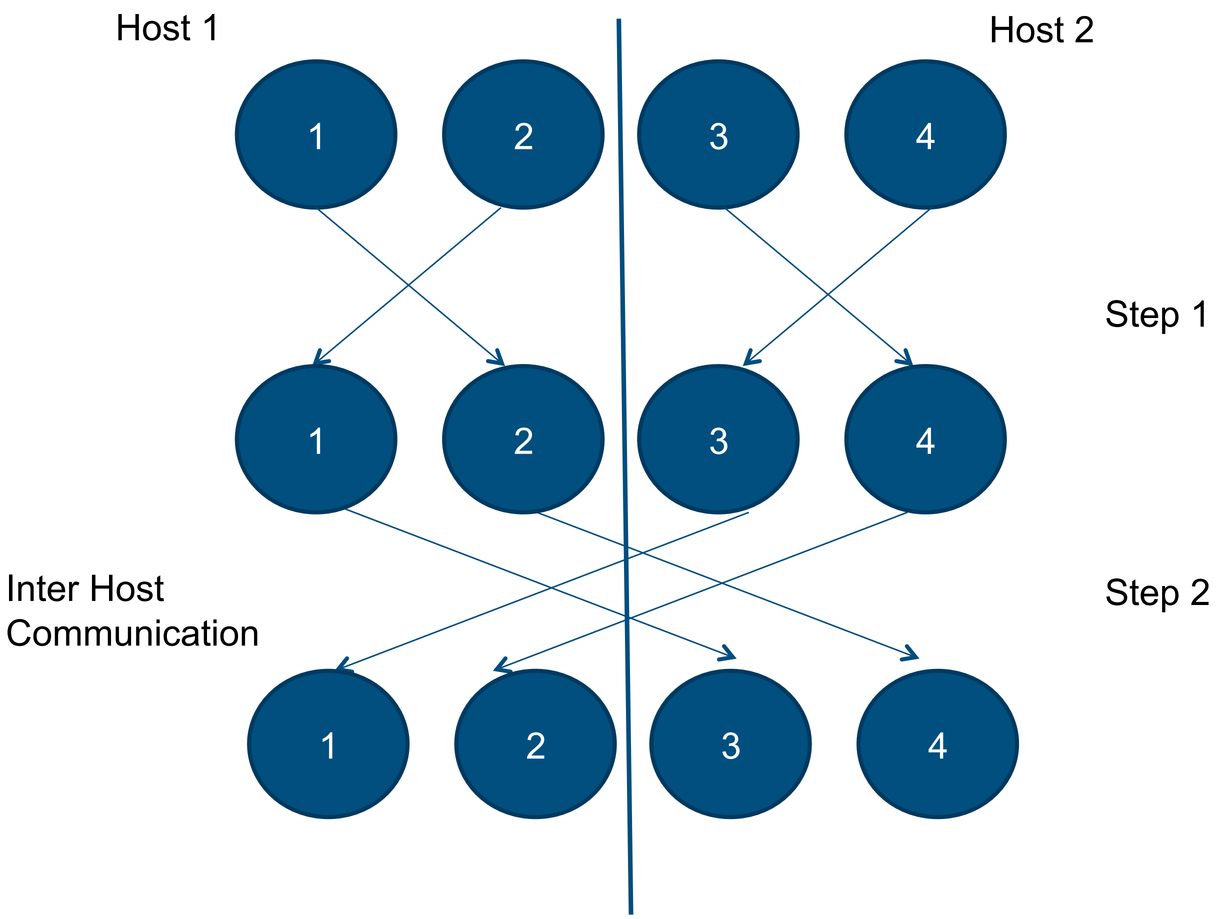


# Open MPI and Cray MPI bandwidth saturate at ~2 Gbp/s

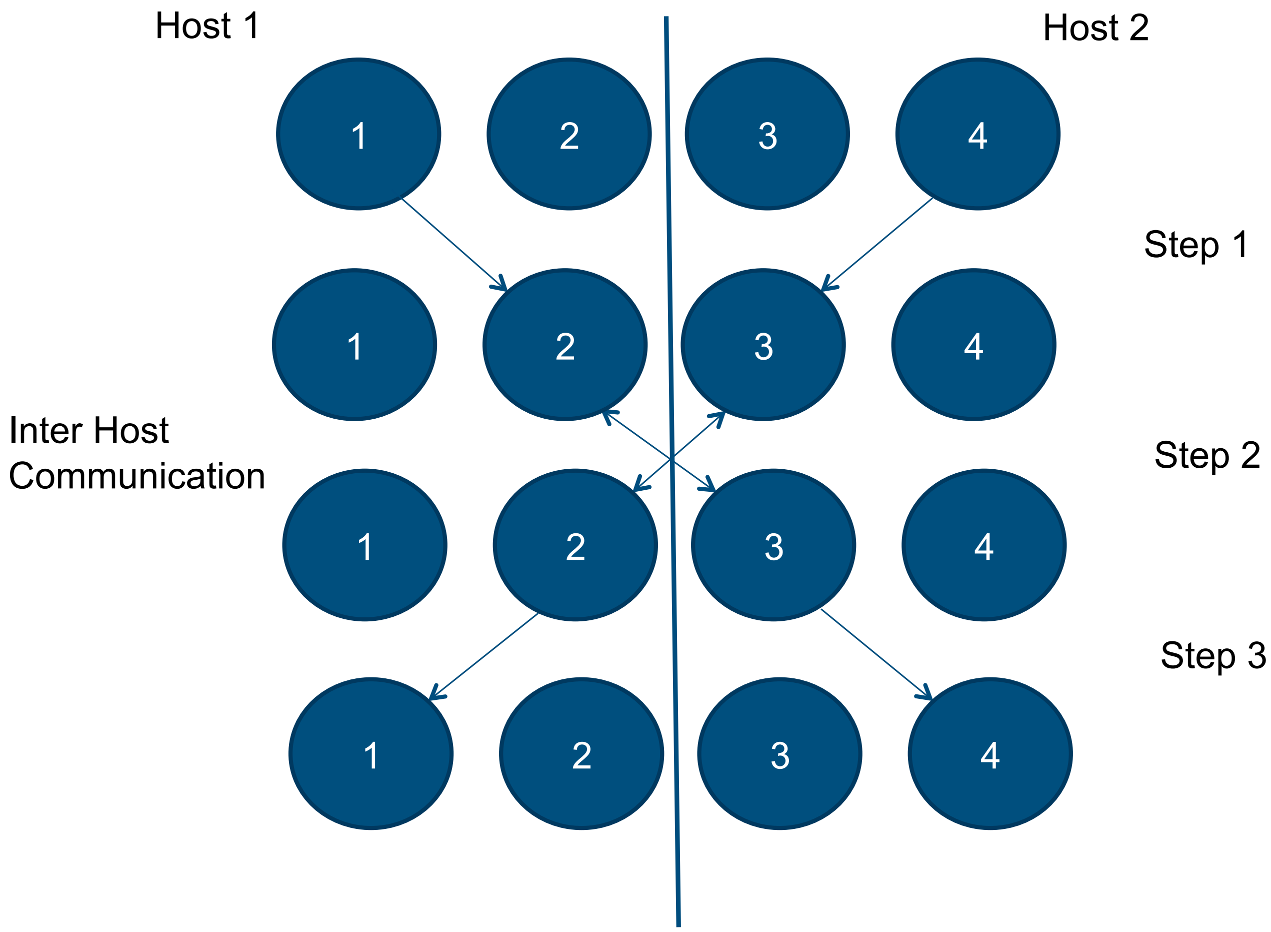


# Hierarchical Collective Algorithms

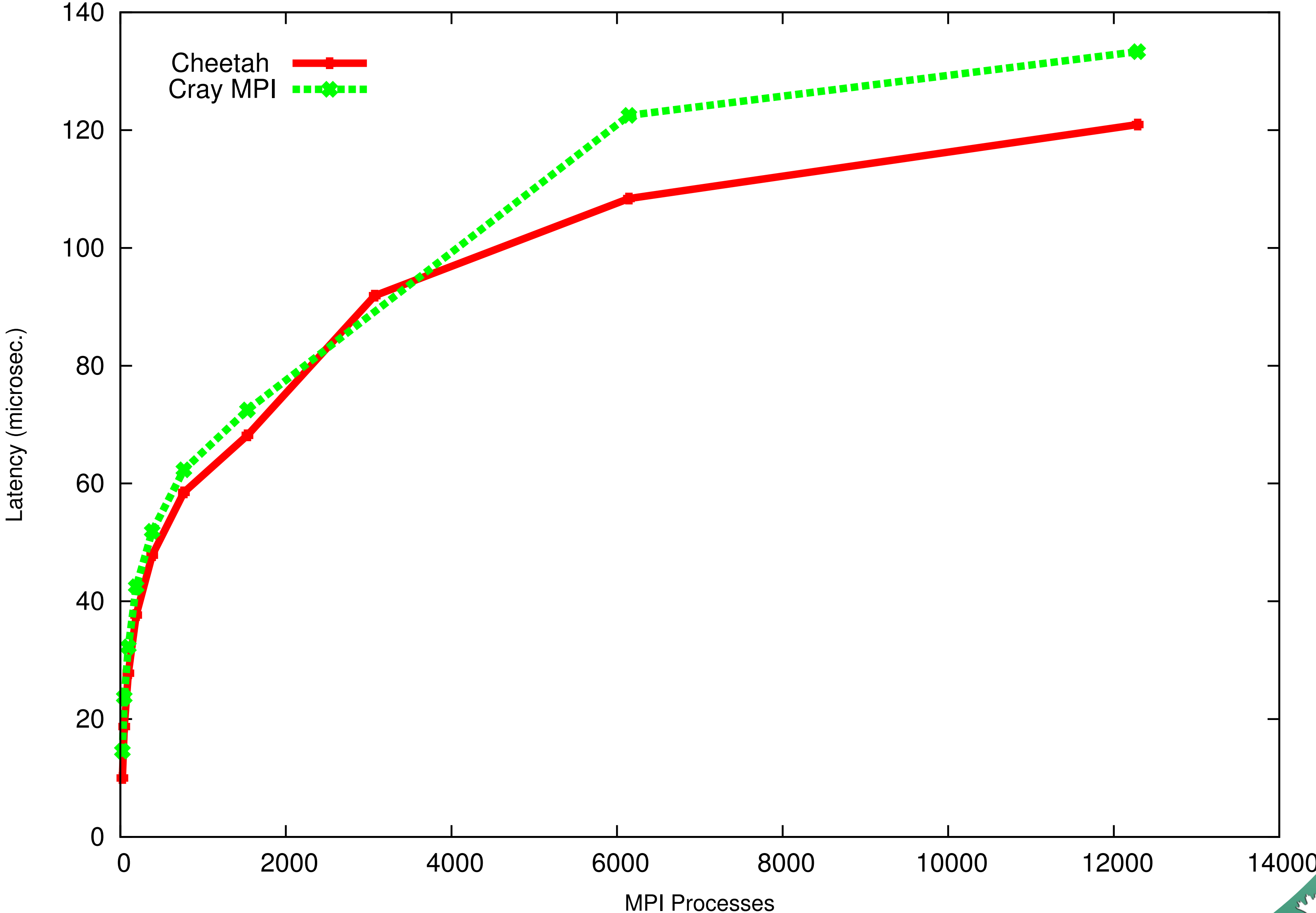
# Flat Barrier Algorithm



# Hierarchical Barrier Algorithm



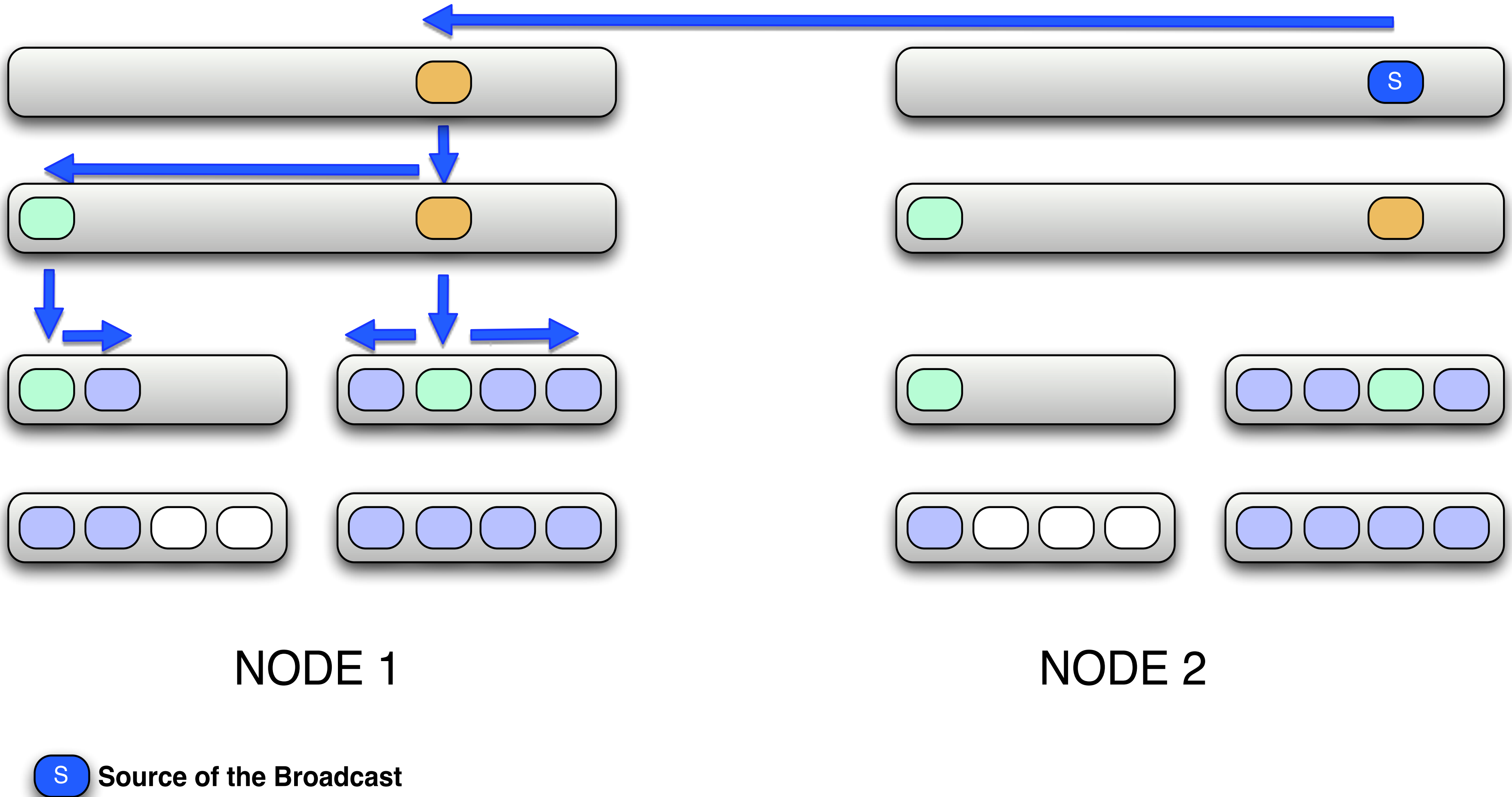
# Cheetah's Barrier Collective Outperforms the Cray MPI Barrier by 10%






OAK RIDGE National Laboratory

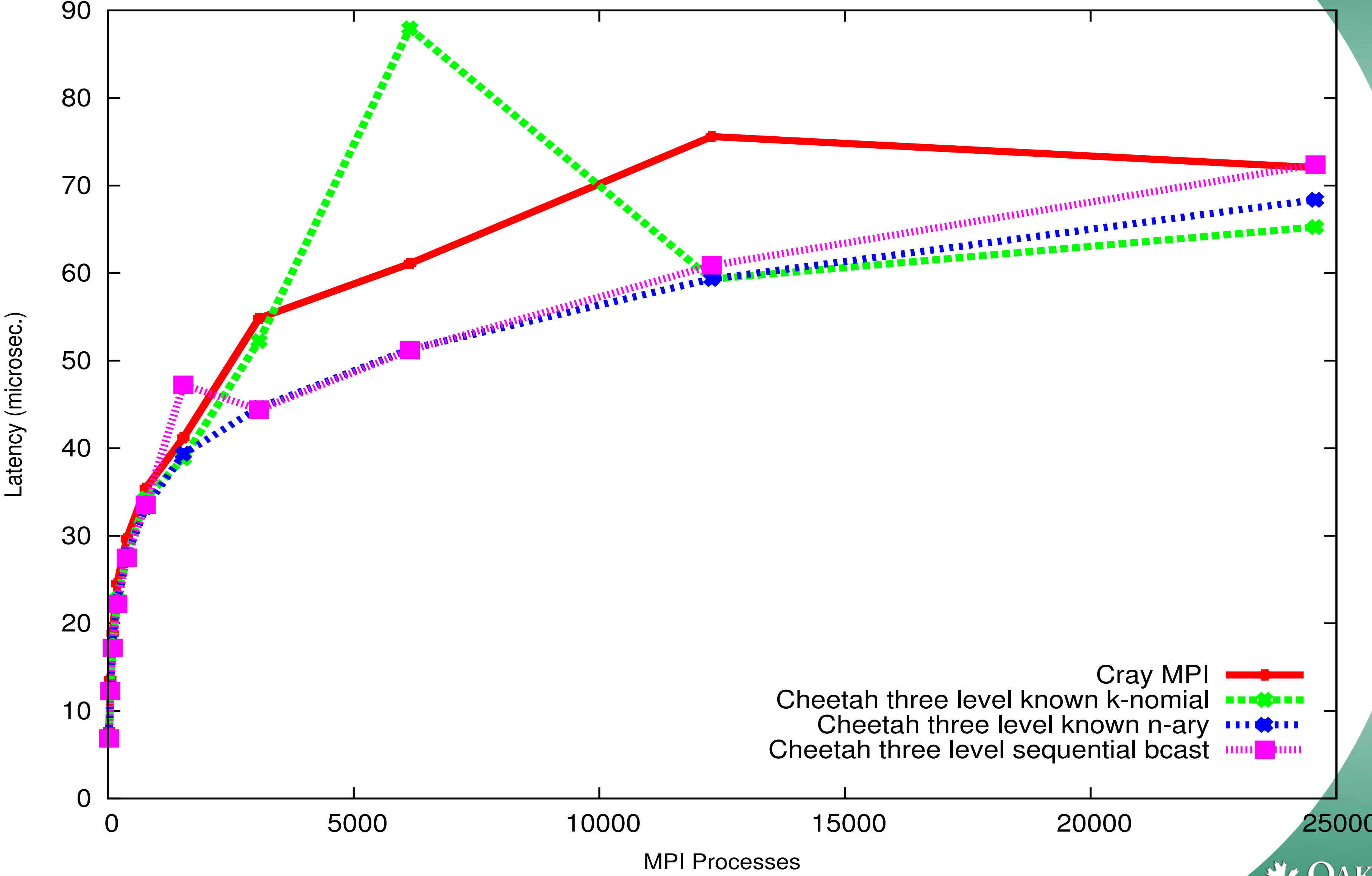
# Data Flow in a Hierarchical Broadcast Algorithm



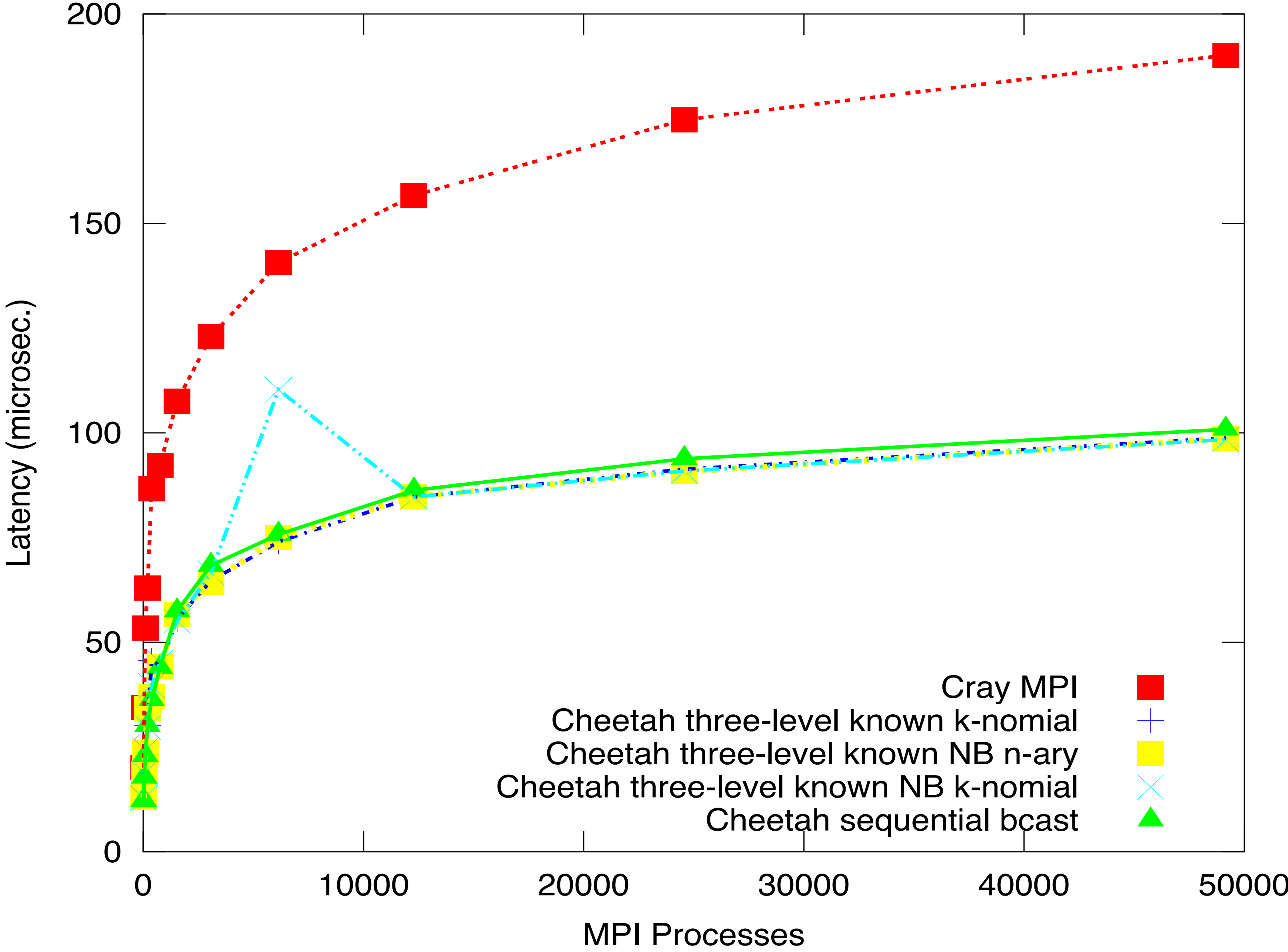
# Hierarchical Broadcast Algorithms

- **Knownroot Hierarchical Broadcast**
  - the suboperations are ordered based on the source of data
  - the suboperations are concurrently started after the execution of suboperation with the source of broadcast
  - uses k-nomial tree for data distribution
- **N-ary Hierarchical Broadcast**
  - same as Knownroot algorithm but uses N-ary tree for data distribution
- **Sequential Hierarchical Broadcast**
  - the suboperations are ordered sequentially
  - there is no concurrent execution

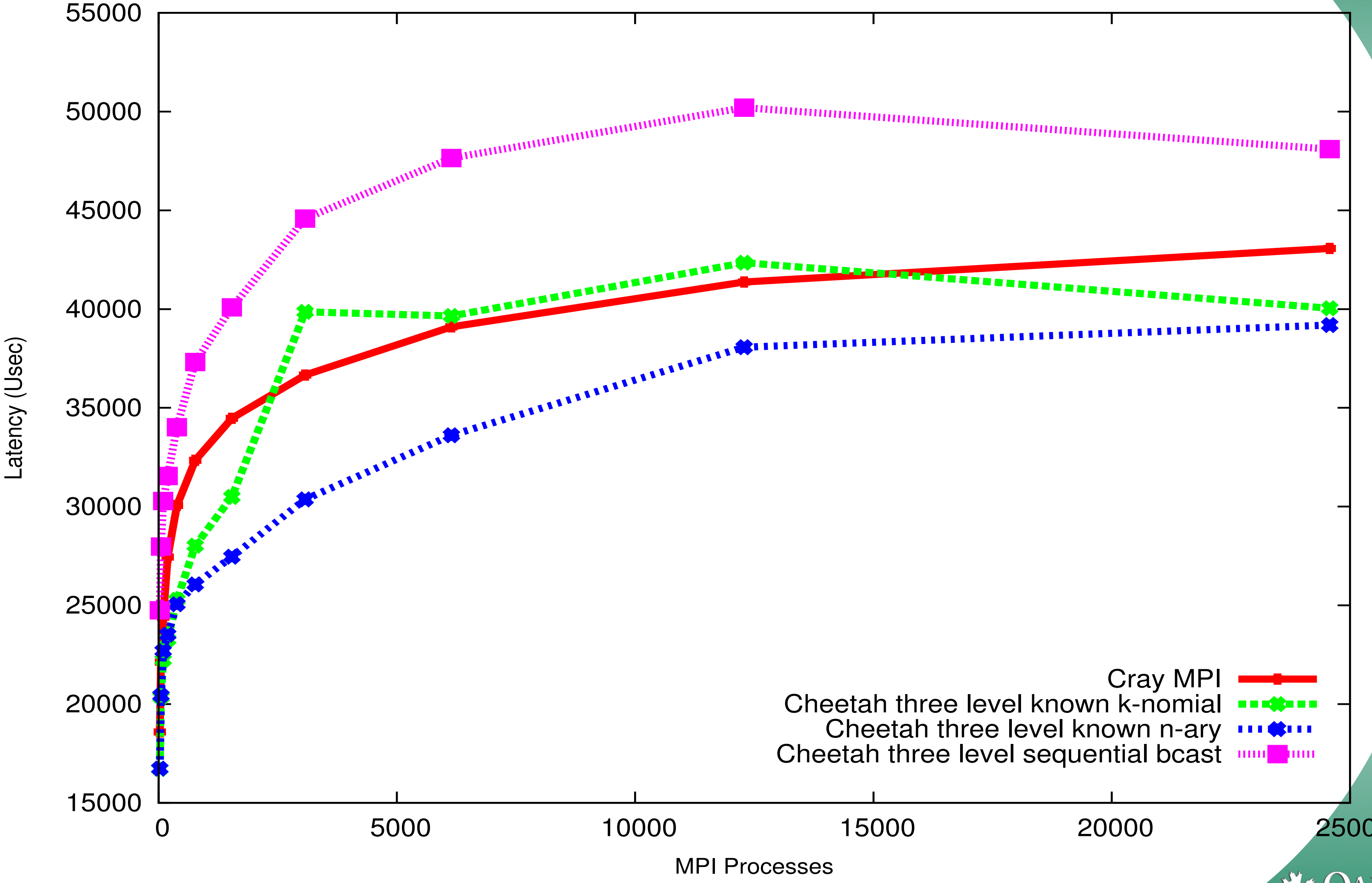
# Cheetah's Broadcast Collective Outperforms the Cray MPI Broadcast by 10% (8 Byte)



# Cheetah's Broadcast Collective Outperforms the Cray MPI Broadcast by 92% (4 KB)



# Cheetah's Broadcast Collective Outperforms the Cray MPI Broadcast by 9% (4 MB)



# Summary

- Cheetah's Broadcast is 92% better than the Cray MPI's Broadcast
- Cheetah's Barrier outperforms Cray MPI's Barrier by 10%
- Open MPI point-to-point message latency is 15% better than the Cray MPI (1 byte message)
- The key to the performance and scalability of the collective operations
  - Concurrent execution of sub-operations
  - Scalable resource usage techniques
  - Asynchronous semantics and progress
  - Customized collective primitives for each of communication hierarchy

# Acknowledgements


- US Department of Energy ASCR FASTOS program
- National Center For Computational Sciences, ORNL