Using MVAPICH2-X for Hybrid MPI + PGAS (OpenSHMEM and UPC) Programming

MVAPICH2 User Group (MUG) Meeting

by

Jithin Jose

The Ohio State University

E-mail: jose@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~jose

Introduction

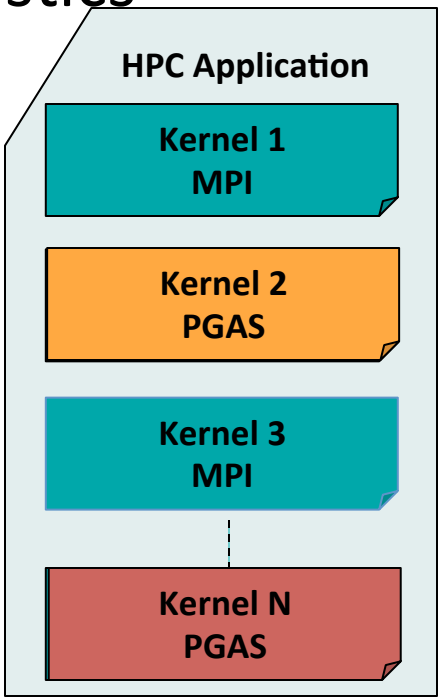
- MPI - the de-facto programming model for scientific parallel applications
- Offers attractive features for High Performance Computing (HPC) applications
  - Non blocking, One sided, etc.
- MPI Libraries (MVAPICH2) over InfiniBand optimized to the hilt
- Emerging Partitioned Global Address Space (PGAS) models -Unified Parallel C (UPC), OpenSHMEM

Partitioned Global Address Space (PGAS) Models

- PGAS Model
  - Shared memory abstraction over distributed systems
  - Better programmability
- OpenSHMEM (http://openshmem.org/)
  - Open specification to standardize the SHMEM (SHared MEMory) model; Library based
- Unified Parallel C (UPC)
  - Based on extensions to ISO C99; Compiler based
- *Will applications be re-written entirely in PGAS model?*
  - Probably not; PGAS models still emerging
  - Hybrid MPI+PGAS might be better

Hybrid (MPI+PGAS) Programming

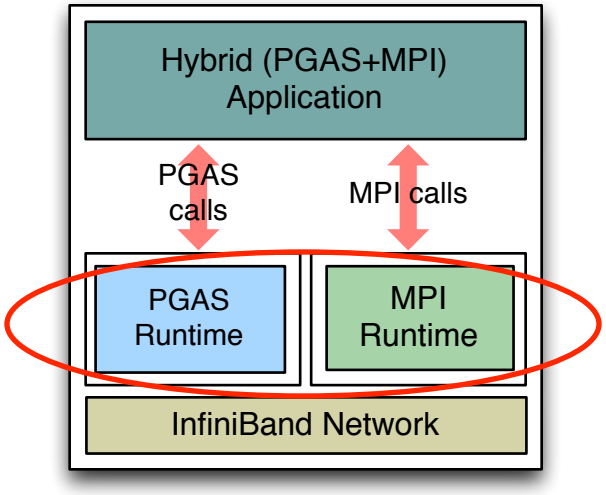
- Application sub-kernels can be re-written in MPI/ PGAS based on communication characteristics
- Benefits:
  - Best of Distributed Computing Model
  - Best of Shared Memory Computing Model
- Exascale Roadmap*:
  - "Hybrid Programming is a practical way to program exascale systems"



* The International Exascale Software Roadmap, Dongarra, J., Beckman, P. et al., Volume 25, Number 1, 2011, International Journal of High Performance Computer Applications, ISSN 1094-3420

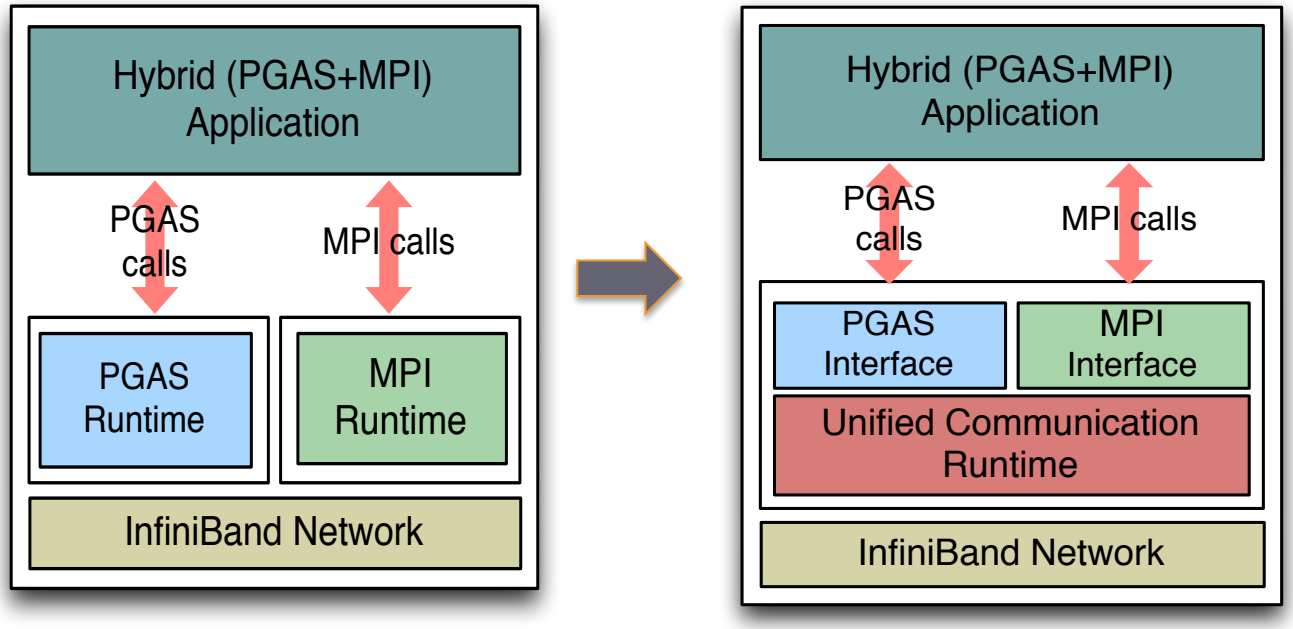
Current approaches for Hybrid Programming

- Layering one programming model over another
  - Poor performance due to semantics mismatch
- Separate runtime for each programming model



- Need more network resources
- Might lead to deadlock!
- Poor performance

Our Approach:
Unified Communication Runtime



- Optimal network resource usage
- No deadlock because of single runtime
- Better performance

MVAPICH2-X

- Unified communication runtime for MPI, UPC, OpenSHMEM available with MVAPICH2-X 1.9 onwards!
  - http://mvapich.cse.ohio-state.edu
- Feature Highlights
  - Supports MPI(+OpenMP), OpenSHMEM, UPC, MPI(+OpenMP) + OpenSHMEM, MPI(+OpenMP) + UPC
  - MPI-3 compliant, OpenSHMEM v1.0 standard compliant, UPC v1.2 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

MVAPICH2-X RPMs

- MVAPICH2-X RPMs available for:
  - Enterprise Linux 5
    - Compatible with OFED 1.5.4.1 and Mellanox OFED (based on OFED 1.5.3)
  - Enterprise Linux 5 (Stock InfiniBand Packages)
    - Compatible with OFED 3.5 and RHEL5 InfiniBand packages
  - Enterprise Linux 6
    - Compatible with OFED 1.5.4.1 and Mellanox OFED (based on OFED 1.5.3)
  - Enterprise Linux 6 (Stock InfiniBand Packages)
    - Compatible with OFED 3.5 and RHEL6 InfiniBand packages
  - Please contact us at mvapich-help@cse.ohio-state.edu for other platforms
- MVAPICH2-X RPMs are relocatable

Downloading and Installing MVAPICH2-X

- Downloading MVAPICH2-X RPMs
  - wget http://mvapich.cse.ohio-state.edu/download/mvapich2x/mvapich2-x -2.0a.rhel6.tar.gz
- Tarball contents:
  - GNU and Intel RPMs for MVAPICH2-x, OpenSHMEM, and UPC
- Install using rpm command
  - rpm –Uvh [--prefix=install-path] *.rpm --force --nodeps
  - Default installation location is /opt/mvapich2-x

Compiling programs with MVAPICH2-X

- Compile MPI programs using mpicc
  - $ mpicc -o helloworld_mpi helloworld_mpi.c
- Compile UPC programs using upcc
  - $ upcc -o helloworld_upc helloworld_upc.c
- Compile OpenSHMEM programs using oshcc
  - $ oshcc -o helloworld_oshm helloworld_oshm.c
- Compile Hybrid MPI+UPC programs using upcc
  - $ upcc -o hybrid_mpi_upc hybrid_mpi_upc.c
- Compile Hybrid MPI+OpenSHMEM programs using oshcc
  - $ oshcc -o hybrid_mpi_oshm hybrid_mpi_oshm.c

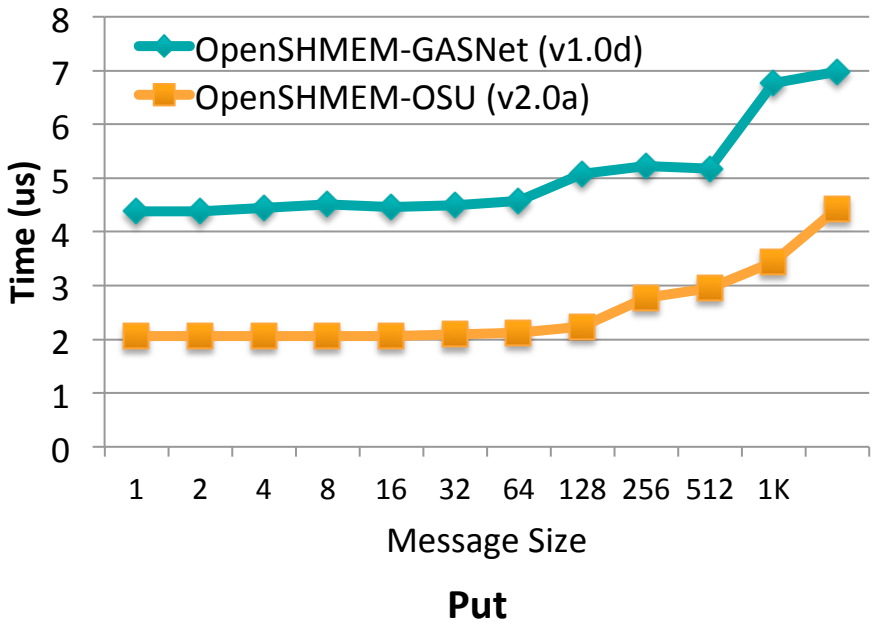
Running Programs with MVAPICH2-X

- MVAPICH2-X programs can be run using
  - mpirun_rsh and mpiexec.hydra (MPI, UPC, OpenSHMEM and hybrid)
  - upcrun (UPC)
  - oshrun (OpenSHMEM)
- Running using mpirun_rsh/mpiexec.hydra
  - $ mpirun rsh -np 4 -hostfile hosts ./test
  - $ mpiexec -f hosts -n 2 ./test
- Running using upcrun
  - $ export MPIRUN CMD="<path-to-MVAPICH2-X-install>/bin/mpirun rsh -np %N -hostfile hosts %P %A"
  - $ upcrun -n 2 ./test
- Running using oshrun
  - $ oshrun -f hosts -np 2 ./test

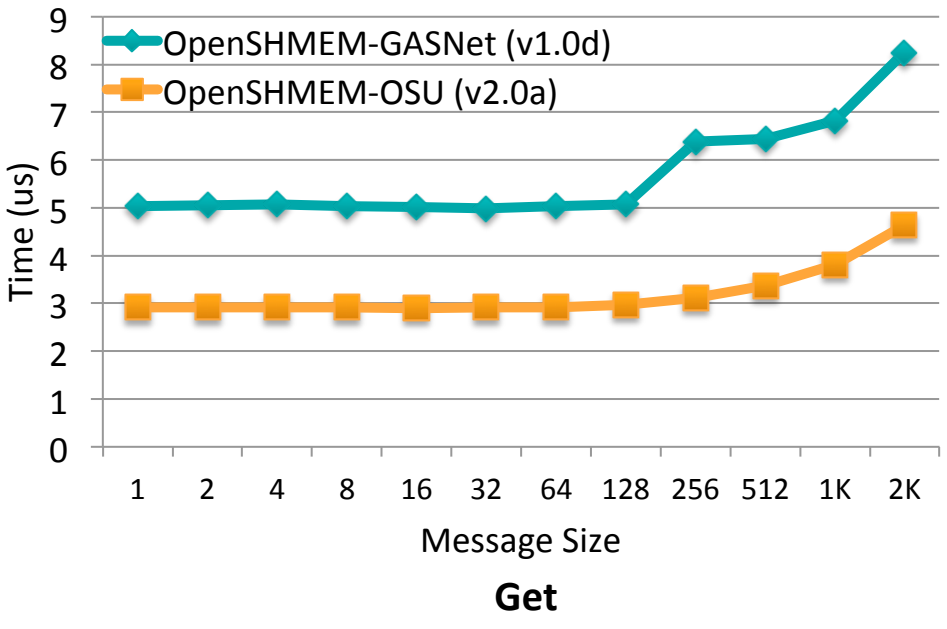
OSU Microbenchmarks – UPC and OpenSHMEM

- OpenSHMEM benchmarks
  - osu_oshm_put – Put latency
  - osu_oshm_get – Get latency
  - osu_oshm_put_mr – Put message rate
  - osu_oshm_atomics – Atomics latency
  - osu_oshm_collect – Collect latency
  - osu_oshm_broadcast – Broadcast latency
  - osu_oshm_reduce - Reduce latency
  - osu_oshm_barrier - Barrier latency
- UPC benchmarks
  - osu upc memput – Put latency
  - osu upc memget - Get latency

OpenSHMEM Put/Get Performance



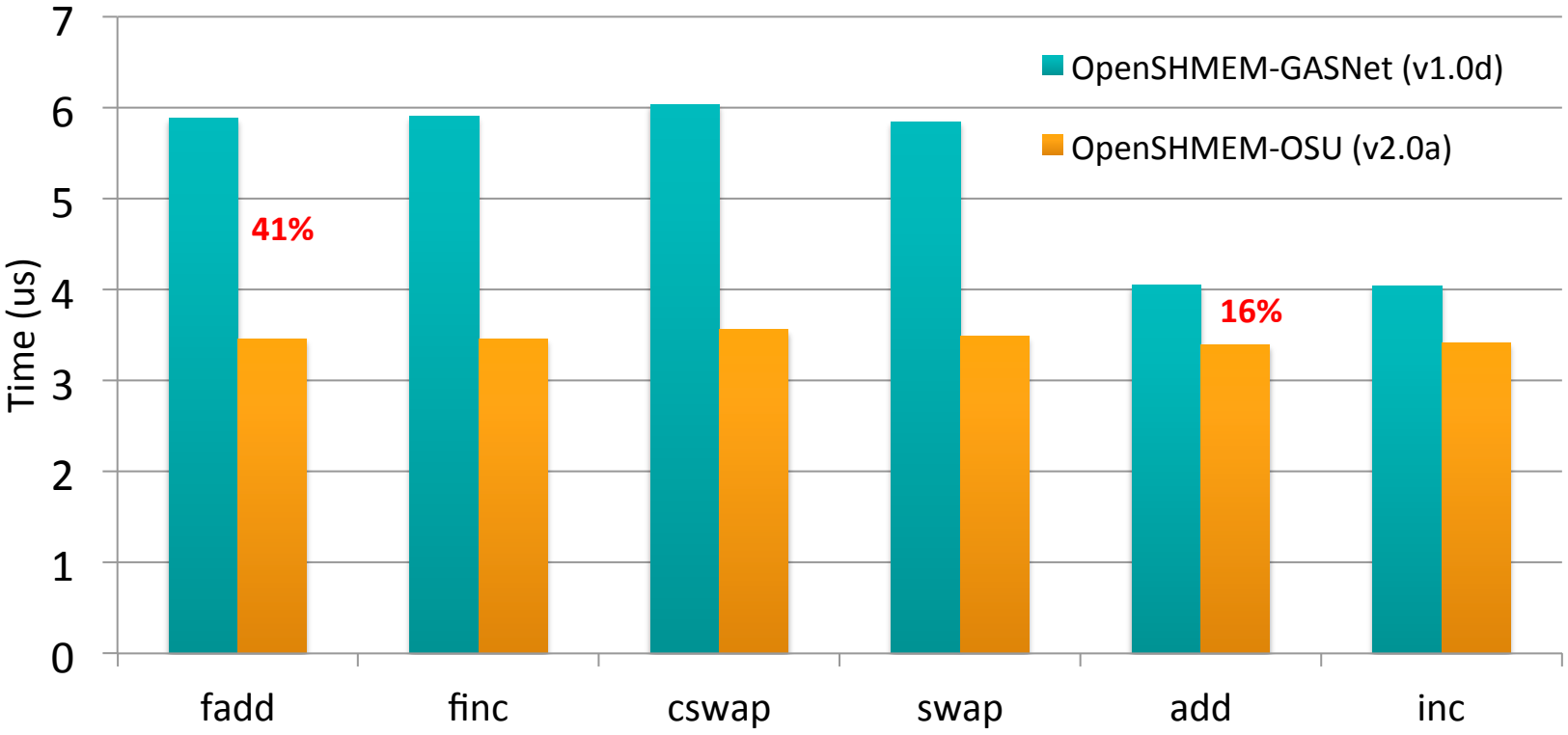
Put



Get

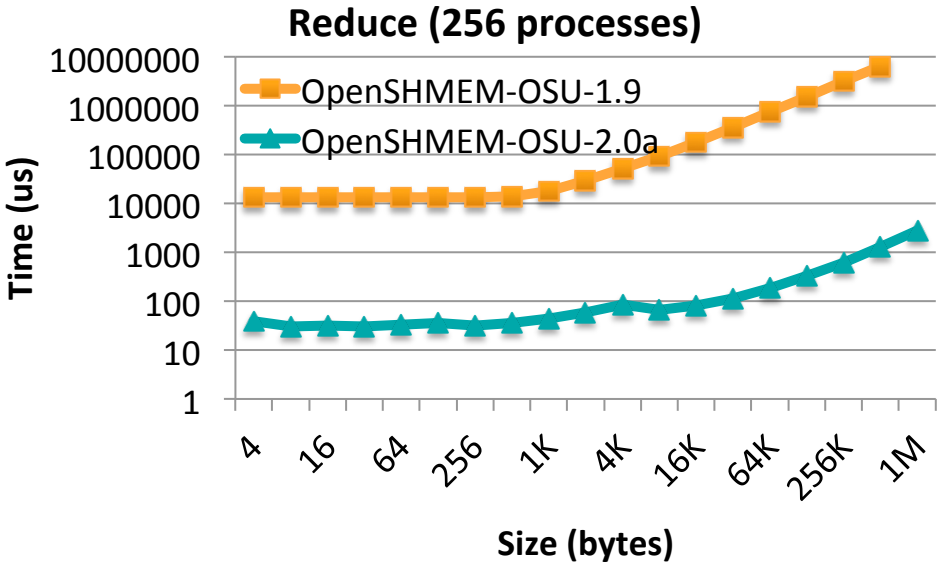
- OSU OpenSHMEM micro-benchmarks (OMB v4.1)
- Better performance for OpenSHMEM put and get with OSU design

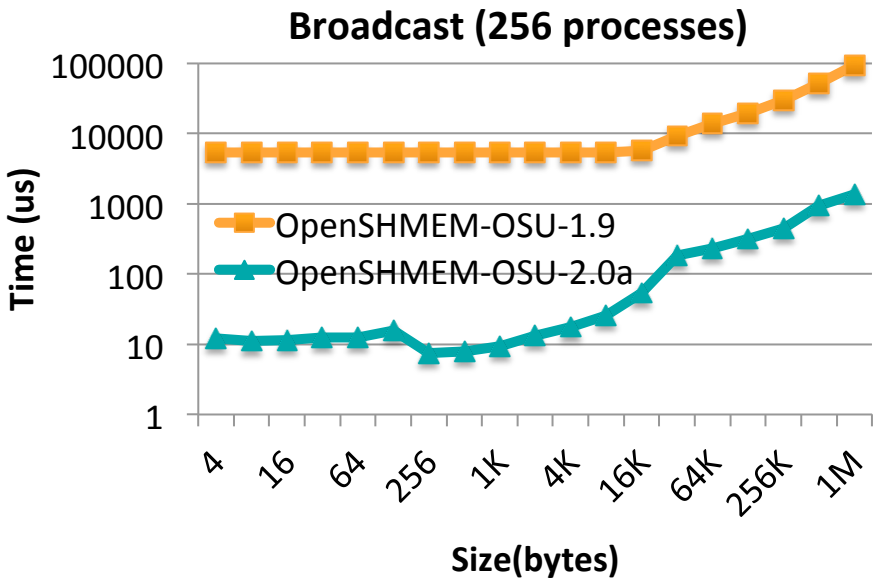
OpenSHMEM Atomics Performance

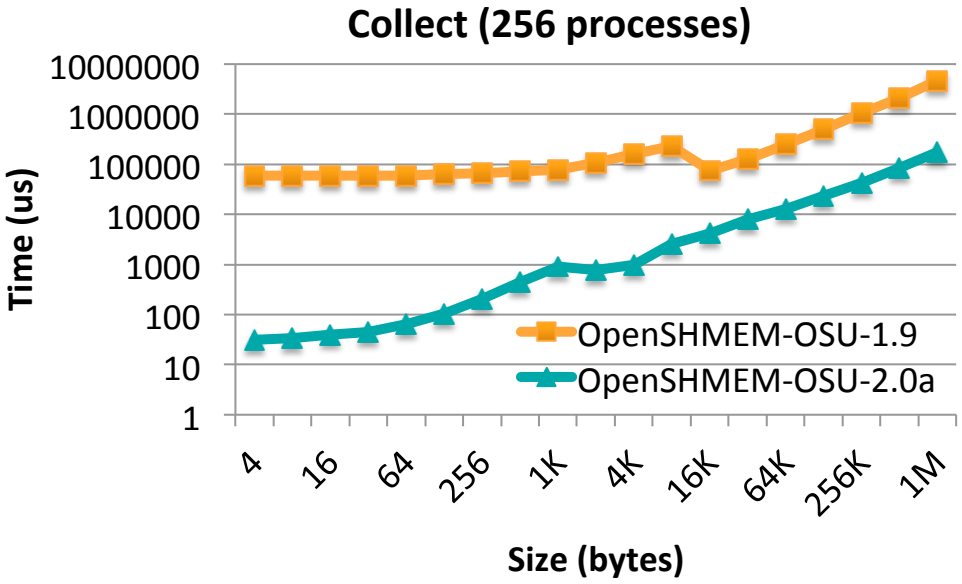


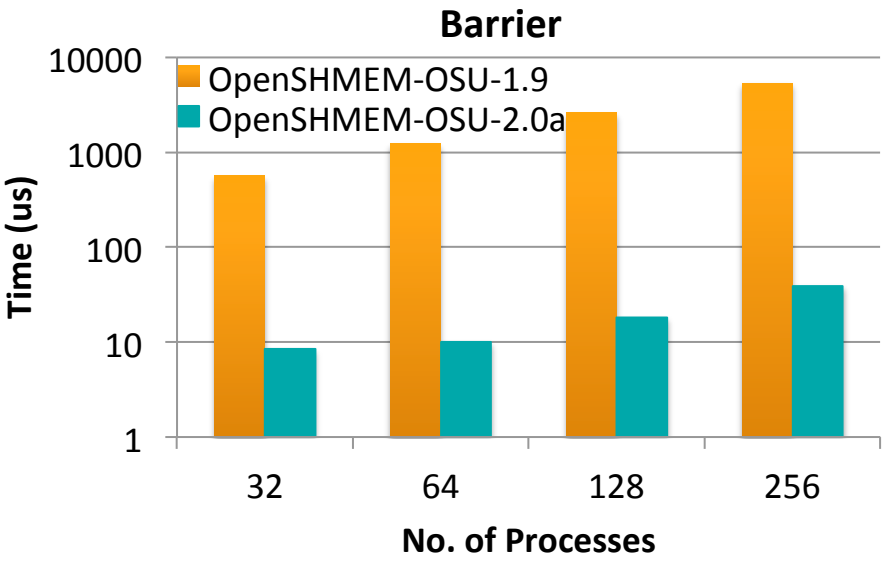
- OSU OpenSHMEM micro-benchmarks (OMB v4.1)
- Better performance for OpenSHMEM atomics with OSU design

Optimized OpenSHMEM Collectives in MVAPICH2-X 2.0a

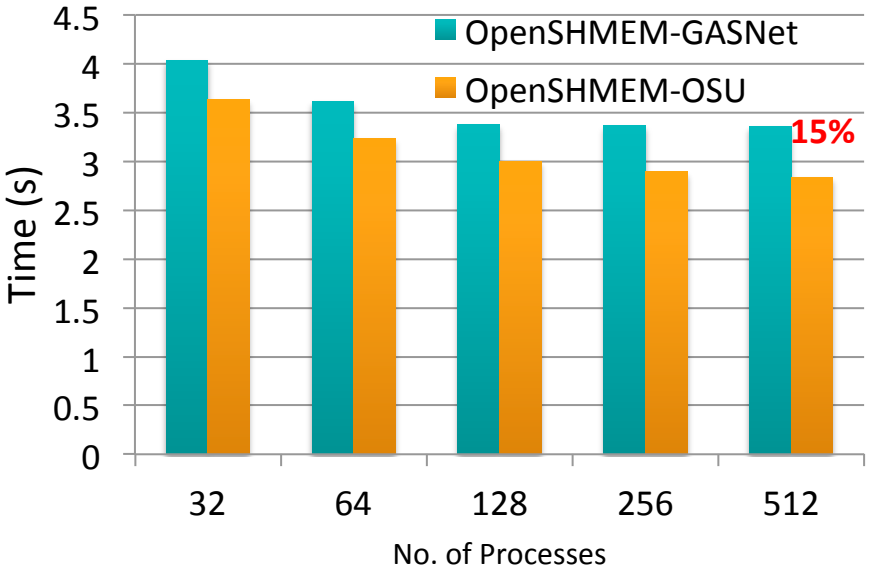




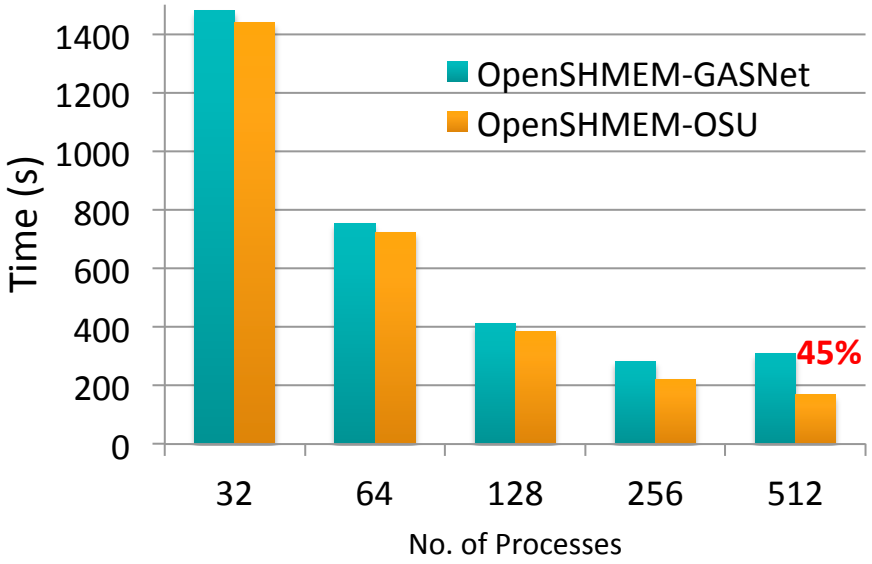




OpenSHMEM Application Performance



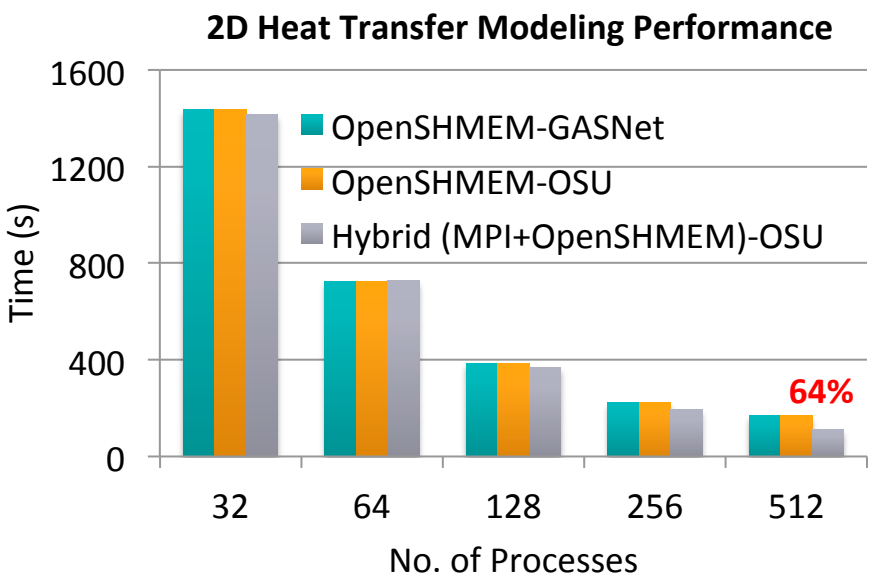
2D FFT

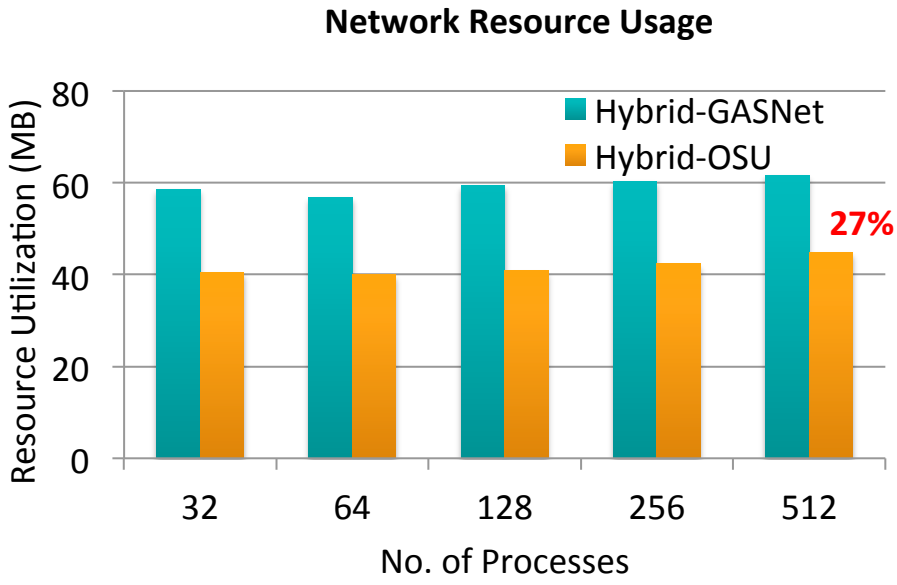


2D Heat Transfer Modeling

- 2D FFT with 8K input matrix
  - 15% improved performance for 512 processes
- 2D Heat Transfer Modeling (8K x 8K)
  - 45% improved performance for 512 processes

Performance of Hybrid (OpenSHMEM+MPI) Applications

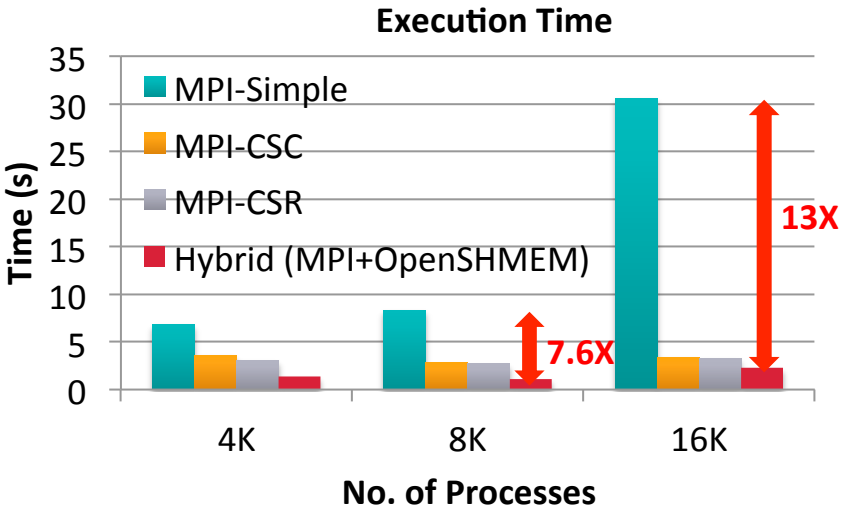


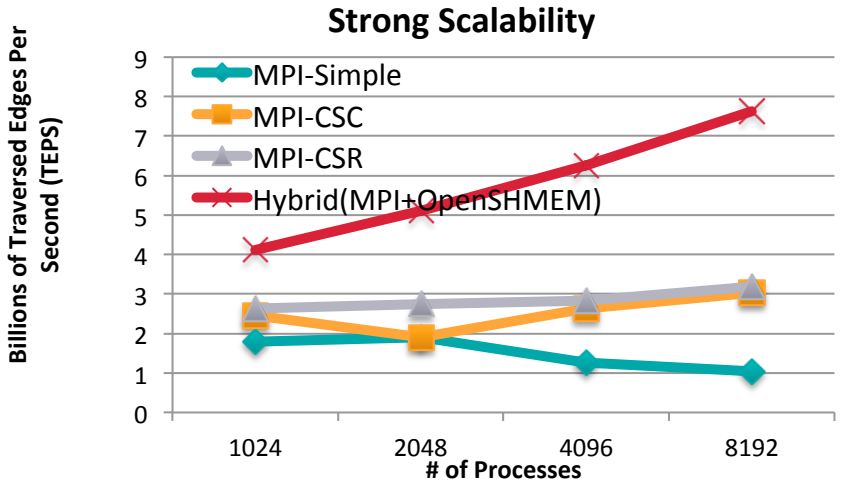


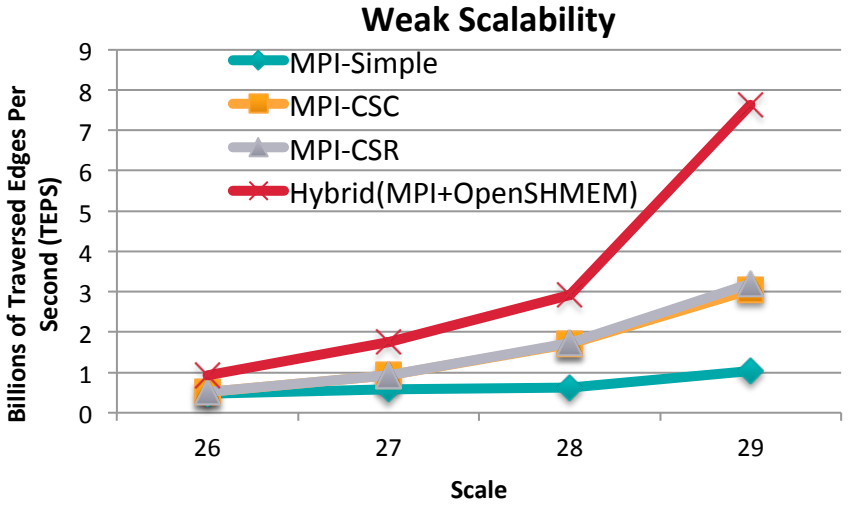
- Improved Performance for Hybrid Applications
  - 64% improvement for Hybrid (MPI+OpenSHMEM) 2DHeat Transfer Modeling with 512 processes over pure OpenSHMEM application
  - 34% improvement over Hybrid GASNet application
- Our approach with single Runtime consumes 27% lesser network resources

J. Jose, K. Kandalla, M. Luo and D. K. Panda, Supporting Hybrid MPI and OpenSHMEM over InfiniBand: Design and Performance Evaluation, Int'l Conference on Parallel Processing (ICPP '12), September 2012

Hybrid MPI+OpenSHMEM Graph500 Design



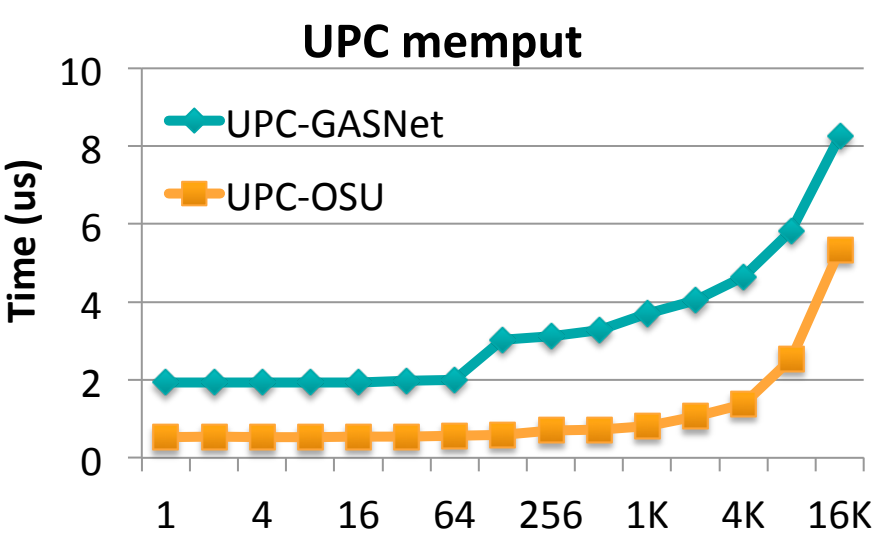


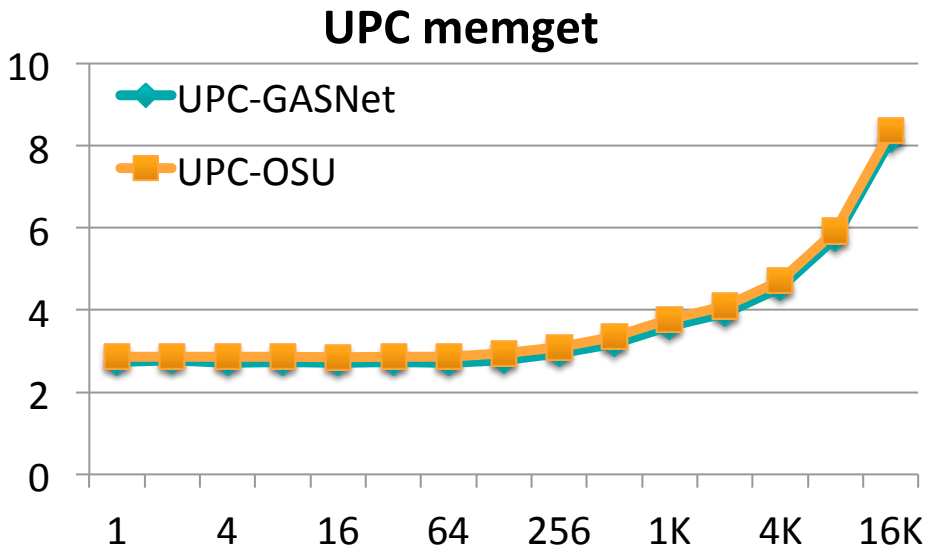


- Performance of Hybrid (MPI+OpenSHMEM) Graph500 Design
  - 8,192 processes
    - **2.4X** improvement over MPI-CSR
    - **7.6X** improvement over MPI-Simple
  - 16,384 processes
    - **1.5X** improvement over MPI-CSR
    - **13X** improvement over MPI-Simple

J. Jose, S. Potluri, K. Tomko and D. K. Panda, Designing Scalable Graph500 Benchmark with Hybrid MPI+OpenSHMEM Programming Models, International Supercomputing Conference (ISC'13), June 2013

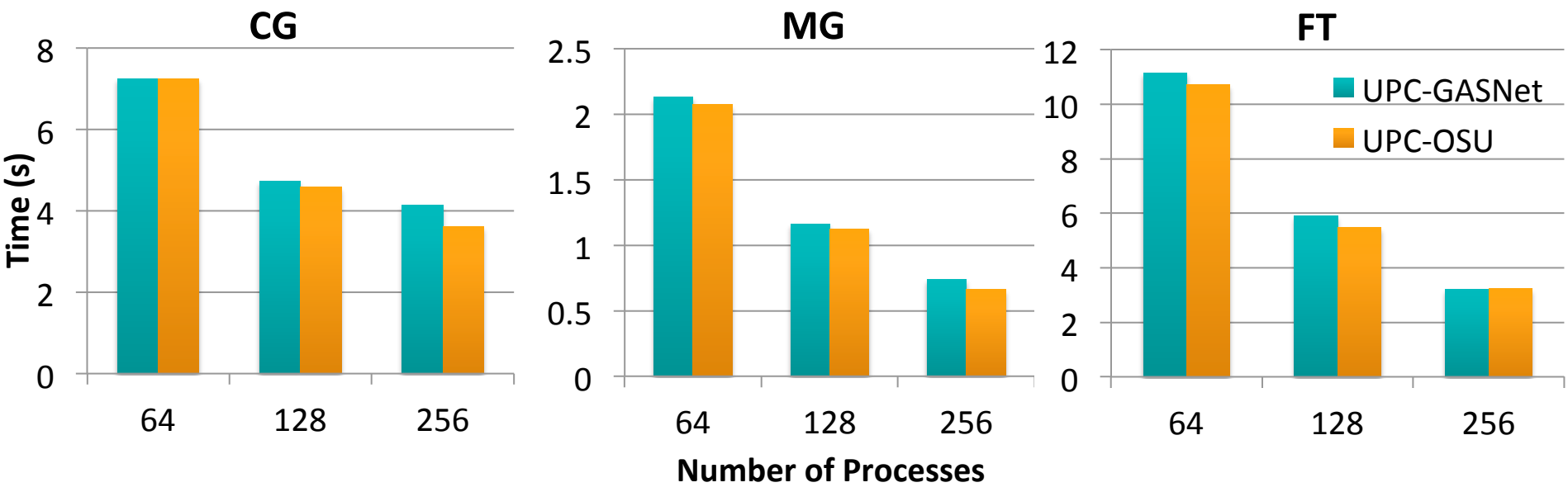
UPC Micro-benchmark Performance





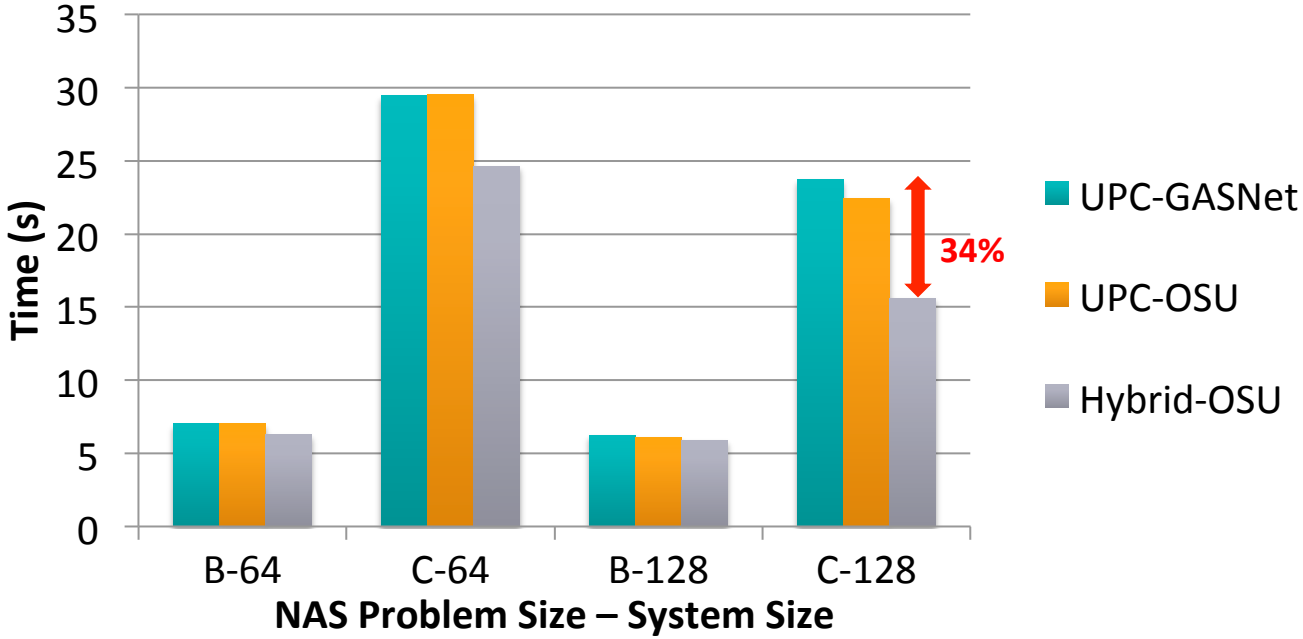
- OSU UPC micro-benchmarks (OMB v4.1)
- Better performance for UPC memput performance with UPC-OSU
- UPC memget performance is almost identical with both native UPC-GASNet and UPC-OSU conduits

Evaluation using UPC NAS Benchmarks



- Evaluations using UPC-NAS Benchmarks v2.4 (Class C)
- UPC-OSU performs equal or better than UPC-GASNet
- **11%** improvement for CG (256 processes)
- **10%** improvement for MG (256 processes)

Evaluation of Hybrid MPI+UPC NAS-FT



- Modified NAS FT UPC all-to-all pattern using MPI_Alltoall
- Truly hybrid program
- For FT (Class C, 128 processes)
  - **34%** improvement over UPC-GASNet
  - **30%** improvement over UPC-OSU

J. Jose, M. Luo, S. Sur and D. K. Panda, Unifying UPC and MPI Runtimes: Experience with MVAPICH, Fourth Conference on Partitioned Global Address Space Programming Model (PGAS '10), October 2010

MVAPICH2-X: FAQs

- Inadequate shared heap size
  - Set appropriate heap size

Parameter	Significance	Default
UPC_SHARED_HEAP_SIZE	Set UPC shared heap size	64M
OOSHM_USE_SHARED_HEAP_SIZE	Set OpenSHMEM symmetric heap size	512M

- Can't install mvapich2-x rpm in /opt
  - Use --prefix option when installing via rpm command

MVAPICH2-X: Looking forward

- Support for Accelerators and Co-processers
  - For UPC, OpenSHMEM and hybrid MPI+X (UPC/OpenSHMEM)
- Hybrid transport protocols for scalability
- Multi-end point runtime to improve network utilization
- Improving intra-node communication using CMA/LiMIC
- Optimizing collective communication in UPC/OpenSHMEM

Web Pointers

NOWLAB Web Page

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

MVAPICH Web Page

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