Impact of InfiniBand DC Transport Protocol on Energy Consumption of All-to-all Collective Algorithms

H. Subramoni[1], A. Venkatesh[1], K. Hamidouche[1],
K. Tomko[2] and D. K. Panda[1]

[1] Department of Computer Science and Engineering
The Ohio State University

[2] Ohio Supercomputing Center
Columbus, Ohio

Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
- Conclusions and Future Work

Current Trends in HPC

- Supercomputing systems scaling rapidly
  - Multi-core architectures and
  - High-performance interconnects
- InfiniBand is a popular HPC interconnect
  - 257 systems (51.4%) in Jun’15 Top500
- Message Passing Interface (MPI) used by vast majority of HPC applications
- MPI collective operations very popular due to ease of use and performance portability

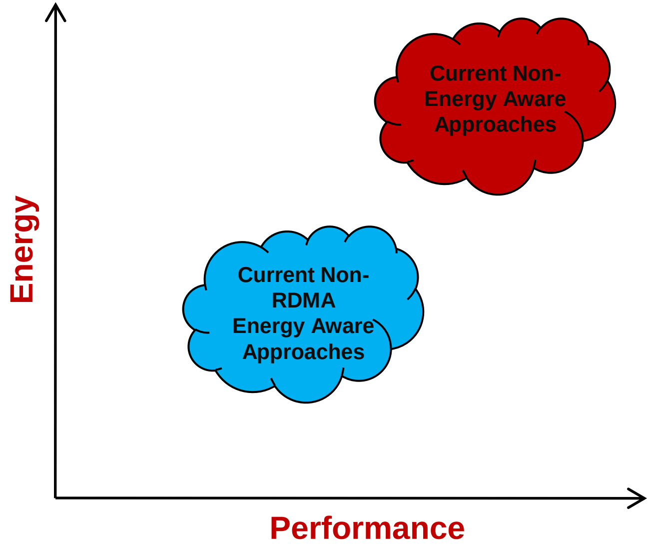
Stampede@TACC

SuperMUC@LRZ

Nebulae@NSCS

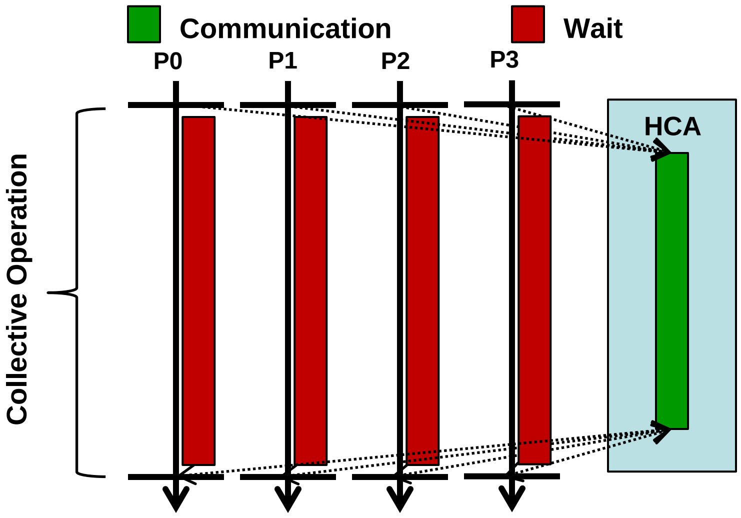
Energy Aware Collective Design

- Non-energy aware approaches to collective design are prevalent
- Current Non-RDMA based Energy-Aware approaches sub-optimal
  - Reduced performance
  - Room to obtain more energy savings
- Most (if not all) of the existing “white-box” approaches to fine-grained energy savings are dependent of throttling of CPUs using DVFS
  - Needs super user privileges
  - Not practical on shared HPC systems



RDMA-Aware Design of Blocking Collectives

- Several attempts to create RDMA-Aware designs for blocking collectives
  - Gupta et al., Sur et al
- Different methods available for progress
  - Basic RDMA schemes
    - Uses basic RDMA operations
      - RDMA_Write / RDMA_Read
  - Dedicated hardware progress engines
    - e.g.: CORE-Direct from Mellanox
    - Venkata et al., Kandalla et al.



CORE-Direct / Basic RDMA

Metric	Naive RDMA RC-Based	CORE-Direct	???
Communication Latency	Good	Fair	Good
Network Scalability	Fair	Fair	Good
Preventing Network Congestion	Poor	Good	Good

Can Modern Transport Protocols Help?

- IB offers several communication protocols with different performance and memory characteristics
  - Reliable Connection (RC)
  - eXtended Reliable Connection (XRC)
  - Unreliable Datagram (UD)
  - Dynamic Connected (DC)

Metric	RC	XRC	UD	DC
Network Scalability	Fair	Good	Very Good	Very Good
Memory Scalability	Fair	Good	Very Good	Very Good
RDMA Support	Yes	Yes	No	Yes

- No work explores how to design efficient blocking collective operations using RDMA primitives on top of different transport protocols for
  - Reducing energy consumption and
  - Achieving good communication latency

Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
- Conclusions and Future Work

Problem Statement

- Can RDMA primitives in conjunction with modern transport protocols be used to design efficient collective operations with the following characteristics
  - Good communication latency
  - Good network scalability
  - Limited network congestion and
  - Good energy footprint

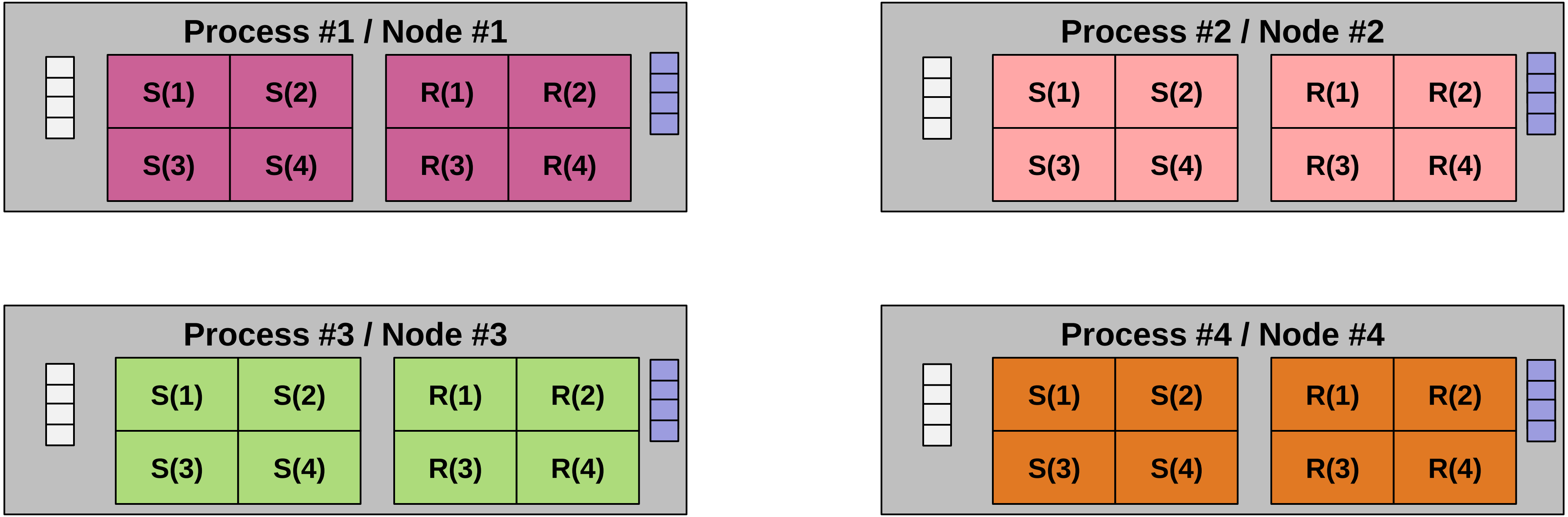
Contributions

- Investigate transport protocol and energy-aware designs for blocking All-to-all collectives for IB networks
- Identify the correct set of transport protocols and algorithms that lead to best energy savings for different All-to-all communication patterns
- Perform a careful analysis of the benefits of our approaches with
  - OSU microbenchmarks
  - NAS parallel benchmarks and
  - P3DFFT application kernel

Outline

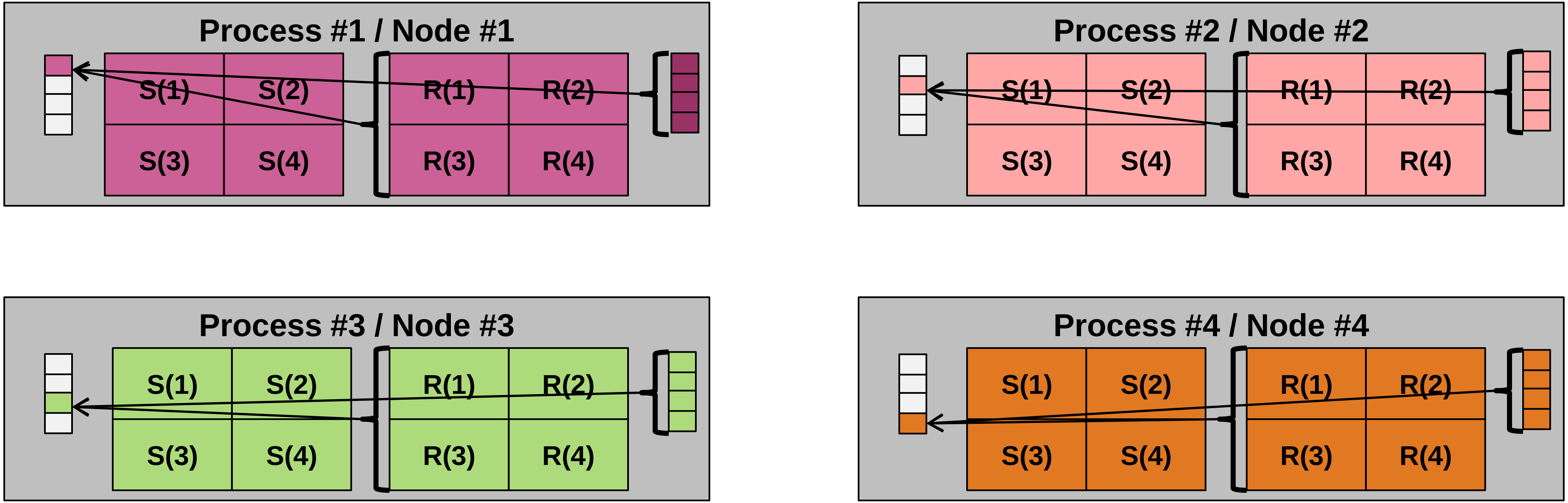
- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
- Conclusions and Future Work

Design of RDMA-Aware All-to-all



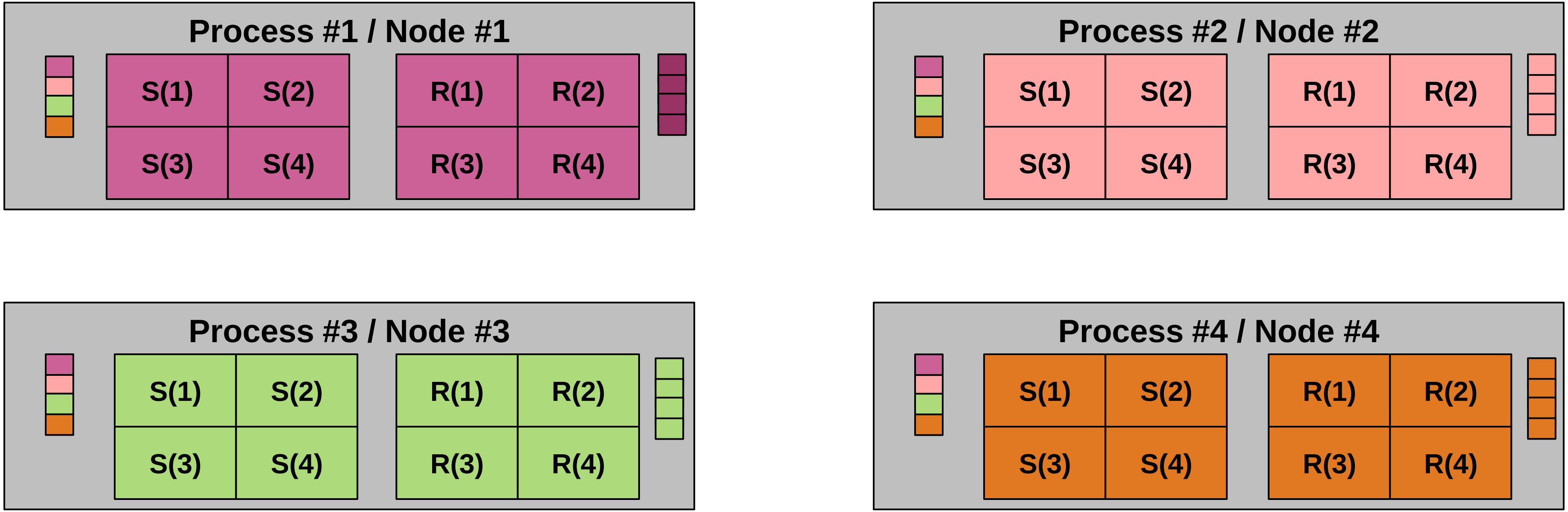
- Receive send/receive buffer information from application
- Allocate temporary buffers to
  - Receive completion notification from remote processes and (size = 1 byte per process in job)
  - Store IB registration and address information from all processes

Register Receive / Completion Buffers



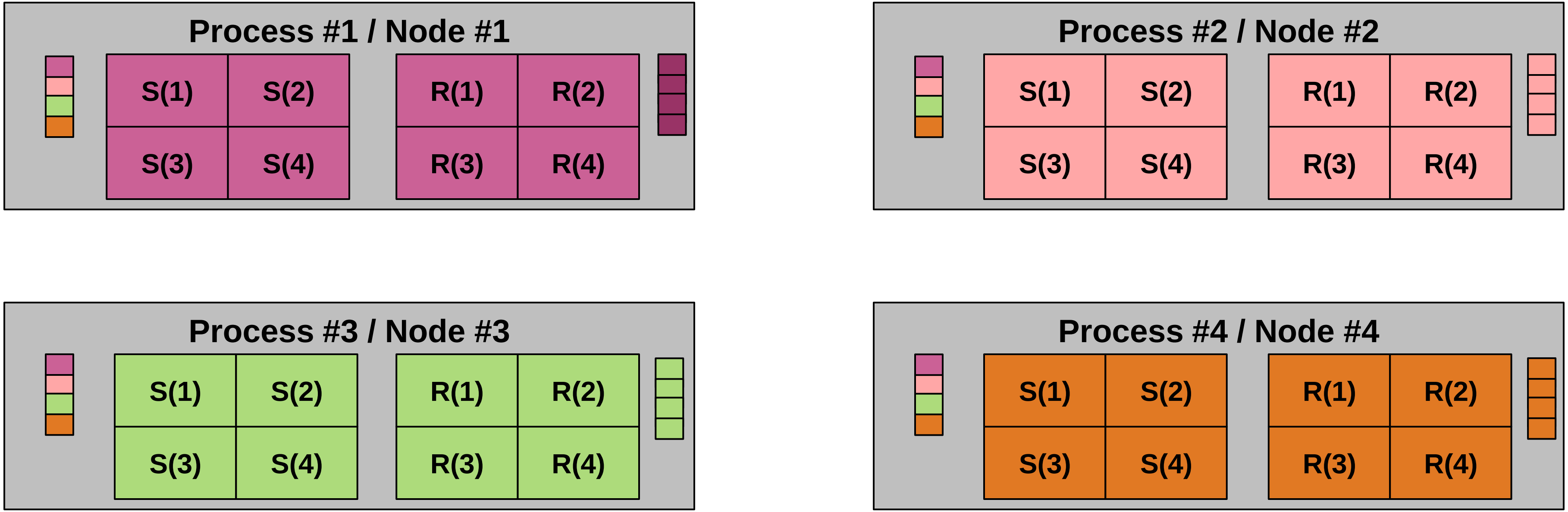
- Register Send / Receive / Completion buffer with IB HCA
- Store IB registration info and address for Receive / Completion buffers

Exchange memory / rkey Information



- Perform MPI_Allgather (24 bytes) and collect
  - IB registration information and
  - Receive / completion buffer address from all processes

Exchange Data / Notify Completion



- Initiate RDMA_Write operations to remote processes using information collected in Allgather
  - Place data and
  - Notify completion

Other Design Considerations

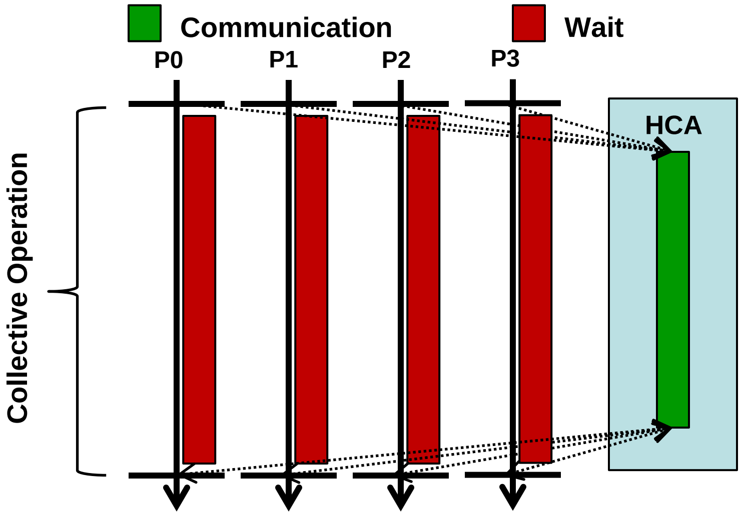
- Caching Mechanism to Avoid MPI_Allgather
  - Cache <target memory address, rkey> from all processes
  - Compare <memory address, rkey> of current invocation with cached value
  - Perform MPI_Allreduce with MPI_LAND on result of comparison
  - MPI_Allreduce significantly less expensive and scalable
- RDMA_Write vs RDMA_Read
  - Throughput of RDMA_Write higher than RDMA_Read
    - Possibly due to limitation on the number of back-to-back RDMA_Read operations that can be posted to IB HCA
- Temporary Memory Overhead
  - Memory overhead negligible
    - Consume about 3.0 MB of memory per process for an All-to-all of any message size involving 131,072 (128 K) processes

Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
- Conclusions and Future Work

Design Goals & Challenges

- Design Goals
  - Good communication latency
  - Good network scalability
  - Limited network congestion
  - Good energy footprint
- Design Challenges
  - Can we accurately identify the time processor needs to be in low energy state?
  - How can processor be forced into a low energy state for a specified duration?
  - Can intelligent use of modern transport protocols aid the design of efficient energy aware All-to-all collective algorithms?



CORE-Direct / RDMA Based

Estimating Communication Time

- Heuristics
  - Use one-way latency and number of transfers expected with All-to-all
  - Maintain internal communication latency tables
    - Tables maintained for a range of message sizes for different systems
- Log(GP) model[1]
- Application can tell the MPI library through MPIT
  - Requires application changes
- Profile the time taken for the All-to-all operation
  - Done on a per communicator basis
  - Found to be more accurate

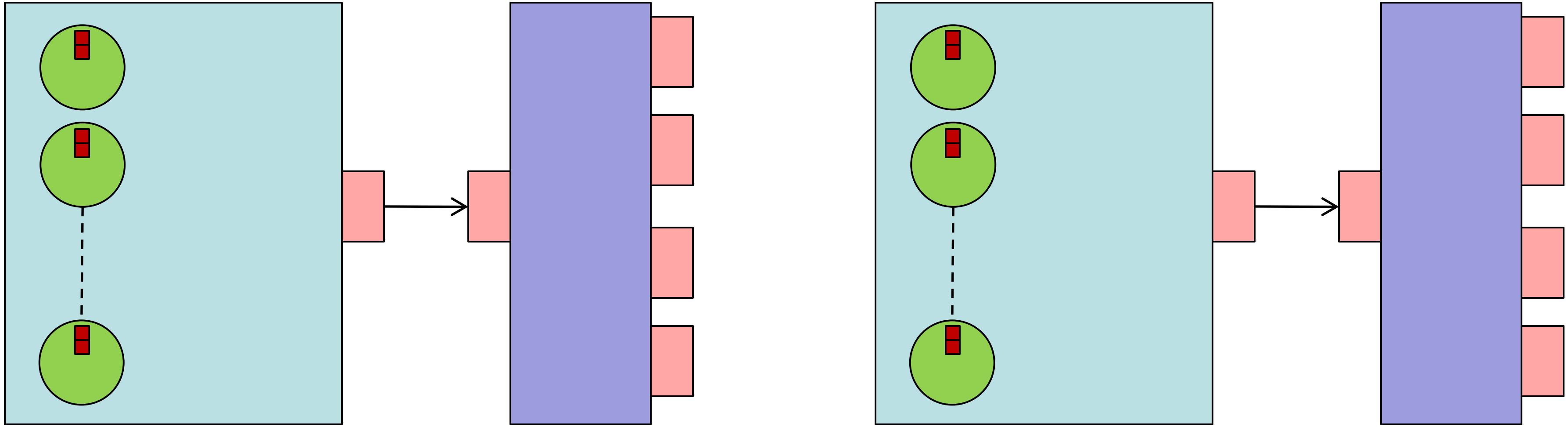
[1] Alexandrov et al.; LogGP: incorporating long messages into the LogP model—one step closer towards a realistic model for parallel computation; Proceedings of the seventh annual ACM symposium on Parallel algorithms and architectures (SPAA'95).

Moving Processor to Low Power State

- Use RAPL interface
  - Requires elevated (super-user / root) privileges
  - Not practical on shared HPC systems
- Rely on the Linux kernel
  - Kernel smart enough to move the processors to a low energy state if cores are idle
  - Ensure that the MPI process is idle
  - Multiple options
    - Enter interrupt based progress mode
      - Allows for progress of other communication
      - Cannot support multi-channel (shared memory / IB) communication
    - Call “usleep” for estimated communication time
      - Supports multi-channel (shared memory / IB) communication
      - Cannot progress other communication
        - » All-to-all is very communication intensive
        - » May be better to avoid other communication during this time

Behavior of RC & DC with Low Communication Load

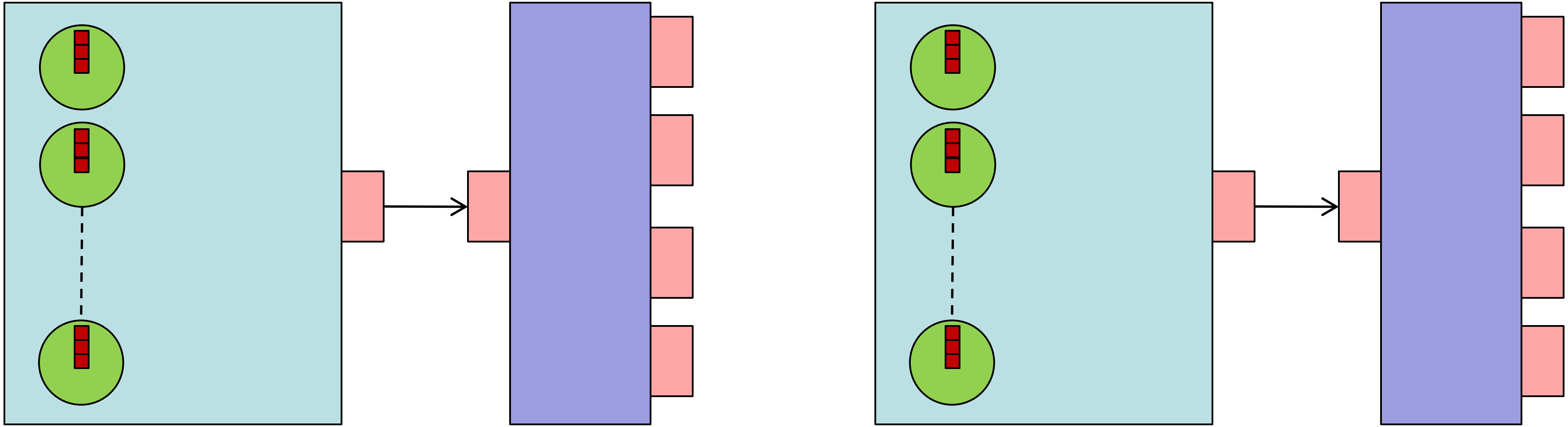
Reliable Connected **Dynamic Connected**



- RC delivers excellent performance
- Inherent serialization in DC results in slightly reduced performance

Behavior of RC & DC with Medium Communication Load

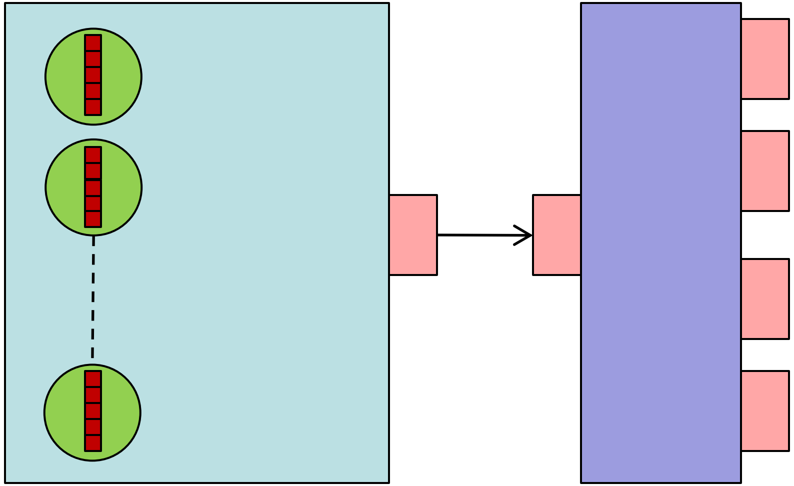
Reliable Connected | **Dynamic Connected**



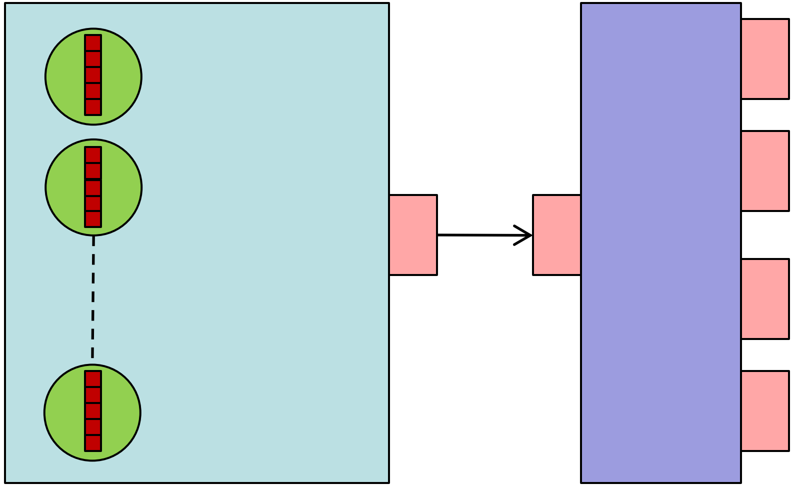
- Multiple concurrent operations in RC results in slightly reduced performance
  - QP cache trashing
  - Performance still equivalent to DC
- Inherent serialization in DC results in good network behavior

Behavior of RC & DC with High Communication Load

Reliable Connected



Dynamic Connected



- Multiple concurrent operations in RC significantly reduced performance
  - QP cache trashing
- Inherent serialization in DC results in good network behavior

Intelligent Protocol Selection

- RC Protocol
  - Best performance at low to medium network load
  - Performance degrades as network load increases
  - Choose for applications / communication patterns with low to medium network load
- DC Protocol
  - Inherent serialization in DC causes
    - Performance overhead at low to medium network load
    - Good network behavior at high network load
  - Choose for applications / communication patterns with high network load

Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
  - Microbenchmark Level Evaluation
  - Evaluation with Application Kernels: NAS & P3DFFT
- Conclusions and Future Work

Experimental Setup

- 32 Node Intel Ivybridge cluster
- Each node equipped with
  - Intel Ivybridge dual ten-core sockets
  - 2.80 GHz with 32GB RAM
  - MT4113 FDR ConnectIB HCAs (56 Gbps data rate)
  - PCI-Ex Gen3 interfaces.
  - RHEL release 6.2, with kernel version 2.6.32-220.el6
  - Mellanox OpenFabrics version 2.4-1.0.4
- Evaluations done with
  - OSU Microbenchmarks v5.0
  - NAS Parallel Benchmarks v3.3
  - P3DFFT Kernel with
    - “-DUSE EVEN” build option; use MPI_Alltoall instead of MPI_Alltoallv
    - Weak scaling experiments; problem size increases with job size
    - Problem size configured to take 75% - 80% of total system memory

MVAPICH2 Software

- **High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RoCE**
  - MVAPICH (MPI-1) , Available since 2002
  - MVAPICH2 (MPI-2.2, MPI-3.0 and MPI-3.1), Available since 2004
  - MVAPICH2-X (Advanced MPI + PGAS), Available since 2012
  - Support for GPGPUs (MVAPICH2-GDR), Available since 2014
  - Support for MIC (MVAPICH2-MIC), Available since 2014
  - Support for Virtualization (MVAPICH2-Virt), Available since 2015
  - Used by more than 2,450 organizations in 76 countries
  - More than 281,000 downloads from the OSU site directly
  - Empowering many TOP500 clusters (Jun'15 ranking)
    - 8th ranked 519,640-core cluster (Stampede) at TACC
    - 11th ranked 185,344-core cluster (Pleiades) at NASA
    - 22nd ranked 76,032-core cluster (Tsubame 2.5) at Tokyo Institute of Technology and many others
  - Available with software stacks of many IB, HSE, and server vendors including Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu
- **Empowering Top500 systems for over a decade**
  - System-X from Virginia Tech (3rd in Nov 2003, 2,200 processors, 12.25 TFlops) ->
  - Stampede at TACC (8th in Jun'15, 462,462 cores, 5.168 Plops)

Designs Used for Performance Evaluation

- All-to-all Algorithm
  - Default
    - Default implementation of blocking All-to-all collective (uses pair-wise algorithm)
  - R-Aware
    - The RDMA-Aware scheme proposed in [1] adapted for blocking collectives
  - R-P-Aware
    - The RDMA-Aware scheme with designs to move processor to lower energy state
- IB Transport Protocol
  - RC
    - The standard RC transport protocol of IB
  - DC
    - The DC transport protocol of IB with the DCPool design described in [2]
    - Uses a pool of DC QPs for communication
  - DC-E-UD
    - The DC transport protocol of IB with the DC-E-UD described in [2]
    - Uses only one DC QP for communication

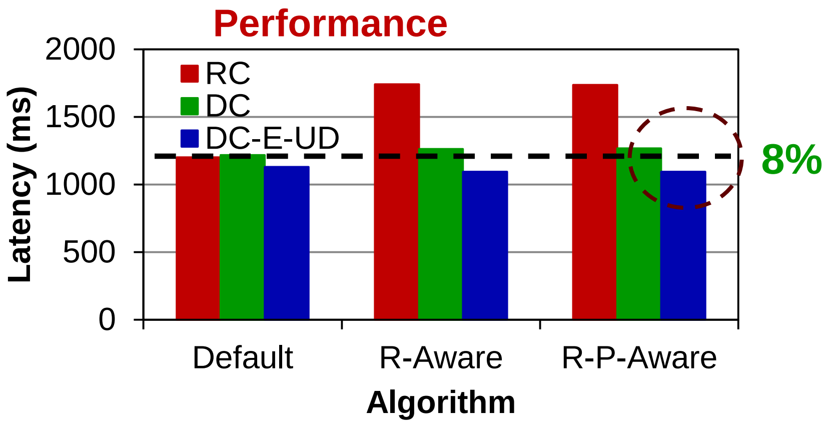
[1] Designing Non-Blocking Personalized Collectives with Near Perfect Overlap for RDMA-Enabled Clusters; H. Subramoni, A. Awan, K. Hamidouche, D. Pekurovsky, A. Venkatesh, S. Chakraborty, K. Tomko, and D. K. Panda; ISC '15, Jul 2015

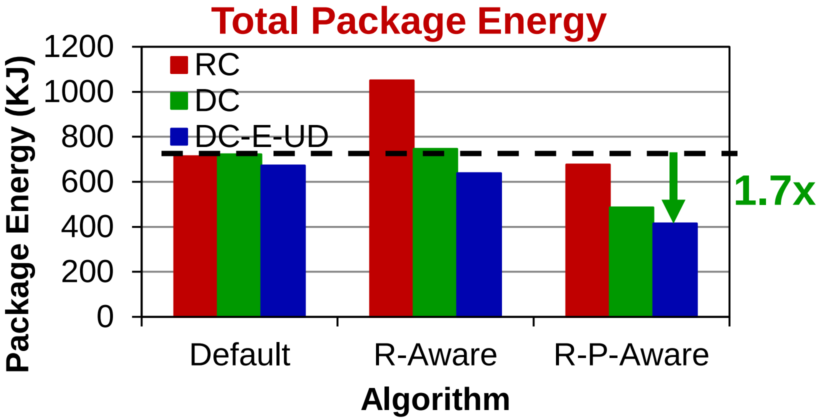
[2] Designing MPI Library with Dynamic Connected Transport (DCT) of InfiniBand : Early Experiences; H. Subramoni, K. Hamidouche, A. Venkatesh, S. Chakraborty, and D. K. Panda; IEEE International Supercomputing Conference (ISC '14), Jun 2014

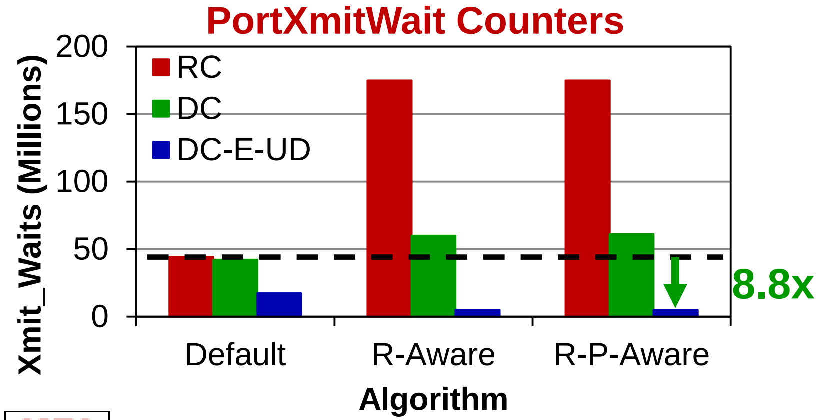
Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
  - Microbenchmark Level Evaluation
  - Evaluation with Application Kernels: NAS & P3DFFT
- Conclusions and Future Work

512 KB Global All-to-all at 640 Processes

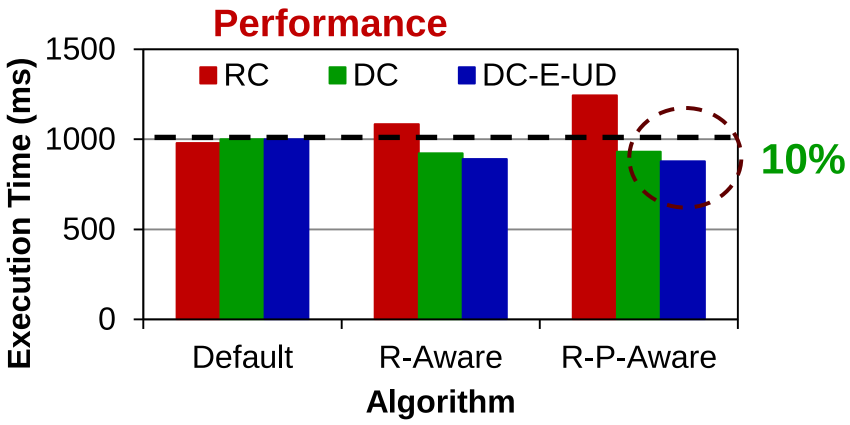


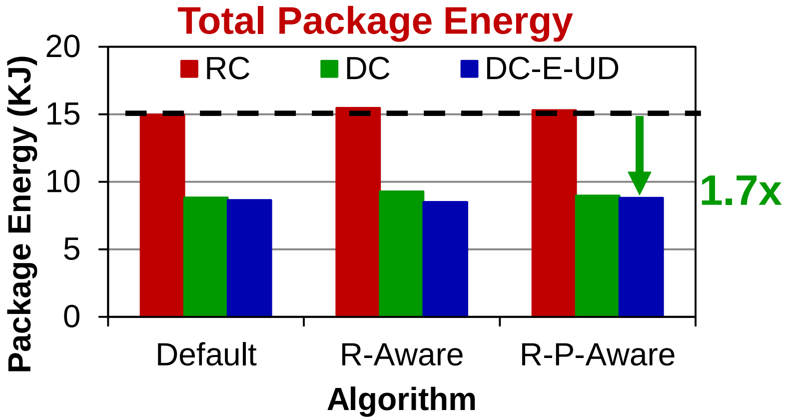


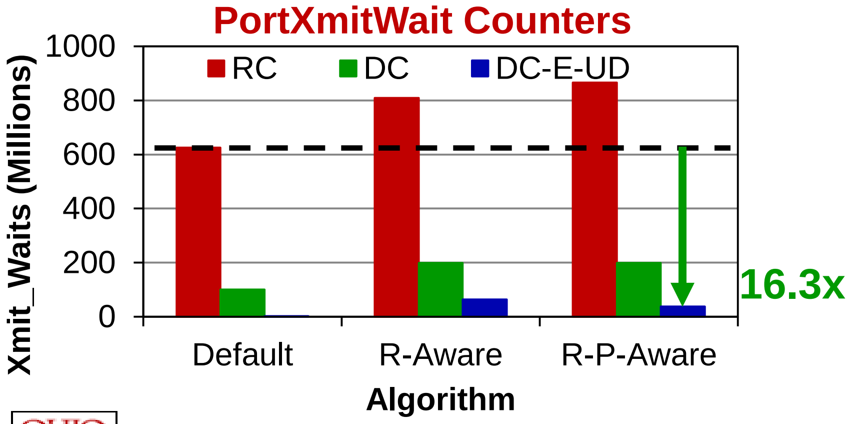


- R-P-Aware + DC-E-UD
  - Significant energy savings
    - Able to save 1.7x (44%) energy
  - Improves communication performance
    - 8% improvement in latency
  - Significant reduction in network congestion
    - 8.8 times reduction in congestion

Class C NAS FT @ 512 Processes

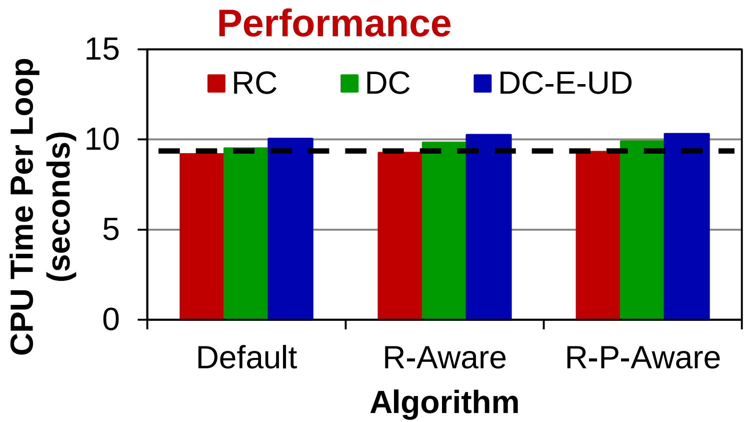


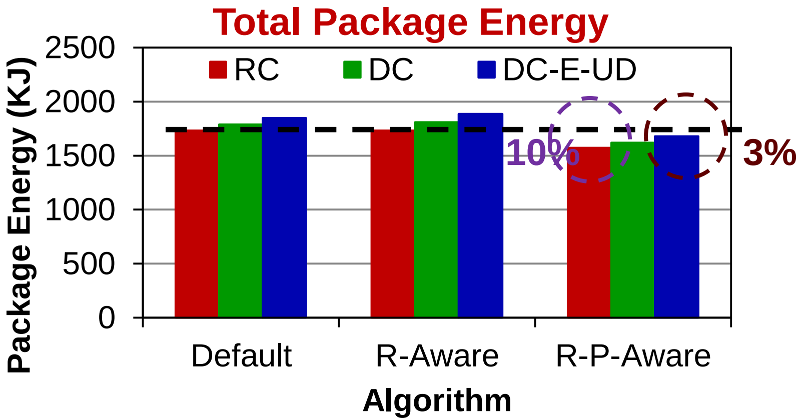


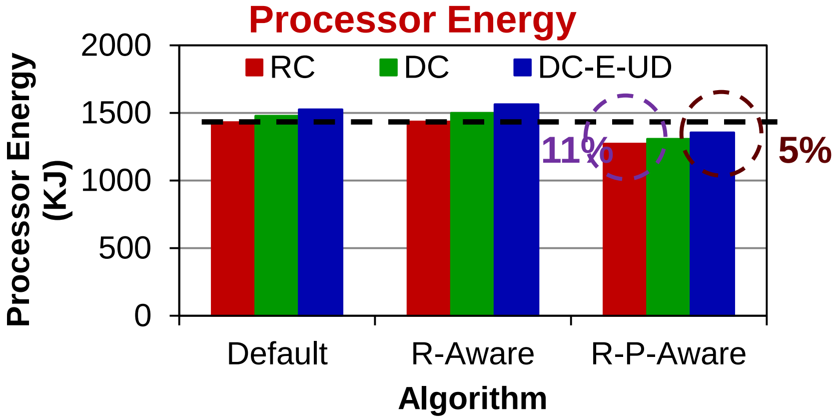


- R-P-Aware + DC-E-UD
  - Significant energy savings
    - Able to save 1.7x (44%) energy
  - Improves communication performance
    - 10% improvement in execution time
  - Significant reduction in network congestion
    - 16.3 times reduction in congestion
- Default + DC-E-UD
  - Best for reducing congestion
  - Incurs slight performance penalty

P3DFFT Kernel @ 640 Processes







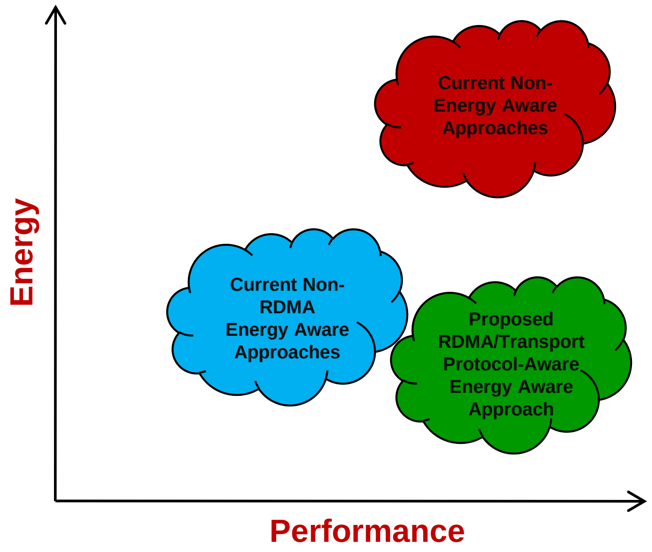
- P3DFFT performs row / column All-to-all
- Less dense than global All-to-all
- RC expected to perform best
- R-P-Aware + RC
  - Performance similar to Default + RC
  - Energy savings
    - Able to save 10% energy
- Algorithms using DC & DC-E-UD
  - Incurs performance hit due to serialization

Outline

- Introduction
- Problem Statement & Contributions
- Background
- Design of Efficient Transport Protocol and Energy Aware RDMA Based All-to-all Algorithms
- Performance Evaluation
- Conclusions and Future Work

Conclusions

- Studied the impact of RDMA and transport protocol aware designs on the energy and performance of dense collective operations like All-to-all
- Proposed transport protocol / energy-aware designs for blocking All-to-all
- Demonstrated drawbacks in using single transport protocol for different applications / communication patterns
- Identify the correct set of transport protocols and algorithms that lead to energy savings for different All-to-all communication patterns
- Proposed approach improves energy efficiency by
  - **1.7 times** for large message MPI_Alltoall at 640 processes
  - **1.7 times** for Class C NAS FT benchmark at 512 processes
  - **10%** for P3DFFT kernel at 640 processes



Future Work

- Study the impact transport protocol and energy aware designs can have on other collective communication patterns like All-to-one and One-to-all
- Evaluate advanced All-to-all algorithm designs to avoid network congestion with RC protocol
- Evaluate the impact of proposed algorithms on other RDMA-enabled networks like RoCE
- Distribute RDMA / energy aware designs with future releases of MVAPICH2

Thank you!

{subramon, akshay, hamidouc, panda}
@cse.ohio-state.edu
ktomko@osc.edu

Network-Based Computing Laboratory
http://mvapich.cse.ohio-state.edu/

Network Based Computing Laboratory

OHIO STATE

MVAPICH

Ohio Supercomputer Center