Experiences of MVAPICH2 at Huawei

Pak Lui
Futurewei Technologies

6th Annual MVAPICH User Group (MUG)

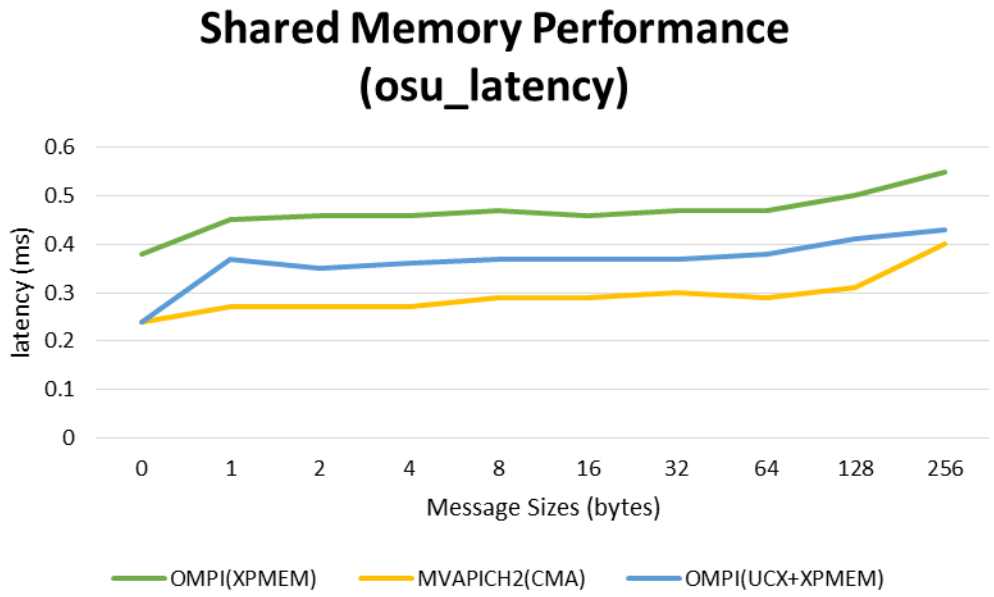
LEADING NEW ICT

Introduction

- To describe the experiences and share some of the findings
  - Using the latest MVAPICH2 and MVAPICH2-GDR software
  - With the work in progress with MVAPICH2 team
- OSU MPI Benchmarks are used for the following studies
  - OSU developed benchmarks as de-facto standard for benchmarking MPI communications
- To achieve better performance
  - Tuning MPI parameters in MPI
  - Comparing performance with another MPI
  - Evaluating with other supporting libraries

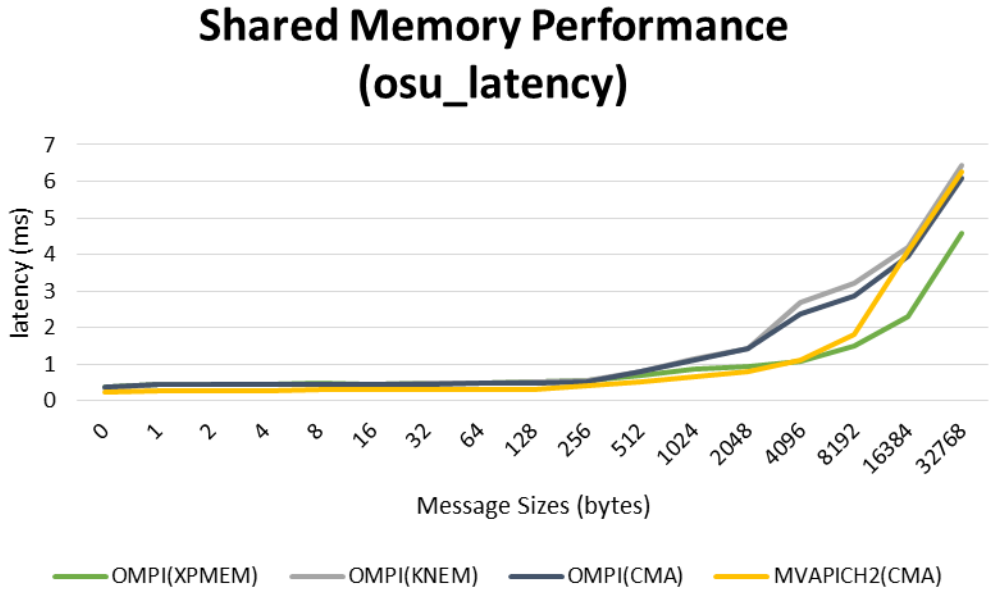
MVAPICH2: Intra-node Communications

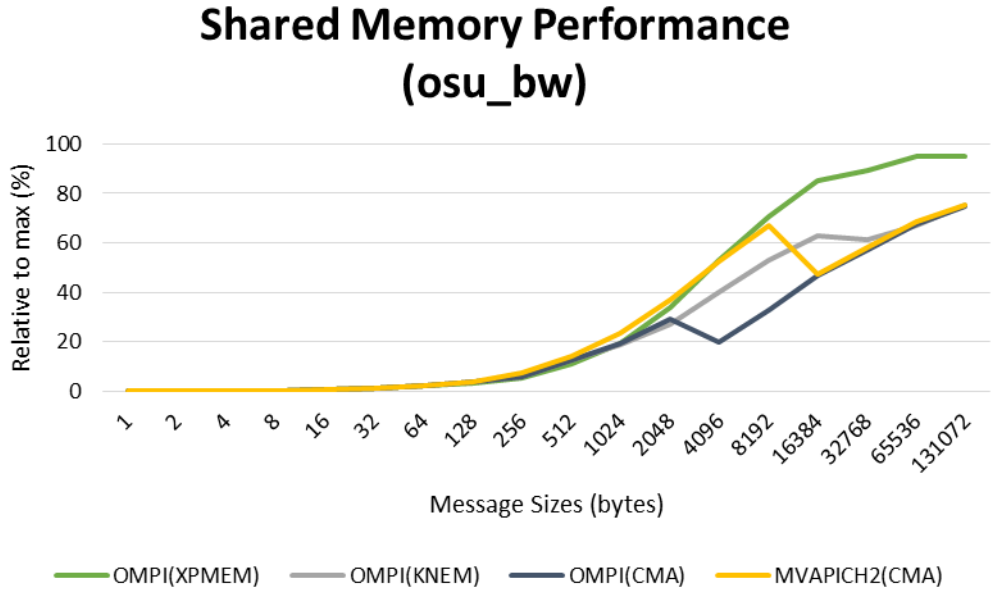
- A few of the kernel assisted intra-node communication models are available in MPI
  - XPMEM - Cross Partition Memory (https://gitlab.com/hjelmn/xpmem)
  - KNEM – (http://knem.gforge.inria.fr)
  - CMA – Cross Memory Attach, available in Linux kernel
  - Other approach to rely on the POSIX SHMEM
    - Rely on processes copy-in, copy-out from MMAP region
    - Shown to have less performance than the kernel assisted methods
  - MVAPICH2 demonstrates lower latency compared to Open MPI on small messages
    - XPMEM or CMA made difference in performance for large messages



MVAPICH2: Intra-node Communications

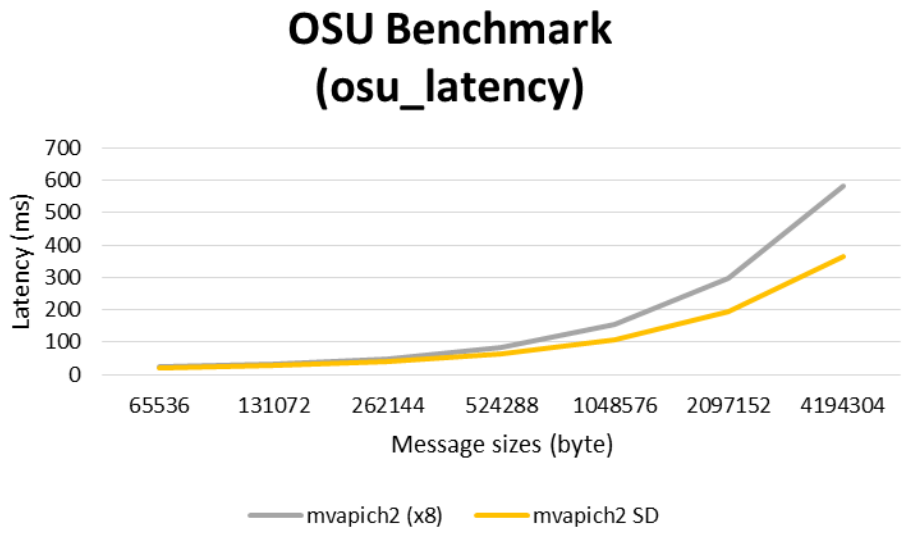
- Default for shared memory support in MVAPICH2 is CMA
  - Kernel support for shared memory communications in MVAPICH2 (CMA)
- XPMEM shown to deliver higher bandwidth for the kernel assisted intra-node modules tested
  - XPMEM performs best among the ones tested (XPMEM, KNEM and CMA) for Open MPI
  - Defaults limits for OMPI vader btl: <4KB eager, <32KB for pipeline and above for rendezvous
  - MVAPICH2 to consider XPMEM as an option for intra-node communications

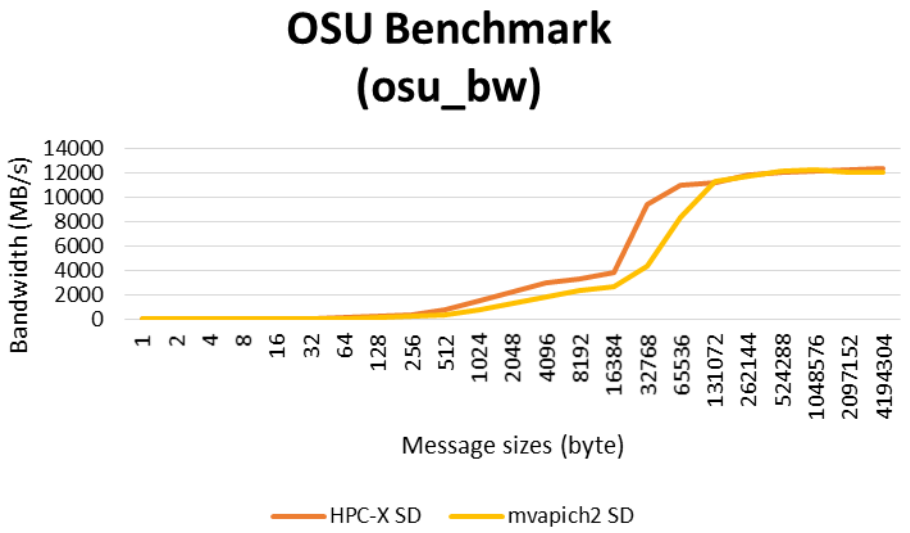




MVAPICH2: Inter-node Communications

- Large message latency improved by using round robin between HCAs
  - Using MV2 multi-rail support for ConnectX-5 Socket Direct
  - Improve mid/large message performance on multi-rail configuration
- Possibility to exploit process-to-rail mapping for this HCA device
  - MV2_PROCESS_TO_RAIL_MAPPING can be set to FIXED_MAPPING
  - Communications to CPU closes to the HCA to avoid traffic crossing CPU interconnect
- Opportunity for improvement in MVAPICH2 in tuning for the midrange messages





MVAPICH2: Collective Communications

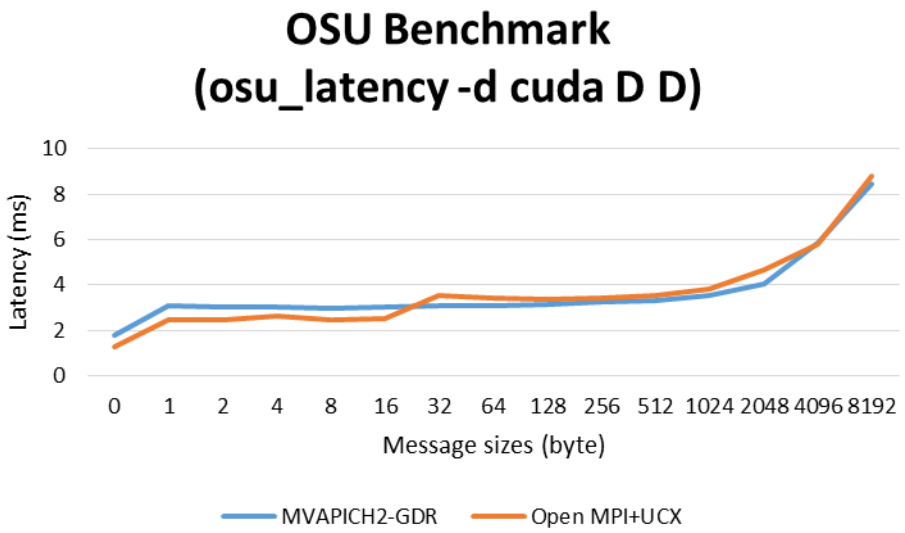
- Shared memory collectives tuning improves performance on dense core CPUs
  - MV2_USE_SHMEM_COLL=1
    - Improve on shared memory based collective communications
  - MV2_USE_MCAST=0
    - MCAST supports MPI_Bcast, MPI_Scatter, and MPI_Allreduce
    - Appears to show higher latency when run at small scale, thus disabled
- Tune for specific hardware architecture
  - MVAPICH2 shows warning for undefined hardware architecture
  - Would require help from MVAPICH2 team to tune specifically for particular hardware
  - To generate header files that show the best algorithm used for particular collective ops
  - Changes are made at compile time; No runtime tuning parameters
- To use code examples as a guideline provided by MVAPICH2-2.3
  - Detection of architecture:
    - Example: mvapich2-2.3/src/mpid/ch3/channels/common/include/mv2_arch_hca_detect.h
  - Detection of network:
    - Example: mvapich2-2.3/src/mpid/ch3/channels/common/src/detect/arch/mv2_arch_detect.c
- To try different collective tuning algorithms based on hardware architecture
  - For example: mvapich2-2.3/src/mpi/coll/tuning/scatter_arch_tuning.h

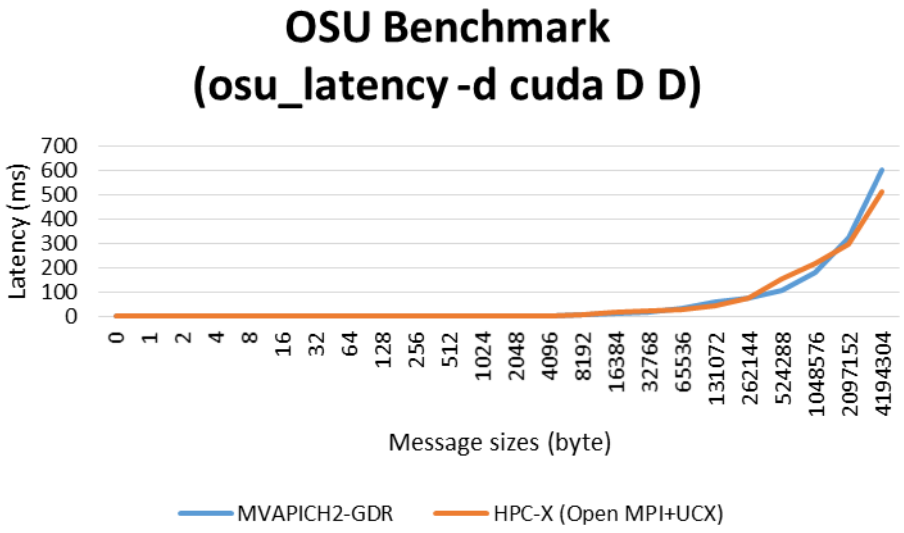
MVAPICH2-GDR – Inter-node GPU Communications

- Commands:
  - MVAPICH2-GDR 2.3a:
  - ```
    mpirun_rsh -export -np 2 -host /home/pak/hostfile/hostfile_2h
    MV2_USE_CUDA=1 MV2_USE_GPUDIRECT=1 MV2_USE_GPUDIRECT_GDRCOPY=1
    MV2_GPUDIRECT_GDRCOPY_LIB=/usr/lib64/libgdrapi.so MV2_IBA_HCA=mlx5_0:1
    MV2_NUM_HCA=1 MV2_SHOW_CPU_BINDING=2 MV2_CPU_MAPPING=1
    LD_PRELOAD=/opt/mvapich2/gdr/2.3a/mcast/no-
    openacc/cuda9.2/mofed4.3/mpirun/gnu4.8.5/lib64/libmpi.so /home/pak/osu-
    micro-benchmarks-5.3.2-mvapich2/mpi/pt2pt/osu_XXXXXX -d cuda D D
    ```
  - HPC-X 2.2 (OMPI 3.1.2a1 +UCX 1.4.0):
  - ```
    mpirun -x LD_LIBRARY_PATH -mca coll_hcoll_enable 0 -mca pml ucx -x
    UCX_NET_DEVICES=mlx5_0:1 -np 2 -host node01,node02 -x
    UCX_TLS=rc_x,cuda_copy,gdr_copy /home/pak/hpcx-v2.2.0-gcc-MLNX_OFED_LINUX-
    4.3-1.0.1.0-redhat7.4-x86_64/ompi/tests/osu-micro-benchmarks-5.3.2-
    cuda/osu_XXXXXX -d cuda D D
    ```
- Configuration: E5-2697v3, ConnectX-4 IB, K80, CUDA 9.2, MLNX_OFED 4.3, CentOS 7.4, EDR IB switch

MVAPICH2-GDR – Inter-node GPU Communications

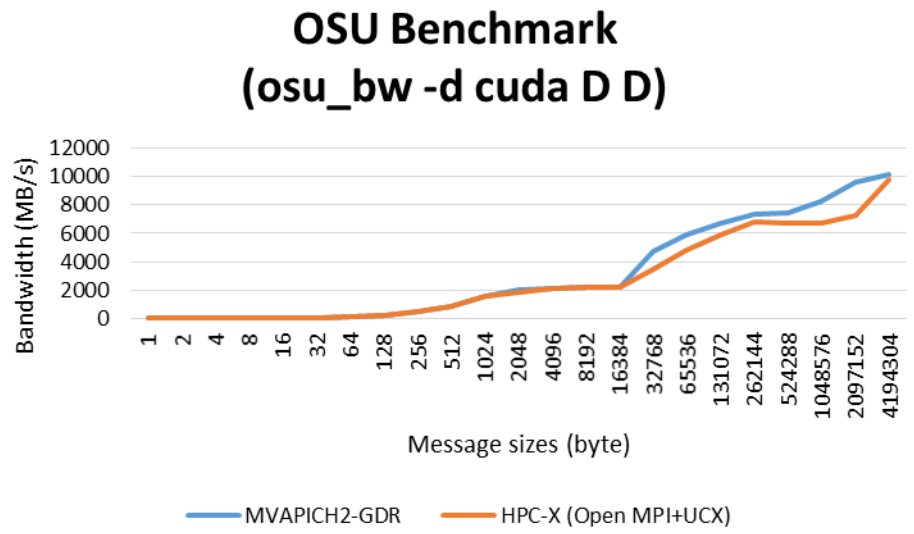
- MVAPICH2-GDR takes advantage of GDRCOPY for performance
  - GPUDirect RDMA support the nv_peer_memory kernel module
  - GDRCOPY is a fast copy library as a kernel module (https://github.com/NVIDIA/gdrcopy)
- Recent changes in UCX bring performance of Open MPI closer to MVAPICH2-GDR
  - The performance of MVAPICH2-GDR and Open MPI with UCX perform at similar level
  - The latency for both MPIs should improve when using the latest GPU to run

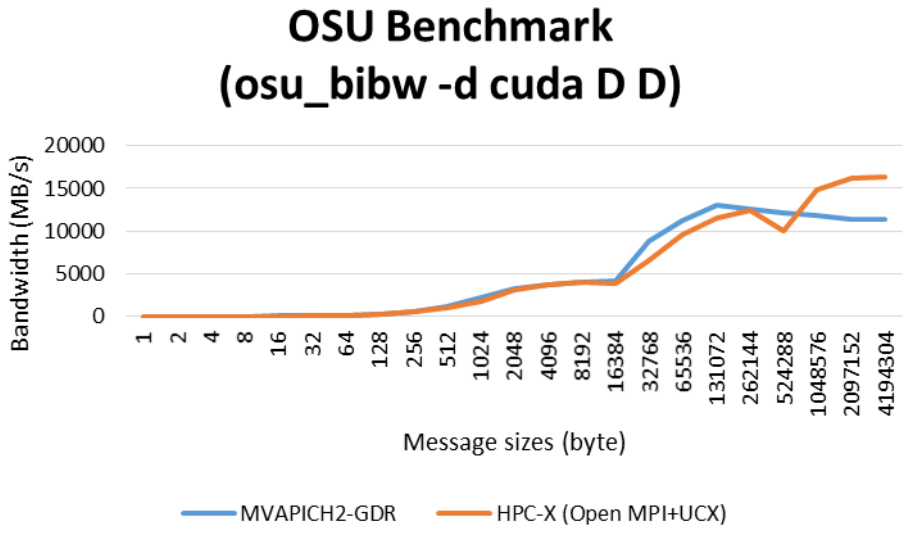




MVAPICH2-GDR – Inter-node GPU Communications

- MVAPICH2 leads osu_bw performance for large messages over 16KB
  - The osu_bw shows higher throughput achieved than Open MPI with UCX support
- Bi-directional Bandwidth for both differentiate at large messages
  - Higher bi-directional bandwidth achieved for Open MPI with UCX
  - Both perform similarly on small messages below 16KB





THANK YOU