Leveraging Network-level parallelism with Multiple Process-Endpoints for MPI

Amit Ruhela, Bharath Ramesh, Sourav Chakraborty, Hari Subramoni,

Jahanzeb Maqbool Hashmi, and Dhabaleswar K. (DK) Panda

E-mail: { ruhela.2, Ramesh.113, chakraborty.52, subramoni.1, hashmi.29, panda.2 } @ osu.edu

Department of Computer Science and Engineering

THE OHIO STATE UNIVERSITY
COLLEGE OF ENGINEERING
Current and Next-Generation Applications
Drivers of HPC

- Multi-core/many-core technologies
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
  - Single Root I/O Virtualization (SR-IOV)
- Accelerators (GPUs, FPGAs, Intel Xeon Phi)
- Solid State Drives (SSDs), NVM, Parallel Filesystems, Object Storage Clusters

Source: Company/Institute Websites
Drivers of HPC

Message Passing Interface (MPI) is the de-facto programming model for writing parallel applications

- MVAPICH2
- Intel MPI
- Open MPI
- Cray MPI
- IBM Spectrum MPI
- And many more...

MPI offers various communication primitives and data layouts

- One-sided Communication
- Point-to-point communication
- Collective Communication

Source: Company Websites
Goal: Design High Performance and Scalable Collective algorithm by exploiting capabilities of modern Hardware
Motivations

1. Collective operations e.g. MPI_Bcast are commonly used across parallel applications, owing to their ease of use and performance portability.

2. Processor and network architectures are constantly evolving - multi-core/many-core architectures, InfiniBand HCA, etc.

3. Existing algorithms for broadcast communication do not effectively utilize the high degree of parallelism and increased message rate capabilities offered by modern architecture.
   - Resources are underutilized

4. Essential to design new algorithms that exploits features of emerging systems and deliver good performance.
Design space of Collective Communication

Blocking and Non-Blocking Collective Algorithms

- Conventional (Flat)
  - Intra-Node Communication
    - Point-to-Point (SHMEM, LiMIC, CMA, XPMEM)
    - Direct Shared-Memory
    - Direct Kernel Assisted (CMA, XPMEM, LiMIC)

- Multi/Many-Core Aware Designs
  - Inter-Node Communication
    - Point-to-Point
    - Hardware-Mcast
    - Multi-Endpoint
Motivation: One-to-All Communication

Single pair of communication among leader nodes
Egress bandwidth not fully utilized for small messages
Early designs tried to improve inter-node communication (Next Slide)
Not efficient for small messages
MOTIVATION : One-to-All Communication

Node

Socket

P_1
P_2
...
P_K

Intra-Socket Communication

Socket

P_{K+1}
P_{K+2}
...
P_{PPN}

Intra-Socket Communication

Socket

P_{K+1}
P_{K+2}
...
P_{PPN}

Leader Communicator 1

Inter-Node Communication

Socket

P_{K+1}
P_{K+2}
...
P_{PPN}

Leader Communicator 2

Inter-Node Communication
Motivation: Multi-pair P2P

Bandwidth

SPEC MPI: Skylake + Omni-Path

Issue:
- Few pairs of communication results in reduced throughput
Motivation: One-to-All P2P Communication

Node 1

Sender Node

Node 2

Node 3

Node 4

PPN number of Sender Processes

Node 1

Node 2

Node 3

Node 4

Sender Node

Receiving Nodes

Single Sender Processes

Receiving Nodes

Node
Motivation: Inter-node Latency

- **Sender-side latency**
- **Receiver-side latency**

**Observations:**
- Both sender and receiver side latencies are inversely proportional to number of send processes on source node.
Design overview of Broadcast Communication

- Proposed Approach:
  - No change in existing intra-node algorithms
  - Leverage multi-endpoints on root-node for concurrent inter-node communication
  - Three designs proposed
    - Design 1: Provides good performance for small system size
    - Design 2: Provides scalability to Design 1
    - Design 3: Tuned version of proposed design 1 and 2, also called **Tuned HYbrid Multi-endpoint (THYME)**
Design 1- MEP Flat Inter-Node Communication

[a]: Root/Leader process copies data to the shared memory

[b]: Non-leader processes read data from shared memory

[c]: Multi-endpoints forward data to leader processes on other nodes

Non-Root nodes follow the same steps [a] and [b]
Design 2: MEP K-nomial Communication

Scalable Multi-endpoints design with degree K (=3)

[1] Root / Leader process copies data to its shared memory

[2] Non-leader processes read data from shared memory


Child nodes as well root node follow the same steps [a], [b], [c]
Design 3: Tuned HYbrid Multi-endpoint (THYME)

- Combines Design 1 and Design 2
- Scalable for various system and problem sizes
- Select algorithm based on empirical evaluations.
## Experimental Setup

<table>
<thead>
<tr>
<th>Cluster</th>
<th>Processor</th>
<th>Memory</th>
<th>Interconnect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Skylake + Omni-Path</td>
<td>2.1 GHz 24-core Intel Xeon Platinum 8160 per socket, 2 sockets, 2 hardware threads/core.</td>
<td>192GB DDR4 RAM</td>
<td>Omni-Path (100Gbps)</td>
</tr>
<tr>
<td>AMD EPYC + InfiniBand</td>
<td>2.4 GHz 32-core AMD EPYC 7551 per socket, 2 sockets, 1 threads/core</td>
<td>512GB DDR3 RAM</td>
<td>IB-EDR (100G)</td>
</tr>
<tr>
<td>OpenPOWER + InfiniBand (No Hyperthreading)</td>
<td>3.4 GHz 24-SMT4 cores Power-9 CPUs per socket, 2 sockets, 8 NUMA, 4 threads per core</td>
<td>512GB DDR3 RAM, 96GB HBM2</td>
<td>IB-EDR (100G) dual-rail</td>
</tr>
<tr>
<td>Cascade Lake + InfiniBand</td>
<td>2.7 GHz 28-core Intel Xeon 8280 per socket, 2 sockets, 2 hardware threads/core.</td>
<td>192GB DDR4 RAM</td>
<td>IB-HDR (100Gbps)</td>
</tr>
</tbody>
</table>

- Evaluations with
  - MVPAPICH2X-2.3rc2, Intel MPI 2018.0.2, Spectrum MPI v10.2.0.11rtm2
  - OSU Microbenchmarks and SPECMPI applications: MILC, SOCCORO, WRF2, ZeusMP2
High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)

- MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.1), Started in 2001, First version available in 2002 (Supercomputing ‘02)
- MVAPICH2-X (MPI + PGAS), Available since 2011
- Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2011
- Support for Virtualization (MVAPICH2-Virt), Available since 2015
- Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
- Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
- Used by more than 3,050 organizations in 89 countries
- More than 615,000 (> 0.6 million) downloads from the OSU site directly
- Empowering many TOP500 clusters (Nov ‘19 ranking)
  - 3rd, 10,649,600-core (Sunway TaihuLight) at National Supercomputing Center in Wuxi, China
  - 5th, 448, 448 cores (Frontera) at TACC
  - 8th, 391,680 cores (ABCI) in Japan
  - 14th, 570,020 cores (Nurion) in South Korea and many others
- Available with software stacks of many vendors and Linux Distros (RedHat, SuSE, and OpenHPC)
- \[http://mvapich.cse.ohio-state.edu\]
- Partner in the #5th TACC Frontera System

Empowering Top500 systems for over a decade
Impact: Message Size

Observations:
1. Up to 54% less latency than tuned broadcast algorithms in MVAPICH2-X
2. Up to 104% less latency than tuned broadcast algorithms in Intel MPI
3. Up to 44% less latency than tuned broadcast algorithms in Spectrum MPI

Network Based Computing Laboratory
Impact: Problem Size

**Observations:**

1. Up to 43% less latency against MVAPICH2 over all problem sizes
2. Up to 73% less latency against Intel MPI algorithms over all problem sizes
3. Up to 30% less latency against Spectrum MPI over all problem sizes
Impact : Applications - Spec MPI

Observations :
Up to 37% lesser latency over default MVAPICH2 broadcast algorithms
Conclusions

• Traditional designs for broadcast communication do not effectively utilize the high degree of parallelism and increased message rate capabilities offered by modern architecture.

• Proposed Multi-endpoints design that leverage multiple process endpoints to effectively use available bandwidth and deliver good performance benefits.

• Validated designs at popular Hardware configurations and against state-of-art MPI libraries which validate the strength of the proposed designs.
Thank You!

ruhela.2@cse.ohio-state.edu

Network Based Computing Laboratory

Network-Based Computing Laboratory
http://nowlab.cse.ohio-state.edu/ https://twitter.com/mvapich

The High-Performance MPI/PGAS Project
http://mvapich.cse.ohio-state.edu/

The High-Performance Big Data Project
http://hibd.cse.ohio-state.edu/

The High-Performance Deep Learning Project
http://hidl.cse.ohio-state.edu/