High-Performance Coarray Fortran Support with MVAPICH2-X: Initial Experience and Evaluation

Jian Lin, Khaled Hamidouche, Xiaoyi Lu, Mingzhe Li, Dhabaleswar K. (DK) Panda

Presenter: Wasiur Rahman

Network-Based Computing Laboratory
Department of Computer Science and Engineering
The Ohio State University, USA

Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

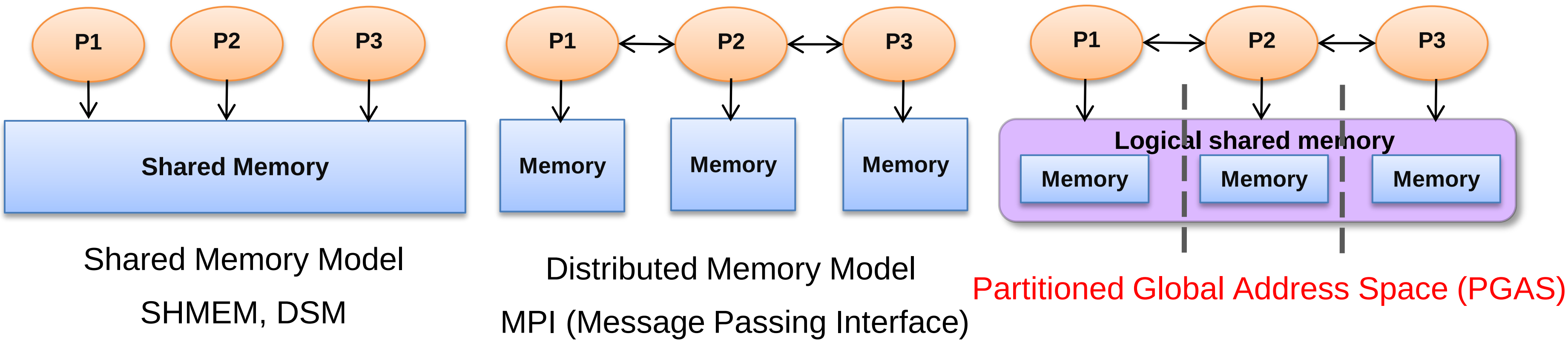
Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

Introduction

- MPI is the de-facto programming model for scientific parallel applications
  - Offers attractive features for High Performance Computing (HPC) applications
  - MPI Libraries (such as MVAPICH2, Open MPI, Intel MPI) have been optimized to the hilt
- Partitioned Global Address Space (PGAS) models are Emerging
  - Global view of data, One sided operations, better programmability
  - Suits for irregular and dynamic applications

Partitioned Global Address Space (PGAS) Models



- Key abstraction
  - Shared memory abstraction over distributed system images
- Library-level solutions
  - OpenSHMEM
  - Global Arrays
  - …
- Language-level solutions
  - UPC
  - Coarray Fortran (CAF)
  - …

Coarray Fortran (CAF): Language-level PGAS support in Fortran

- An extension to Fortran to support global shared array in parallel Fortran applications
- CAF = CAF compiler + CAF runtime (libcaf)

- Coarray syntax and basic synchronization support in Fortran 2008

```
E.g.
real :: a(n)[*]
x(:)[q] = x(:) + x(:)[p]
name[i] = name
sync all
...
```

- Collective communication and atomic operations in upcoming Fortran 2015

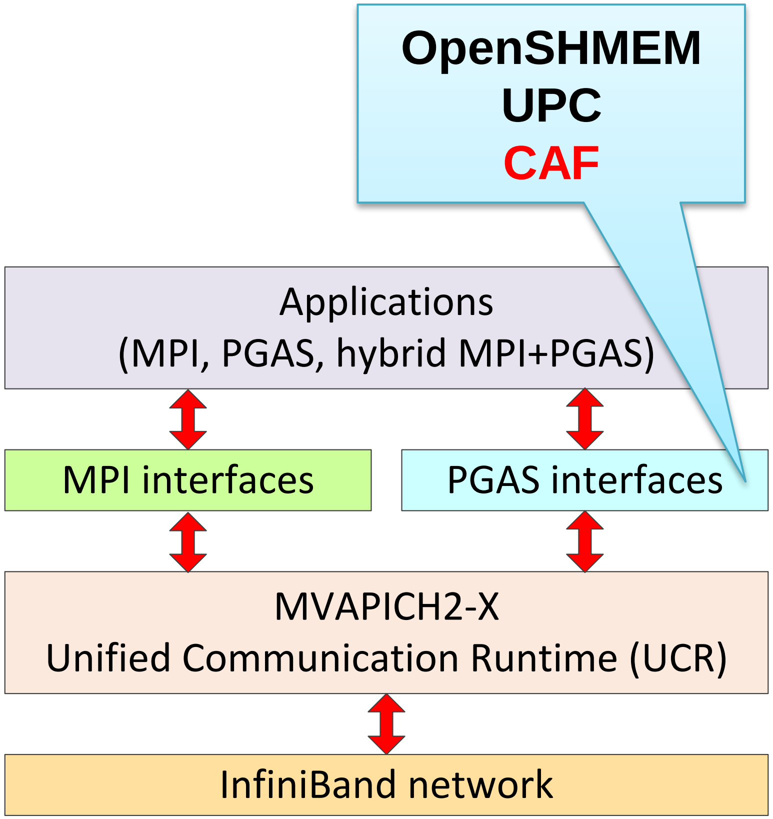
interface	parameters
CO_BROADCAST	A, SOURCE_IMAGE [, STAT, ERRMSG]
CO_MAX	A [, RESULT_IMAGE, STAT, ERRMSG]
CO_MIN	A [, RESULT_IMAGE, STAT, ERRMSG]
CO_SUM	A [, RESULT_IMAGE, STAT, ERRMSG]
CO_REDUCE	A, OPERATOR [, RESULT_IMAGE, STAT, ERRMSG]

```
ATOMIC_ADD
ATOMIC_FETCH_ADD
...
```

MVAPICH2/MVAPICH2-X Software

- High Performance open-source MPI Library for InfiniBand, 10Gig/iWARP, and RDMA over Converged Enhanced Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.0), Available since 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2012
  - Support for GPGPUs and MIC
  - **Used by more than 2,375 organizations in 75 countries**
  - **More than 259,000 downloads from OSU site directly**
  - Empowering many TOP500 clusters (November 2014 ranking)
    - 7th ranked 519,640-core cluster (Stampede) at TACC
    - 11th ranked 160,768-core cluster (Pleiades) at NASA
    - 15th 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
- Partner in the U.S. NSF-TACC Stampede System

MVAPICH2-X: Unified Communication Library for MPI and PGAS



The Basic Architecture of MVAPICH2-X

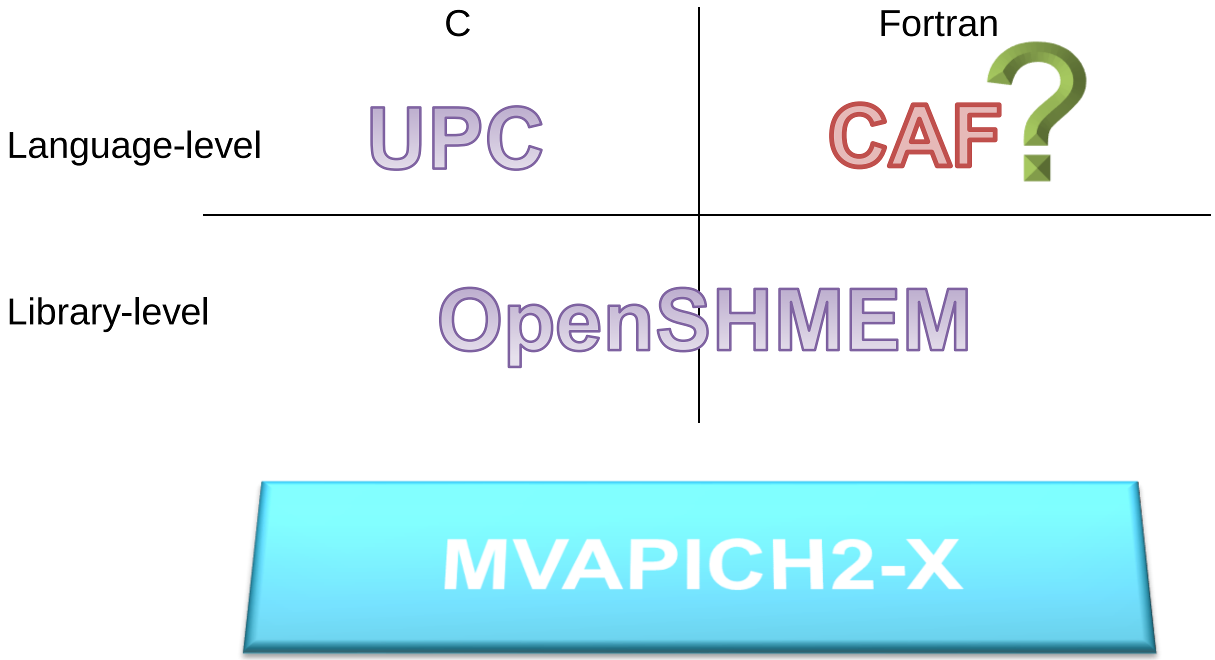
- Feature Highlights
  - Supports MPI(+OpenMP), OpenSHMEM, UPC, MPI(+OpenMP) + OpenSHMEM, MPI(+OpenMP) + UPC
  - MPI-3 compliant, OpenSHMEM v1.0 standard compliant, UPC v1.2 (initial support for UPC v1.3) standard compliant
  - Scalable inter-node communication with high performance and reduced memory footprint
  - Optimized intra-node communication using shared memory schemes
  - Optimized OpenSHMEM collectives
  - Supports different CPU binding policies
  - Flexible process manager support

Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

Motivation

- Can CAF be well supported with MVAPICH2-X to deliver high-performance on InfiniBand clusters?



Challenges

- Can a light-weight and transparent design be proposed for MVAPICH2-X to efficiently support CAF on InfiniBand clusters?

- Can we improve the performance of the new collective operations in CAF through efficiently mapping them onto internal collective operation designs in MVAPICH2-X?

- What are the performance benefits of our proposed approach across various CAF micro-benchmarks and applications with MVAPICH2-X on InfiniBand clusters?

Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

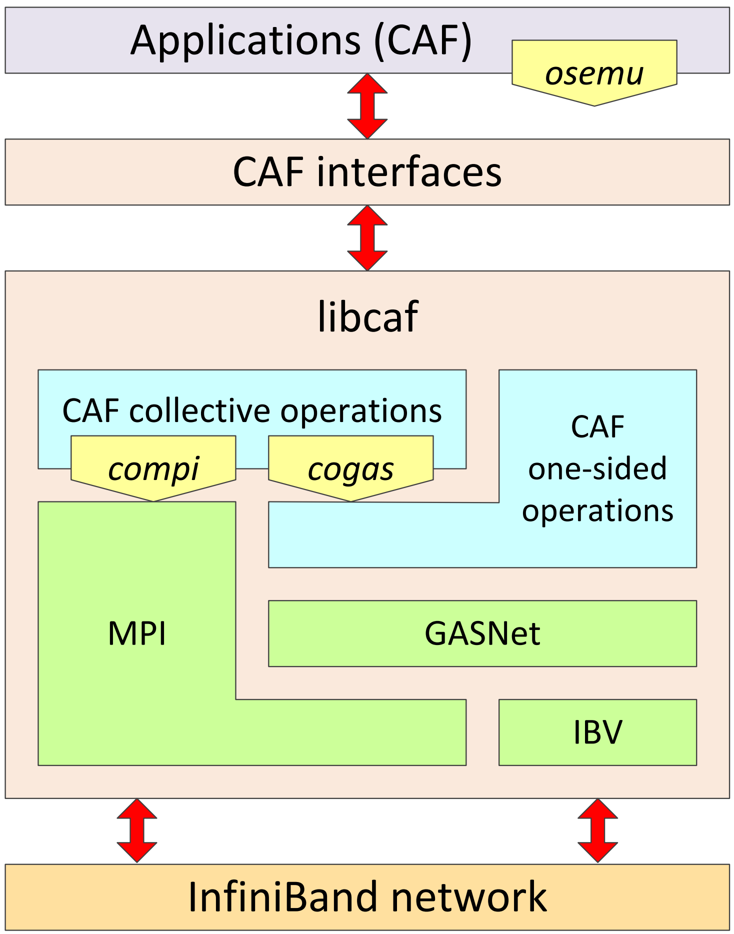
Alternative CAF implementations

- Commercial software
  - Cray Fortran, Intel Fortran, …
- Open-source project
  - OpenUH CAF, GNU Fortran + OpenCoarrays, …

	MPI	GASNet	DMAPP	ARMCI
Cray Fortran	**R**		**C**	
Intel Fortran	**C+R**			
OpenUH CAF	**C+R**	**C+R**		**C**
OpenCoarrays	**C+R**	**C+R**		**C**

C: Communication library   **R**: Runtime management

Communication Design in OpenUH CAF Runtime



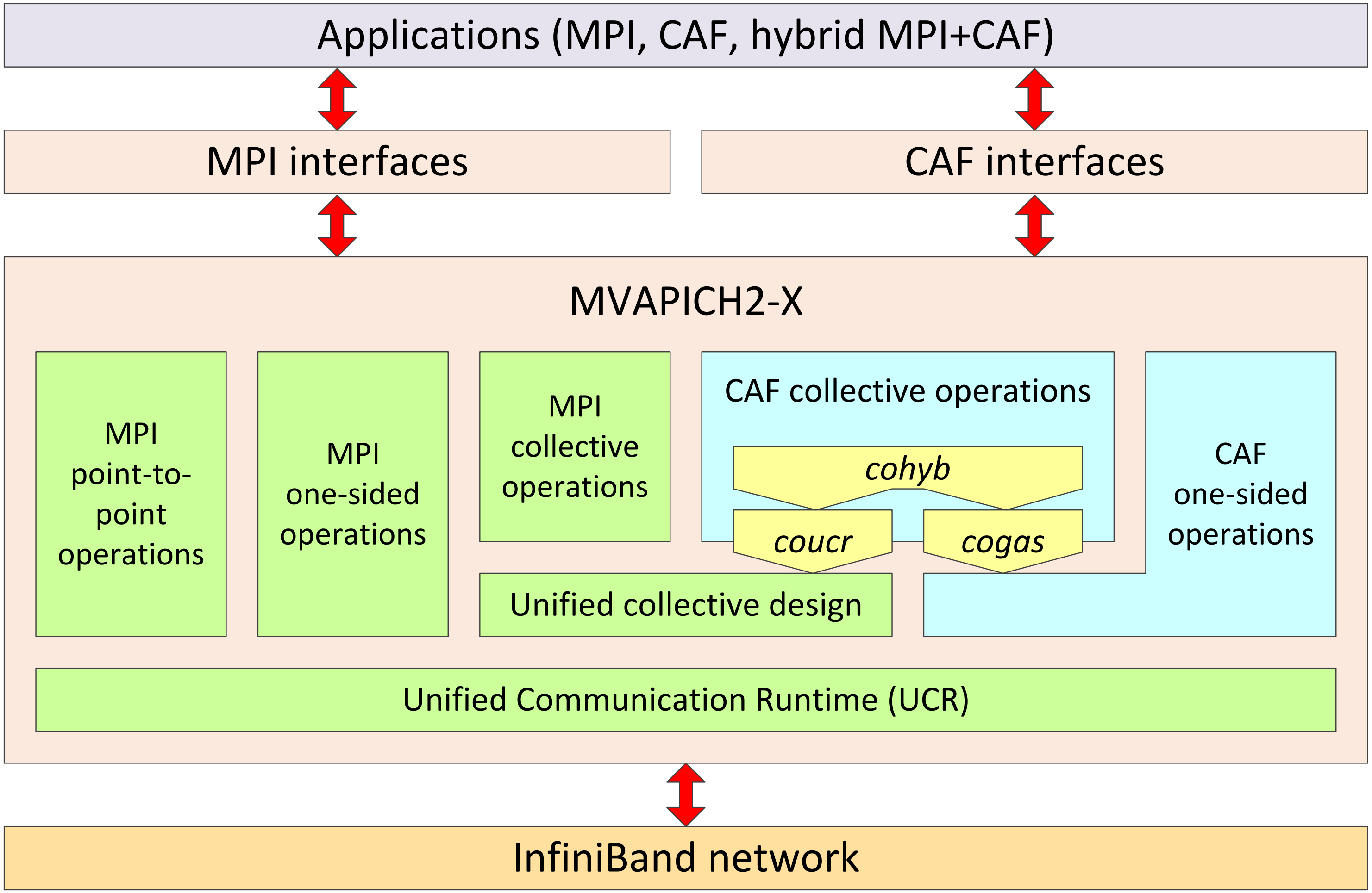
- Why choose OpenUH CAF?
  - Relatively complete support of the new collective communication
- Collective implementations

Name	Description
compi	Calling MPI top-level collective interfaces directly
cogas	Centric or 2-level collective based on non-blocking puts/gets in loop
osemu	Application-level emulation based on one-sided coarray operations

Unified Communication Runtime (UCR) in MVAPICH2-X

- The common communication layer in MVAPICH2-X
  - Lower-level but easy-to-use primitives supporting the common one-sided, two-sided, collective communication and synchronization semantics
  - Optimization for collective communication, shared memory communication, etc.
- UCR-based MVAPICH2-X Conduit for GASNet
  - A complete implementation of GASNet core APIs as well as collective extended APIs
  - Has supported the UPC implementation in MVAPICH2-X

The Overview of Proposed Design



One-sided Operations

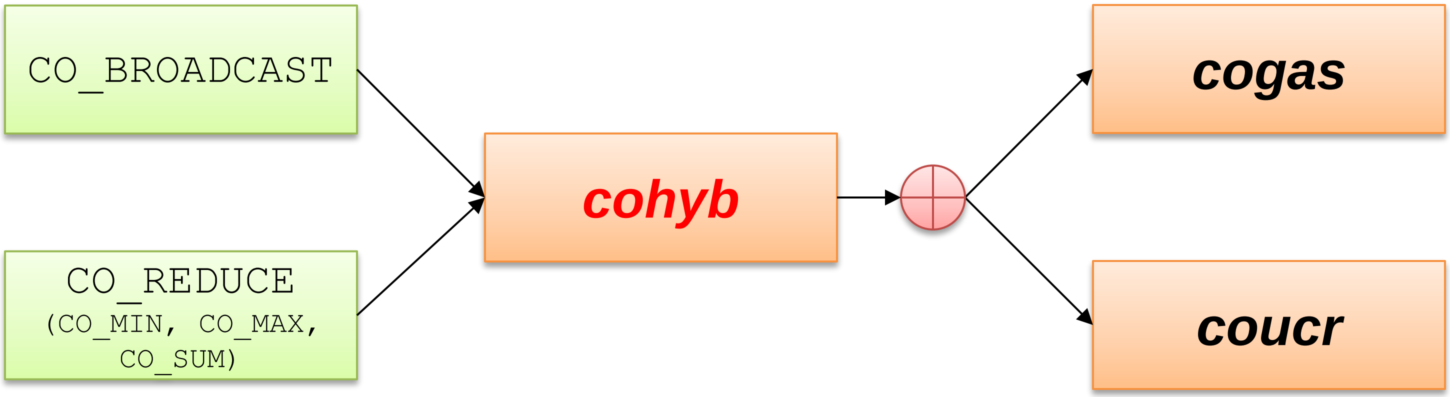
- One-sided coarray access operations can benefit from UCR directly because of MV2X-conduit in GASNet
- Performance optimizations:
  - Small messages: Use pre-registered intermediate buffers to reduce the latency, and increase the bandwidth / message rate
  - Large messages: Use directly the user buffer to read/write the data through RDMA to avoid the overhead of the copies to the intermediate buffers

Collective Operations (1)

- Direct Approach - *compi* and *cogas*
  - *compi* and *cogas* approaches in OpenUH CAF can benefit from UCR directly because of MV2X-conduit in GASNet
  - May introduce additional overhead into the call stacks
- Enhanced Approach - *coucr*
  - Integrate with UCR in depth, bypass the redundant GASNet call stacks and avoid the single-operation-in-loop
    - CO_BROADCAST → UCR broadcast
    - CO_REDUCE → UCR reduce/allreduce
    - Other reduce interfaces → operator mapping logic + UCR reduce/allreduce

Collective Operations (2)

- Hybrid Approach - *cohyb*
  - *cogas* and *coucr* may use different algorithms for the same collective operation
  - No one design can perform better at all message sizes.
  - Switching thresholds are set for different collective subroutines according to the tuning parameters



Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

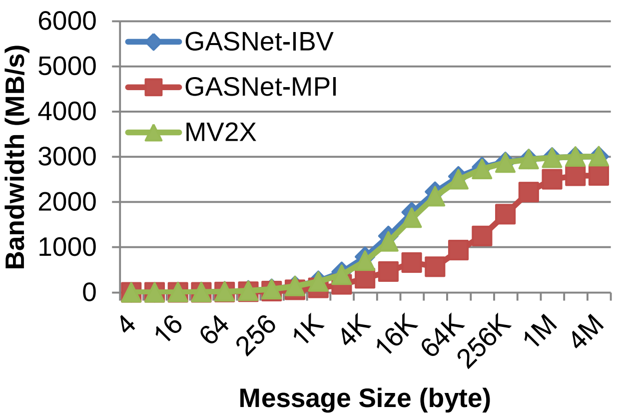
Experiment Setup

- Hardware: RI Cluster @ CSE, OSU
  - Xeon dual 8 core sockets (2.67GHz) with 12GB RAM
  - Mellanox QDR ConnectX HCAs (32 Gbps data rate) with PCI-Ex Gen2 interfaces
- Software stack
  - RHEL 6.3 with OpenFabrics 1.5.3-3
  - MVAPICH2-X 2.1rc2 + OpenUH CAF 3.0.39
- Benchmarks
  - CAF Test Suite from University of Houston (with our modifications for testing collective operations)
  - NPB3.3-CAF from University of Houston

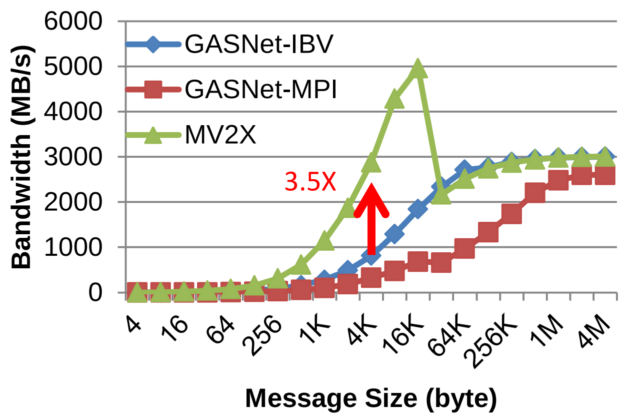
Performance Evaluations for One-sided Communication

NETWORK-BASED COMPUTING LABORATORY

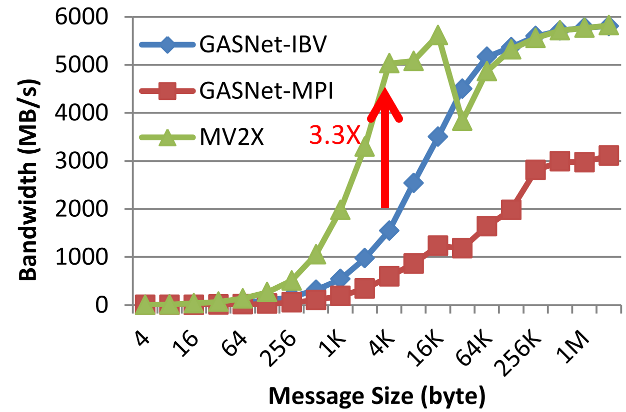
Get

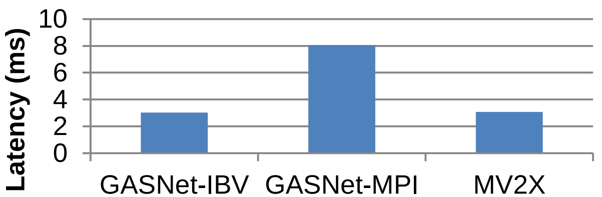


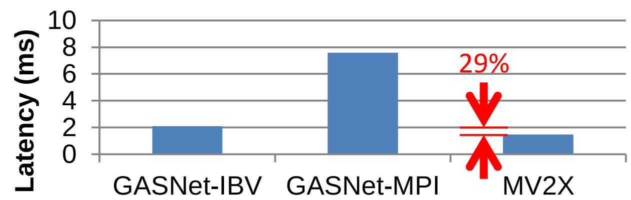
Put



Bidirectional

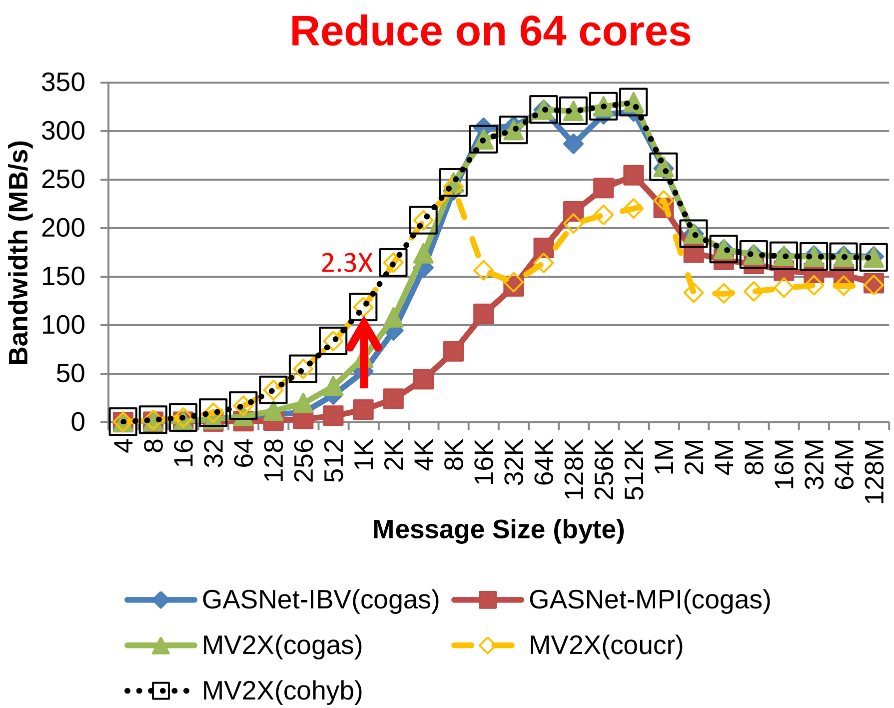


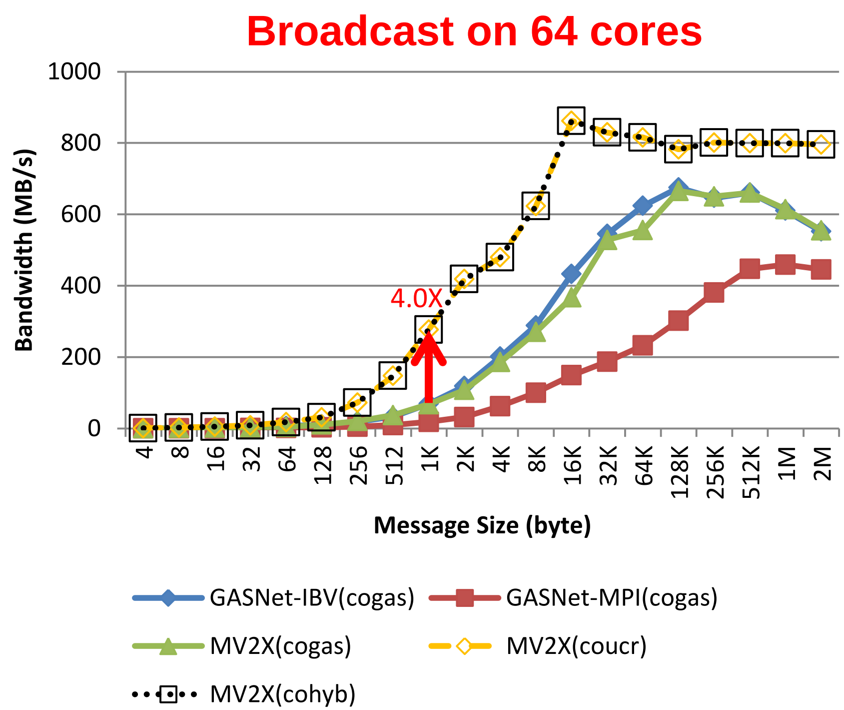




- Micro-benchmark improvement (MV2X vs. GASNet-IBV)
  - Put bandwidth: 3.5X improvement on 4KB; Put latency: reduce 29% on 4B
  - Bidirectional bandwidth: 3.3X improvement on 4KB

Performance Evaluations for Collective Communication

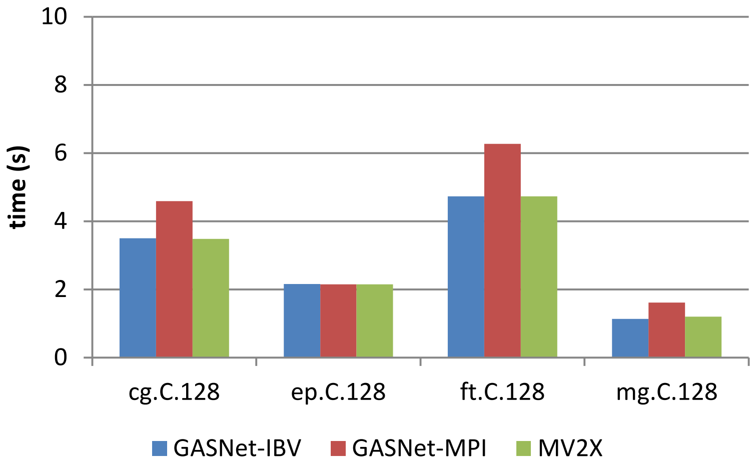




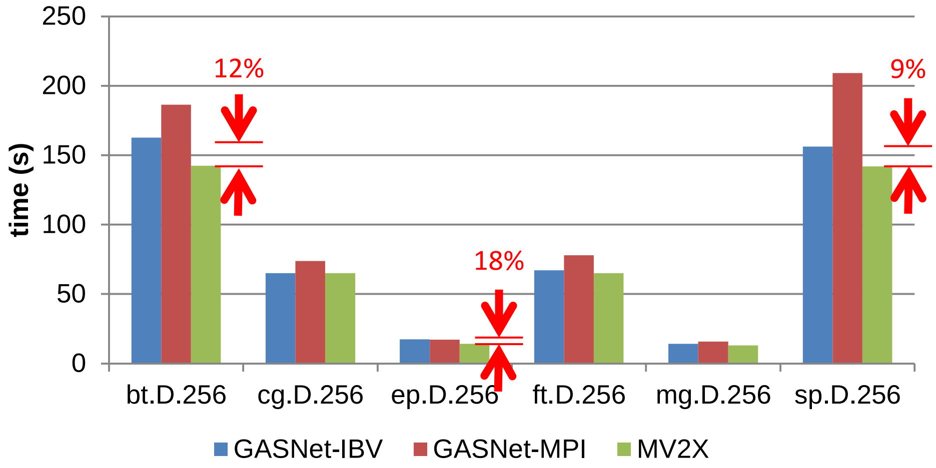
- Bandwidth improvement (MV2X-*cohyb* vs. GASNet-IBV)
  - CO_REDUCE: 2.3X improvement on 1KB, 1.1X improvement on 128KB
  - CO_BROADCAST: 4.0X improvement on 1KB, 1.3X improvement on 1MB

Performance Evaluations for Application (NPB)

C.128



D.256



- Application performance improvement: NPB3.3-CAF (MV2X vs. GASNet-IBV)
  - Reduce the execution time by:
    - 12% (BT.D.256), 18% (EP.D.256), 9% (SP.D.256)

Outline

- Introduction
- Motivation
- Design
- Evaluation
- Conclusion

Conclusion

- CAF can leverage the high-performance provided by MVAPICH2-X by integrating with UCR in depth
  - Improve the bandwidth and latency of one-sided coarray access (e.g. up to 3.5X for inter-node put operation)
  - Improve the bandwidth of collective operations in the upcoming Fortran 2015 (e.g. up to 4.0X for 64-core broadcast)
  - Save the execution time of CAF applications (e.g. by 18% for EP.D.256 in NPB)
- The proposed design is available since MVAPICH2-X v2.1rc2 release!

Future Work

- The next steps for advanced support of CAF on MVAPICH2-X:
  - Further optimize the collective communication subroutines
  - Enable the team-level collective communication semantics
  - Enhance the atomic operations with UCR

Thank you!

{linjia, hamidouc, luxi, limin, panda}@cse.ohio-state.edu

Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

MVAPICH Web Page

http://mvapich.cse.ohio-state.edu/