# Performance Analysis of MPI RMA in Supporting OpenSHMEM Runtime

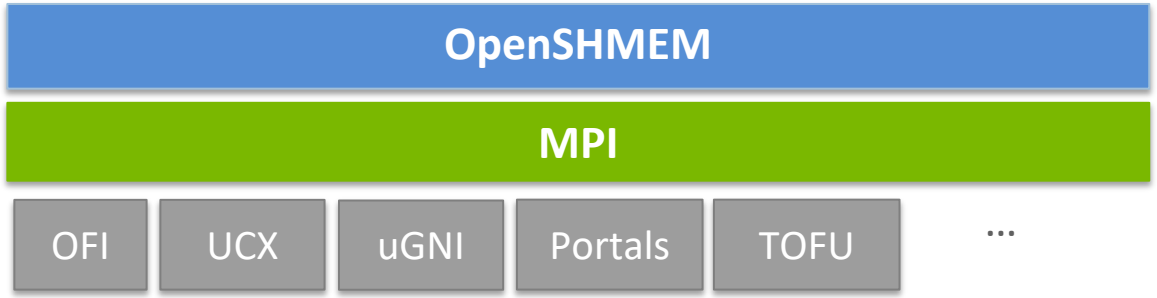
**Min Si**, Huansong Fu, Pavan Balaji

Programming Models and Runtime Systems Group

Argonne National Laboratory, USA

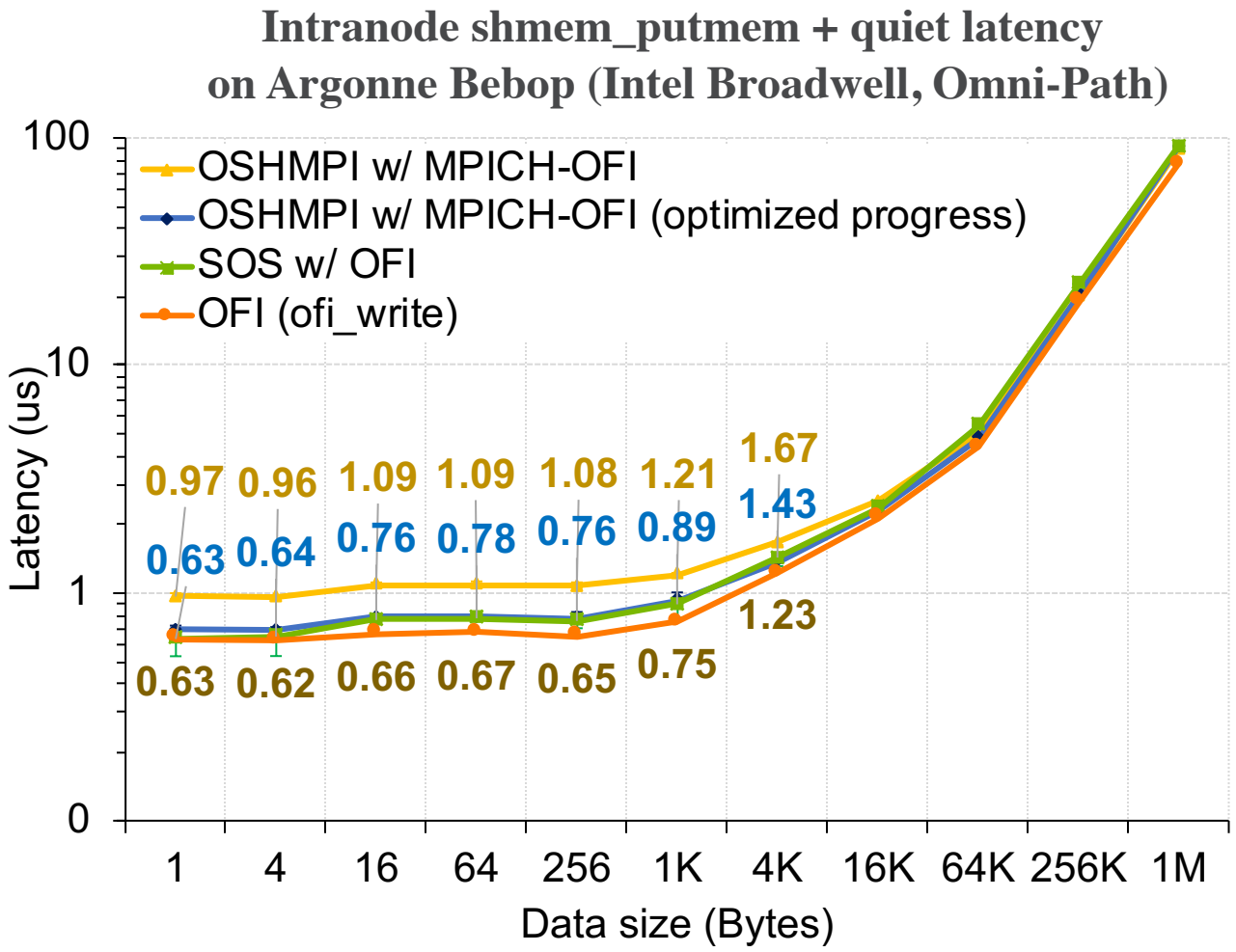
# OpenSHMEM over MPI and Challenges

- **OpenSHMEM**
  - Specialized API for fast one-sided communication
  - Directly mapping to low-level network to ensure high performance
- **MPI**
  - Low level library focusing on completeness of feature (e.g., two-sided, one-sided, collectives, various operation types, various data types)
- **OpenSHMEM over MPI ?**
  - Improve portability but may raise over-generalization issues
  - Is MPI suitable as the underlying runtime of OpenSHMEM ?
    - Performance analysis and optimizations



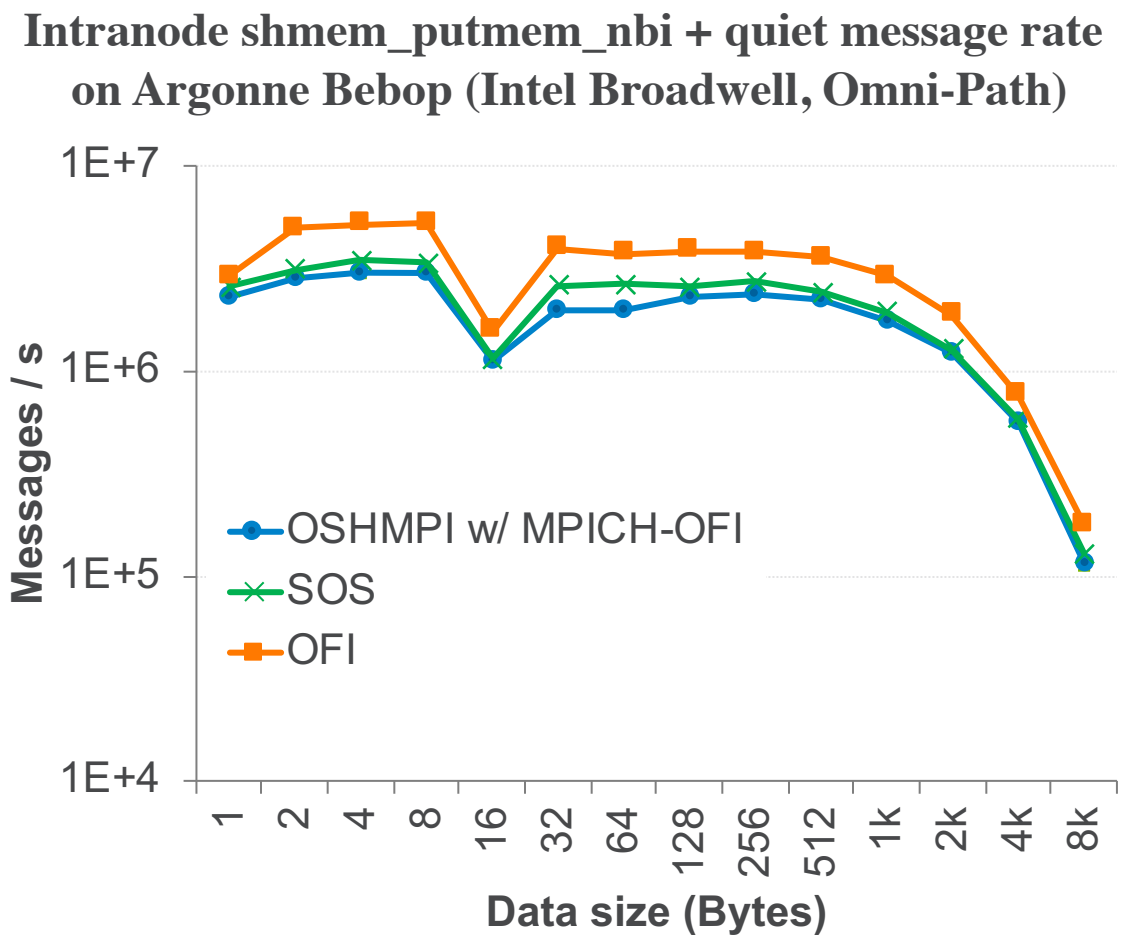
# SHMEM PUT Latency Analysis and Optimizations

- Measured intra-node latency between two processes
  - shmem_putmem + shmem_quiet
  - Inter-node shows similar trend but less gap
- OSHMPI implementation
  - shmem_putmem
    - MPI_Put + MPI_Win_flush_local
  - shmem_quiet
    - MPI_Win_flush_all(symm_heap_win)
    - MPI_Win_flush_all(symm_data_win)
    - MPI_Win_sync(symm_heap_win)
    - MPI_Win_sync(symm_data_win)
- Key bottleneck in OSHMPI/MPICH
  - MPICH always makes **"full progress"** in flush routines to ensure completion of all MPI communications (i.e., two-sided, collective, internal active message)
  - **Optimization**: if only HW RMA is involved, we can skip expensive "full progress polling" if no outstanding RMA exists.



# SHMEM Nonblocking PUT Message Rate Analysis

- Measured message rate between two processes
  - Multiple times shmem_putmem_nbi + shmem_quiet
  - Similar trend in inter-node message rate, but less gap
- OSHMPI implementation
  - shmem_putmem_nbi
    - MPI_Put
  - shmem_quiet
    - MPI_Win_flush_all(symm_heap_win)
    - MPI_Win_flush_all(symm_data_win)
    - MPI_Win_sync(symm_heap_win)
    - MPI_Win_sync(symm_data_win)
- Performance gap between OSHMPI and SOS
  - MPI internal processing overhead (e.g., MPI datatype)
  - Need instruction count level analysis (ongoing)



# Summary

- Analysis summary:
  - Focused on SHMEM PUT/GET latency and nonblocking PUT/GET message rate
  - **MPICH generic progress engine** causes high overhead
  - By **skipping the full progress polling** in MPICH when only RMA is involved, OSHMPI achieves comparable latency and message rate to SOS on an Omni-Path platform
- Software release:
  - OSHMPI 2.0b1 version has been released at SC18
  - Support OpenSHMEM 1.4
  - **Fully inlined functions** to ensure low overhead in OSHMPI layer
  - Provide **active-message based SHMEM atomic** implementation as the generic fallback
    - MPI accumulates cannot be directly used, because MPI does not support atomicity between different operations, e.g., add and cswap.
- Next step:
  - Systemically analyze internal instruction count consumption of put/get and quiet routines
  - Explore ways to safely utilize MPI accumulates for SHMEM atomics, standardize in MPI
  - Analyzing overhead of other SHMEM communication routines