System Software Laboratory

KU KONKUK UNIVERSITY

Kernel-Level Support for MPI Intra-Node Communication (Post-LiMIC2): Project Overview

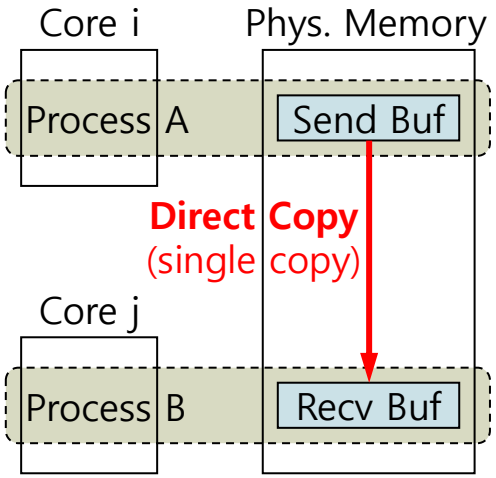
Hyun-Wook Jin
System Software Laboratory
Dept. of Computer Science and Engineering
Konkuk University
jinh@konkuk.ac.kr

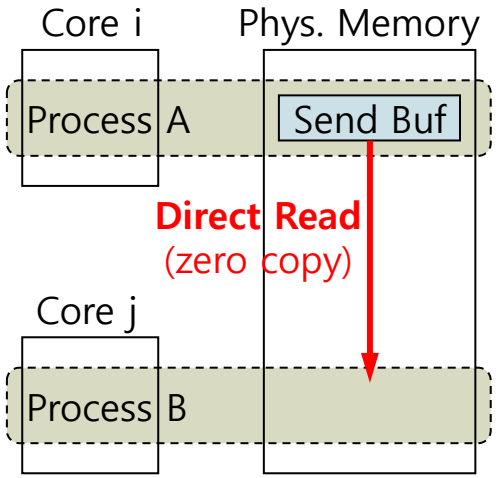
Contents

- Background
- Post-LiMIC2 project
  - Power efficiency
  - Skew tolerance
  - Better manageability
- Concluding remark

Kernel-Level Support for MPI Intra-Node Communication

- Memory mapping
  - Directly move a message from source to destination buffer by means of kernel-level support
  - Single data copy
    - Beneficial for ptp with large messages
  - Zero data copy
    - Beneficial for read-only





Instances of Kernel-Level Support

- LiMIC/LiMIC2
  - Opened the era of one-copy intra-node communication
    - H-W. Jin, S. Sur, L. Chai, and D. K. Panda, "LiMIC: Support for High-Performance MPI Intra-Node Communication on Linux Cluster," in Proc. of ICPP-05, Jun. 2005.
    - H.-W. Jin, S. Sur, Lei Chai, and D. K. Panda, "Lightweight Kernel-Level Primitives for High-Performance MPI Intra-Node Communication over Multi-Core Systems," in Proc. of IEEE Cluster 2007, Sep. 2007.
  - LiMIC2-0.5 was publicly released with MVAPICH2-1.4RC1 (Jun. 2009)
  - LiMIC2-0.5.6 is being released with the latest MVAPICH2
    - mvapich2-src]$ ./configure --with-limic2 [omit other configure options]
    - mvapich2-src]$ mpirun_rsh -np 4 -hostfile ~/hosts MV2_SMP_USE_LIMIC2=1 [path to application]

Instances of Kernel-Level Support

- KNEM
  - Asynchronous communication
    - D. Buntinas, B. Goglin, D. Goodell, G. Mercier, and S. Moreaud, "Cache-Efficient, Intranode Large-Message MPI Communication with MPICH2-Nemesis," in Proc. of ICPP, Sep. 2009.
  - Enhanced collectives
    - T. Ma, G. Bosilca, A. Bouteiller, B. Goglin, J. M. Squyres and J. J. Dongarra, "Kernel Assisted Collective Intra-node MPI Communication among Multi-Core and Many-Core CPUs," in Proc. of ICPP, 2011.
  - Security model
    - B. Goglin and M. Stephanie, "KNEM: a generic and scalable kernel-assisted intra-node MPI communication framework," JPDC, Feb. 2013.
  - MPICH2 and OpenMPI contained KNEM support

Instances of Kernel-Level Support

- CMA
  - In-kernel implementation + New system calls
    - J. Vienne, "Benefits of Cross Memory Attach for MPI Libraries on HPC Clusters," in Proc. of XSEDE 14, Jul. 2014.
  - Default intra-node communication channel for large messages in MVAPICH2

Instances of Kernel-Level Support

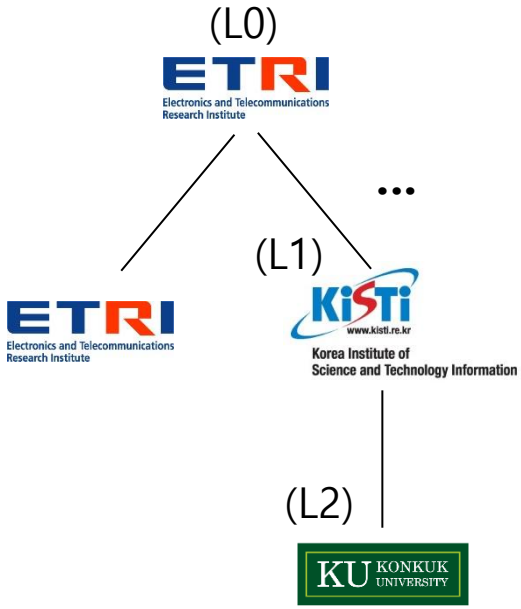
- XPMEM
  - Supports memory mapping to user-level address space
    - B. Kocoloski and J. Lange, "XEMEM: Efficient Shared Memory for Composed Applications on Multi-OS/R Exascale Systems," in Proc. of HPDC 2015, 2015.
  - MVAPICH2 with XPMEM
    - Collectives
      - J. M. Hashmi, S. Chakraborty, M. Bayatpour, H. Subramoni, and D. K. Panda, "Designing Efficient Shared Address Space Reduction Collectives for Multi-/Many-cores," in Proc. of IEEE IPDPS, 2018.
    - Data types
      - J. M. Hashmi, S. Chakraborty, M. Bayatpour, H. Subramoni and D. K. Panda, "FALCON: Efficient Designs for Zero-Copy MPI Datatype Processing on Emerging Architectures," in Proc. of IEEE IPDPS, 2019.

15 Years Old

- Adolescent
  - Strom and stress period (G. Stanley Hall)
    - Conflict with parents
    - Mood disruption
    - Risky behavior
  - Face the challenge of understanding the self
- Challenges of MPI intra-node communication in exascale systems
  - Power efficiency
  - Skew tolerance
  - Better manageability
  - …

Post-LiMIC2 Project

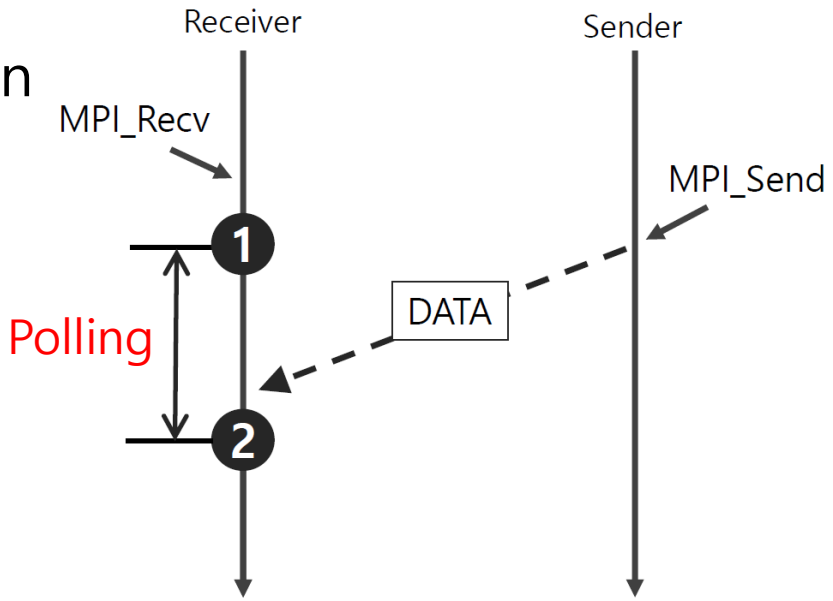
- (L0) Development on computing nodes of supercomputer based on super-parallel processor
  - 2020. 07. 06. ~ 2024. 04. 05.
  - Electronics and Telecommunications Research Institute (ETRI)
- (L1) Development of Supercomputing SW stack on co-designed processor
  - Korea Institute of Science and Technology Information (KISTI)
- (L2: Post-LiMIC2) Research on distributed parallel programming models for super-parallel processors
  - Konkuk University



Power Efficiency

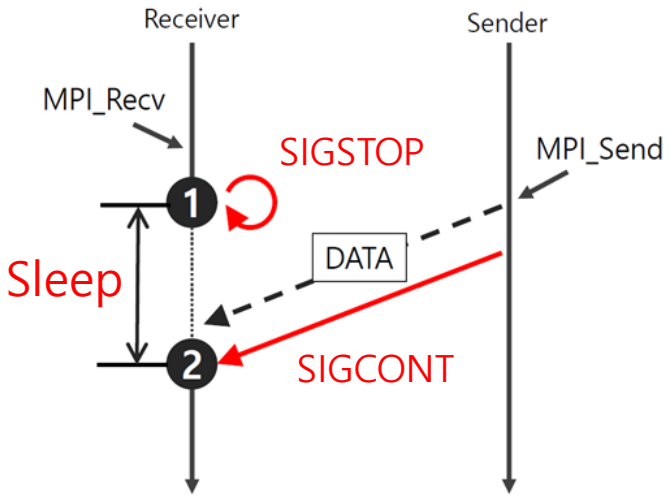
Power Efficiency

- Polling-based blocking
  - Blocking communication
    - MPI_Send, MPI_Recv
  - Nonblocking communication
    - MPI_Wait
  - Good
    - Performance (latency)
  - Bad
    - CPU resources
    - Energy

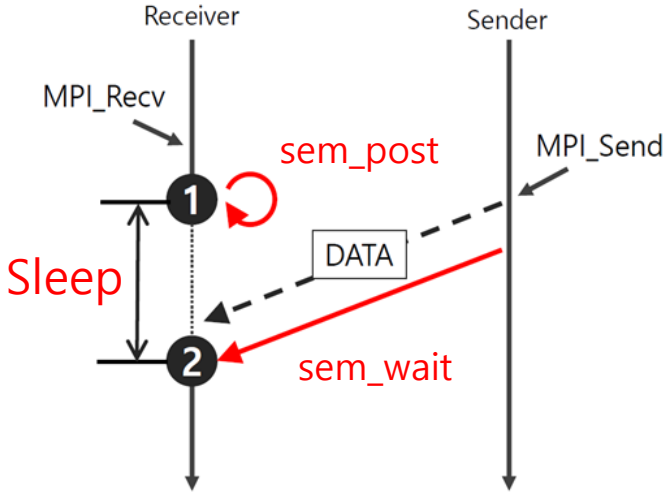


Event-based Blocking

- Signal events
  - Block and resume a process by using signals
    - SIGSTOP
    - SIGCONT
  - Signal can be lost



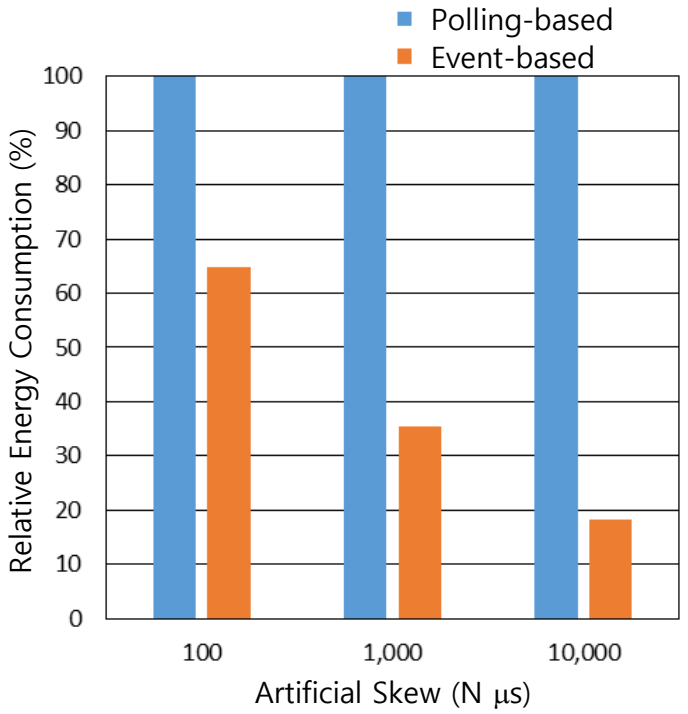
- Semaphore events
  - Block and resume a process by using a semaphore
    - sem_post
    - sem_wait



Preliminary Measurement Results

- Energy consumption
  - The larger the skew (N), the more energy is saved

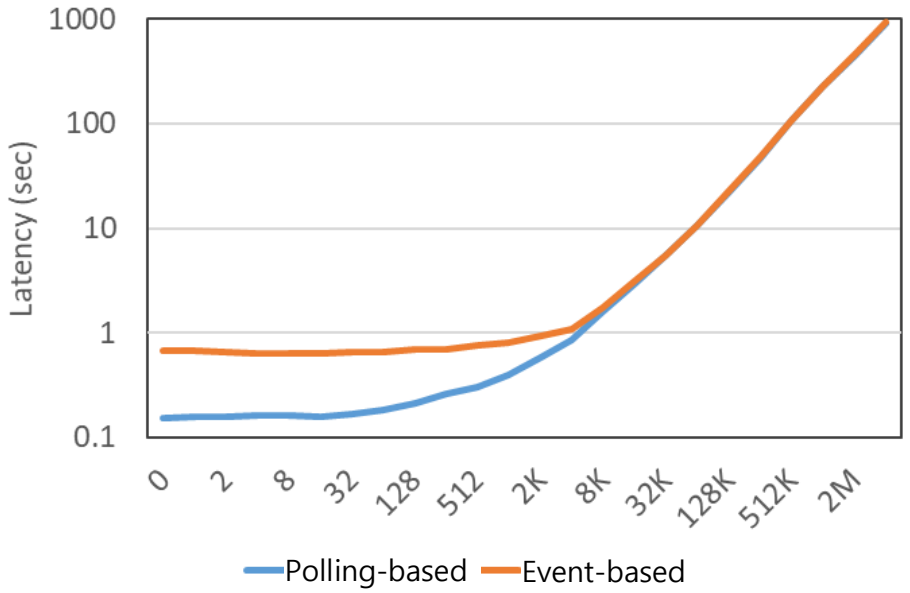
```
1:  procedure OSU_LATENCY
2:      for Number of iterations do
3:          if rank is 0 then
4:              delay for N micro seconds
5:              MPI_Send(to rank 1)
6:              MPI_Recv(from rank 1)
7:          if rank is 1 then
8:              MPI_Recv(from rank 0)
9:              delay for N micro seconds
10:             MPI_Send(to rank 0)
```



- Message size: 8KB (eager mode)
- Number of iterations: 10,000

Preliminary Measurement Results

- Latency
  - Event-based blocking harms latency when there is no skew



- Used the eager mode for all message sizes

Ongoing/Future Work

- Implementation of event-based blocking

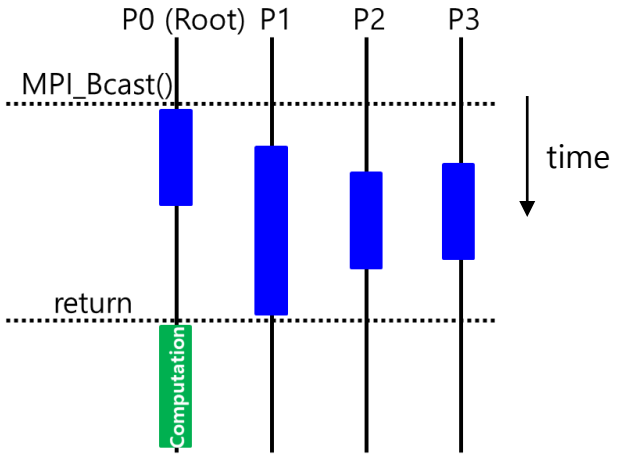
	Blocking APIs (MPI_Send, MPI_Recv)	Nonblocking APIs (MPI_Wait, MPI_Waitall)
Eager	✓	
Rendezvous		

- Harmonization with MVAPICH2-EA

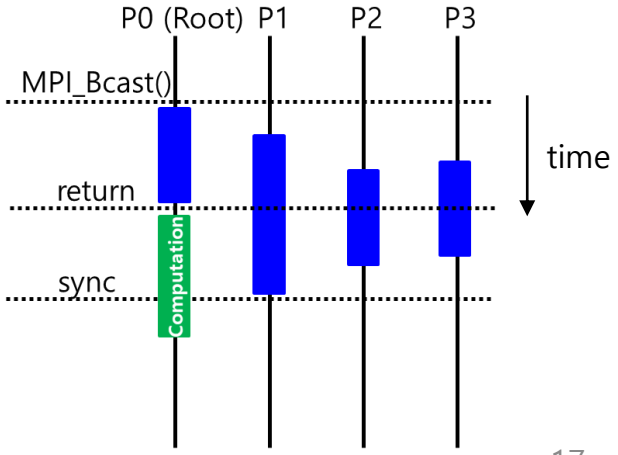
Skew Tolerance

Skew Tolerance

- Skew between MPI processes
  - Large-scale
    - Network delay
  - Uneven workload
    - Sparse matrices
  - Results in synchronization overhead and waste of resources



- Progress without waiting for other processes
  - Asynchronous communication
  - …



Asynchronous Communication

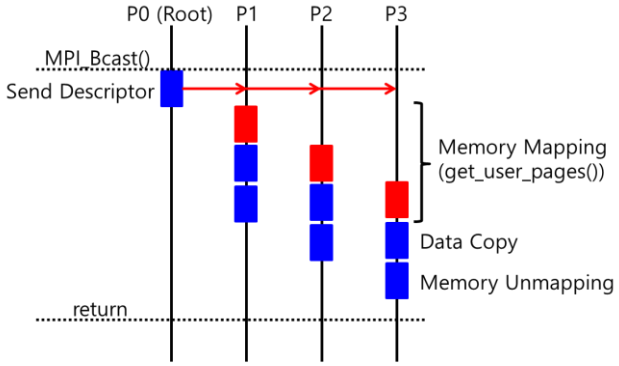
- Memory copy
  - K. Vaidyanathan , L. Chai , W. Huang , and DK Panda, “Efficient Asynchronous Memory Copy Operations on Multi-Core Systems and I/OAT,” in Proc. of IEEE Cluster, Sep 2007.
- Point-to-point
  - D. Buntinas, B. Goglin, D. Goodell, G. Mercier, and S. Moreaud, “Cache-Efficient, Intranode Large-Message MPI Communication with MPICH2-Nemesis,” in Proc. of ICPP, Sep. 2009.
- Collectives

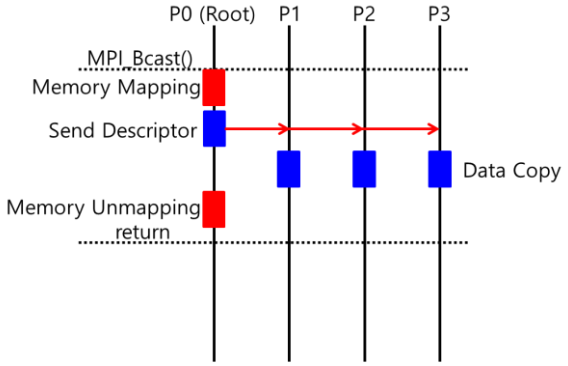
Asynchronous Collectives

- Asynchronous return
  - Page protection to preserve synchronous semantics
    - Page fault when the process tries to write
- Data copy offloading
  - Copy engine
    - DMA engine that moves data between buffers
    - e.g., Intel I/O Acceleration Technology (I/OAT)
  - Per-core request queues
  - Callback function

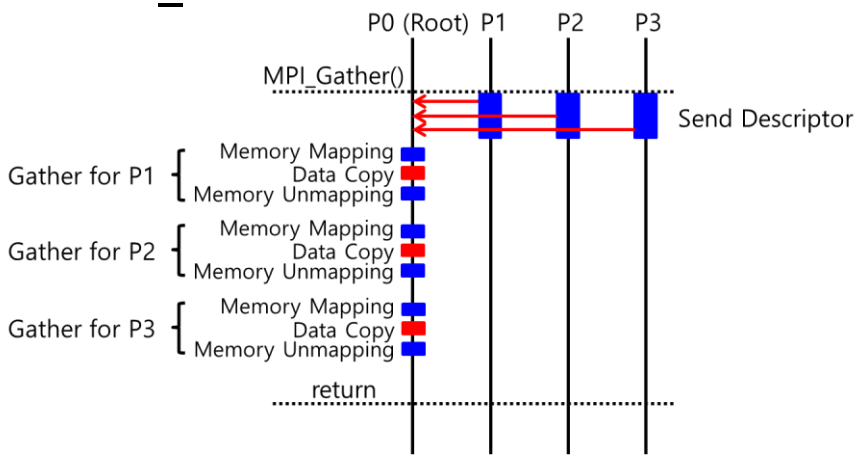
Asynchronous Collectives

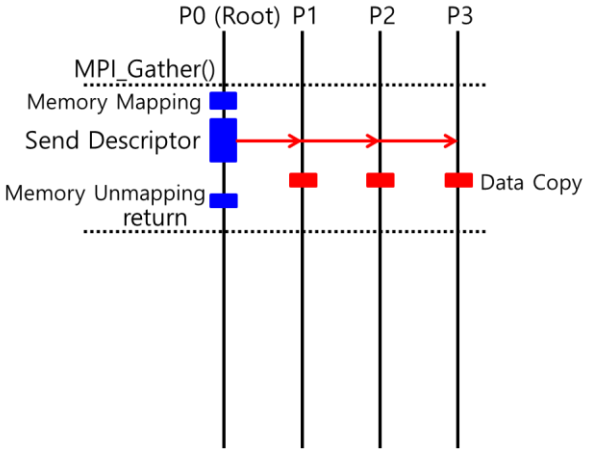
- Rollback to synchronous communication (MUG '19)
  - MPI_Bcast



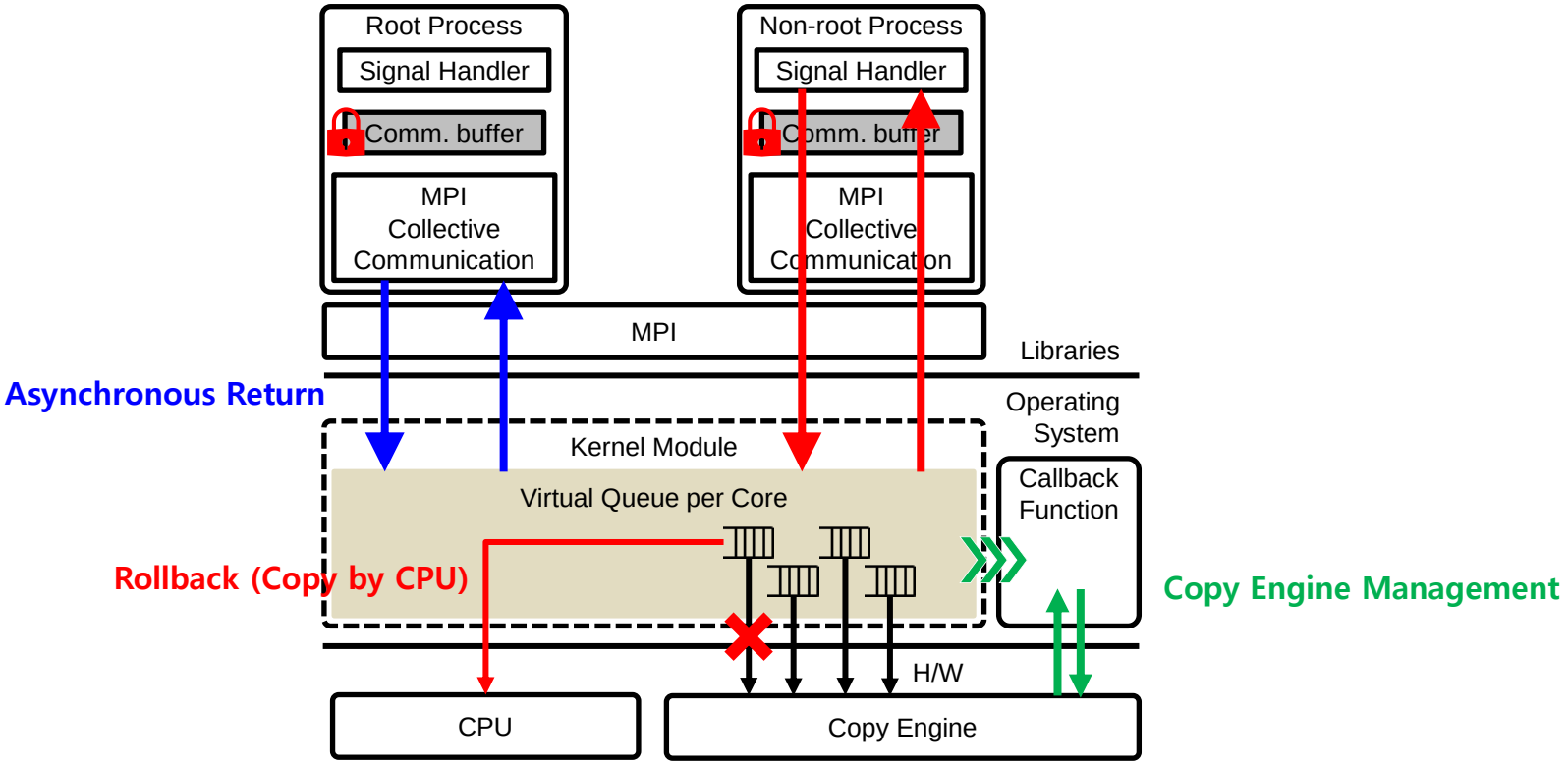


  - MPI_Gather



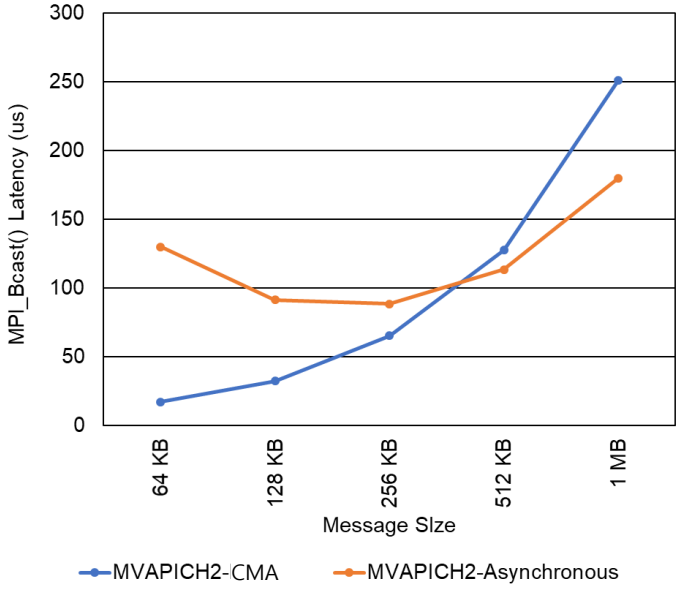


Asynchronous Collectives

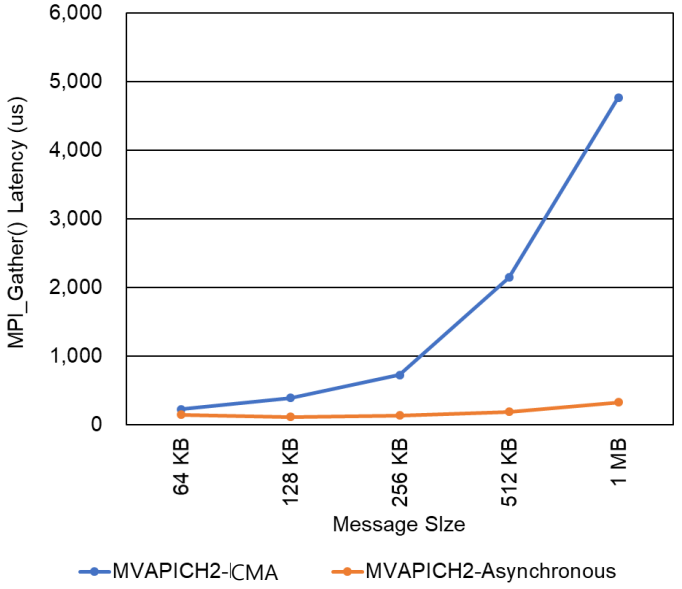


Preliminary Measurement Results

- MPI_Bcast
  - 16 processes/Xeon Haswell
  - 28% improvement with 1MB message
    - Beneficial for only large messages



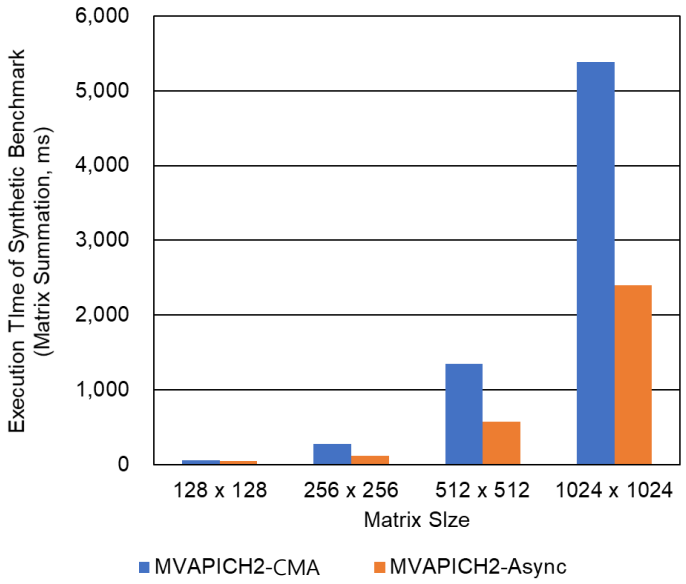
- MPI_Gather
  - 16 processes/Xeon Haswell
  - 93% improvement with 1MB message



Preliminary Measurement Results

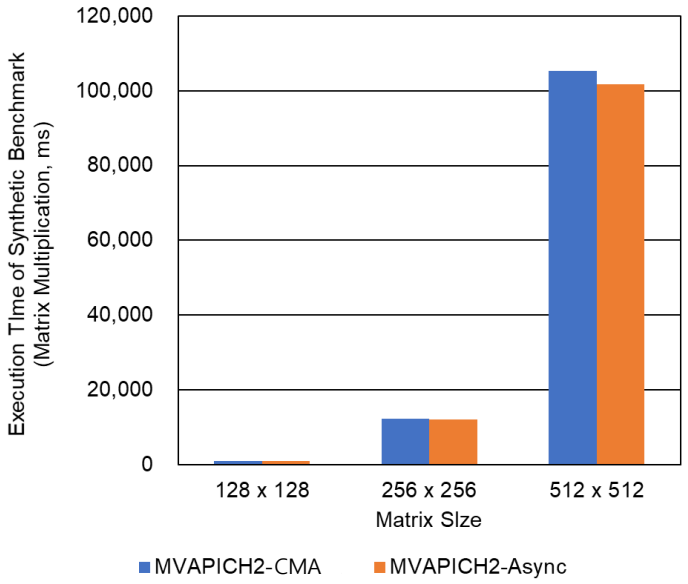
- Matrix summation

```
MPI_Bcast(buf_n)
Loop:
    MPI_Bcast(buf_n+1)
    sum_matrix(buf_n)
    MPI_Gather(buf_n)
    post_process(buf_n)
    n = n+1
```



- Matrix multiplication

```
MPI_Bcast(buf_n)
Loop:
    MPI_Bcast(buf_n+1)
    mul_matrix(buf_n)
    MPI_Gather(buf_n)
    post_process(buf_n)
    n = n+1
```



Ongoing/Future Work

- Implementation of asynchronous collectives
  - Policies of I/OAT callback function
  - Support for other collectives
- Finding realistic workload/benchmark
- Support for nonblocking collectives in MPI-3

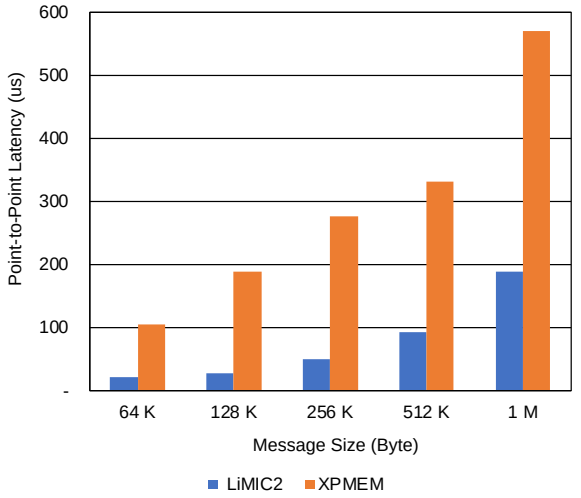
Better Manageability

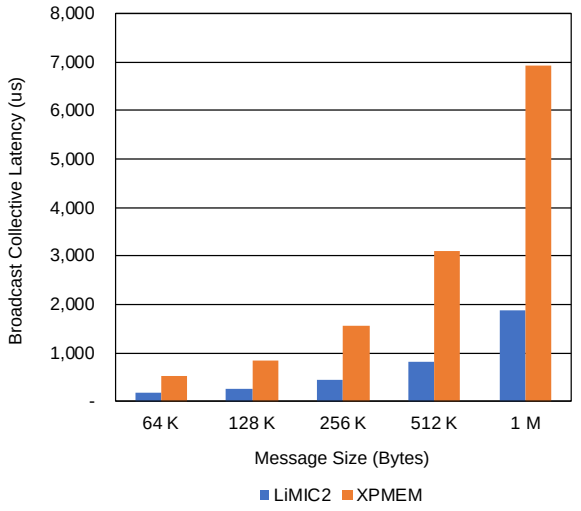
Better Manageability

- Several implementations of kernel-level support for MPI intra-node communication
  - LiMIC2
  - KNEM
  - CMA
  - XPMEM
- Different implementations have their own advantages
  - Hybrid approaches can be a good choice
  - Not a good idea to manage multiple kernel-modules and kernel-patches

Kernel-Level Mapping vs. User-Level Mapping

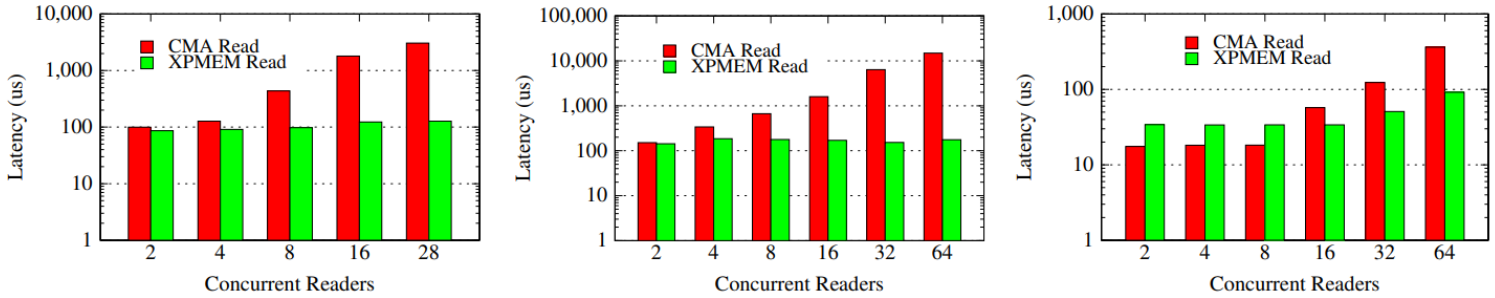
- Map-and-copy
  - Point-to-point pattern
    - XPMEM shows higher latency than LiMIC2 up to 7.4x
  - One-to-all pattern
    - XPMEM shows higher latency than LiMIC2 up to 3.8x
  - Memory mapping and page fault overheads of XPMEM are high





Kernel-Level Mapping vs. User-Level Mapping

- Map-and-read
  - One-to-all pattern



(a) Broadwell (b) Knights Landing (c) OpenPOWER

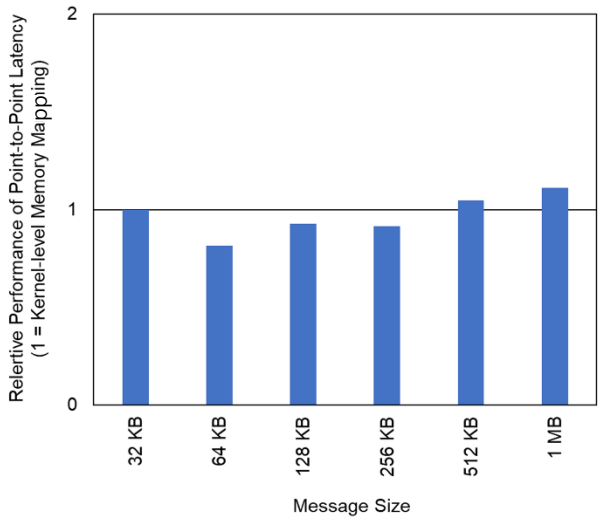
  - Reduction collectives
    - J. M. Hashmi, S. Chakraborty, M. Bayatpour, H. Subramoni, and D. K. Panda, "Designing Efficient Shared Address Space Reduction Collectives for Multi-/Many-cores," in Proc. of IEEE IPDPS 2018, 2018.
  - XPMEM allows a process to operate directly on the remote buffer without additional copies

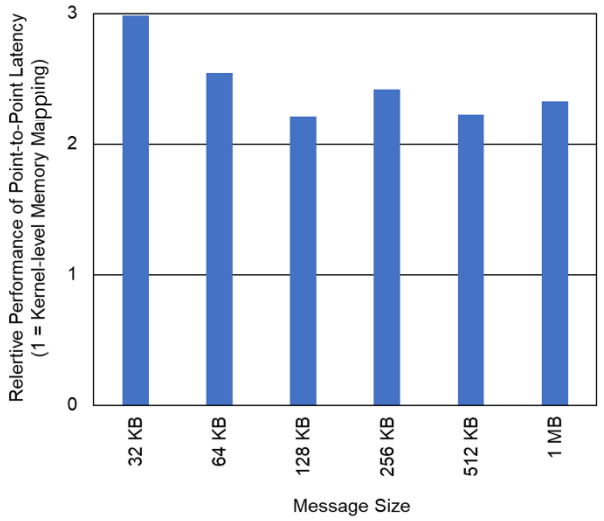
LiMIC2 + XPMEM

- Support for both Kernel- and user-level memory mapping
  - New APIs for user-level memory mapping
    - limicX_rx_comp
    - limicX_rx_comp_nocache
  - Memory mapping cache in LiMIC2
    - Basic ideas was borrowed from MVAPICH2
      - Thank Jahanzeb for his suggestion (MUG '19)
    - Red-black tree
    - Eviction policy: LRU

Preliminary Measurement Results

- Map-and-copy
  - Same buffer
    - Cache hit ratio = 100%
    - Comparable performance with kernel-level mapping even in map-and-copy
  - New buffers
    - Cache hit ratio = 0%





Ongoing/Future Work

- Implementation of LiMIC2+XPMEM
  - Stabilization and optimization
  - Patch for MVAPICH2
- License Issues (?)
  - libxpmem: LGPL
  - xpmem.h, xpmem_internal.h: GPL

Concluding Remark

- Post-LiMIC2 project
  - Power efficiency
    - Event-based blocking
  - Skew tolerance
    - Asynchronous collectives
  - Better manageability
    - LiMIC2 + XPMEM
- Collaboration with MVAPICH team
  - New features in kernel-level support
  - Ways of using kernel-level support in MPI

Thank You!


Ministry of Science and ICT


National Research Foundation of Korea


Korea Institute of Science and Technology Information