Chapter 5: Collective Communication Support
Outline

• Collective communication operations
• Unicast-based (Software) approach
• Path-based (Hardware) approach
• Multidestination-based schemes in direct networks
  – multicast/broadcast
  – barrier
  – reduction
• Multidestination-based schemes for MINs
  – multicast/broadcast
• NIC-based Support
• RDMA-based Support
• CORE-Direct Features of Mellanox ConnectX-2
• Mellanox SHARP support
• Multi-core-aware support (CMA, XPMEM, DPML and SALR)
• Collectives for GPU Clusters
Collective Communication Operations

- One-to-all communication
  - broadcast
  - scatter
- All-to-one communication
  - reduce (combining operators)
  - gather
- All-to-all communication
  - all-broadcast
  - complete exchange
Allreduce Collective Communication Pattern

- Element-wise Sum data from all processes and sends to all processes

```c
int MPI_Allreduce (const void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype,
                   MPI_Op operation, MPI_Comm comm)
```

### Input-only Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sendbuf</td>
<td>Starting address of send buffer</td>
</tr>
<tr>
<td>recvbuf</td>
<td>Starting address of recv buffer</td>
</tr>
<tr>
<td>type</td>
<td>Data type of buffer elements</td>
</tr>
<tr>
<td>count</td>
<td>Number of elements in the buffers</td>
</tr>
<tr>
<td>operation</td>
<td>Reduction operation to be performed (e.g. sum)</td>
</tr>
<tr>
<td>comm</td>
<td>Communicator handle</td>
</tr>
</tbody>
</table>

### Input/Output Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>recvbuf</td>
<td>Starting address of receive buffer</td>
</tr>
</tbody>
</table>
Collective Communication Operations (Cont’d)

- Convenient collective operations
  - all combining (reduce followed by broadcast)
  - barrier synchronization
  - scan (parallel prefix/suffix)
    - p1 -> p1; p1, p2-> p2; p1,p2,p3->p3; ....

- User-Level vs System Level
  - MPI has 11 different collective communication primitives (user level)
  - user-level and system-level operation may be different
  - DSM systems use collective operations in an implicit manner (system level)
Process View vs. Processor View

Process Group 1

0 1
2 3

Process Group 2

0
1 2

Process View

Processor View

0
1
2
3
2 0
2 0
1

Figure 5.6: Process view vs. processor view
Software Multicast

Figure 5.50: An example of multicast in $4 \times 4$ mesh
Dimension-ordered Chain

Figure 5.53: Multicast chain example in $4 \times 4$ mesh
Path-based Multicast

Figure 5.27: An example of dual-path multicast routing in a 6 × 6 mesh
Multi-Path Multicast

Figure 5.30: An example of multi-path multicast routing in a $6 \times 6$ mesh.
Base-Routing Conformed Path (BRCP) Model

Figure 5.36: Examples of multidestination messages on a 2-D mesh under the BRCP model
Broadcast and Multicast

Figure 5.37: Example of multicast routing in a 2-D mesh using XY routing under hierarchical leader-based scheme

Figure 5.38: Broadcast routing in a 2-D mesh using XY routing under hierarchical leader-based scheme
Broadcast

Figure 5.39: Broadcast routing in a 2-D mesh: (a) two-step broadcast using XY routing under hierarchical leader-based scheme; (b) one-step broadcast using non-minimal west-first routing
Deadlock due to Path-based Scheme

Figure 5.40: Deadlock produced by using a single delivery channel
Deadlock in BRCP Model

Figure 5.41: Deadlock produced in a 2-D mesh with three delivery channels per node. Multidestination routing uses the BRCP model and XY routing as the base routing.
How about Barrier/Reduction?

(a) propagation of a gather message

(b) propagation of a broadcasting message

- processor  - processor participating in barrier  - router interface

Figure 5.42: Two-phase implementation of a barrier on a linear array of processors (from [254])
Complete Barrier in a 2D Mesh

(a) basic scheme

(b) an enhanced scheme

Figure 5.45: Complete barrier synchronization on a $k \times k$ mesh using (a) a four step basic scheme and (b) an enhanced scheme with three steps (from [254])
Arbitrary Barrier

Figure 5.46: Barrier synchronization for an arbitrary subset of processors in a 2-D mesh using multideestination gather and broadcasting messages together with unicast message passing (from [254])
Exchange Worm

Figure 5.48: Implementation of the global combining operation on a linear array with a pair of positive and negative exchange messages (from [255])
2-step reduction/combining for 2D Mesh

Figure 5.49: Global combining on a 2-D mesh in two steps using exchange messages (from [255])
Multidestination Message Passing for MINs

Figure 5.21: Deadlock in a 16-node butterfly MIN involving multidestination messages: (a) with asynchronous message replication; (b) with synchronous message replication
Multiport Encoding

Figure 5.23: Multicast on multistage networks using multiport encoding: (a) path followed by a multidestination message; (b) message format using multiport encoding
Further Work

• Reliable Barrier
• Bit-string Encoding
• Incorporation to IBM SP Switch (ISCA ’97 and TPDS ’00 papers)
  – Asynchronous Replication
  – Bit-string Encoding
• Extended results to Irregular Networks
  – PhD Theses of Rajeev Sivaram and Ram Kesavan
    (nowlab.cse.ohio-state.edu -> publications -> PhD Dissertations)
NIC-based Support

- Path-/Tree-based Multidestination Message Passing
  - requires changes to switches/routers
  - may not be adaptable easily
- NICs are becoming intelligent on clusters
  - Implementing multidestination message passing with the help of NIC
- NIC-based
  - completely done at NIC (need to take care of flow control)
  - may not be high performance (due to slow speed of NICs)
- NIC-assisted
  - combining help from both host and NIC
  - hybrid approach
  - good cost-performance trade-off
NIC-based Support

• NIC-assisted solutions on Myrinet (by Darius)
  – Multicast/Broadcast (CANPC ’00)
  – Barrier (IPDPS ’01, CAC ’01)
  – Atomic (SAN ’02)
  – Reduction (SAN ’03)
  – Influencing the designs of next generation Myrinet adapters and MPI on Myrinet (MX)

• NIC-based Support on Quadrics (by Adam Moody)
  – Reduction (SC ’03)

• Reliable and High-Performance Collective Communication with NIC-based support (by Weikuan)
  – Reliable multicast/broadcast with Myrinet/GM-2 (ICPP ’03)
  – Broadcast support in LA-MPI with Quadrics (LACSI ’03)
  – Scalable barrier over Quadrics and Myrinet with a new collective communication processing framework (CAC ’04)
  – NIC-based all gather on Myrinet (Cluster ’04)
Efficient Collective Communication Support with RDMA, Hardware Multicast, and Atomic

- Modern networks like InfiniBand are providing
  - RDMA support (get and put)
  - Hardware multicast support
    - Based on Unreliable Datagram (UD) support currently
    - Plans for Reliable Datagram support in the next generation adapters
  - Hardware support for Atomic (fetch-and-add, compare-and-swap)
- Provides novel ways to implement collective operations with performance and scalability
  - RDMA-based barrier and reduction (Euro PVMPI ’03)
  - RDMA-based all-to-all (ICPP ’04)
  - Hardware multicast-based Broadcast (IPDPS ’04)
  - RDMA+Multicast-based barrier and reduction (Cluster ’04)
  - RDMA-based All-gather (HiPC ’05)
  - Multicast over Multiple Communicators (Euro PVM/MPI ’05)
  - Shared Memory and Multicore-aware Collectives (Euro PVM/MPI ’06, CAC ’08 CCGrid ’08, CAC ’09 and CAC ‘10)
CORE-Direct in Mellanox ConnectX-2 InfiniBand

- Collective Offload Resource Engine (CORE)
- Support is available at the NIC-level only
- Can offload multiple RDMA operations to the NIC as a bulk operation
- Allows flexible mechanisms:
  - Multi_send (same data for every send)
  - Multi_scatter (different data)
  - Recv_Multi_Send (Receive and then send the same data to others)
- Collective algorithms can be designed on top of these
- Non-Blocking Collectives in the MPI standard
- Several solutions and approaches
  - Basic mechanisms (HotI ‘10)
  - Alltoall design and impact on applications using non-blocking alltoall (ISC ‘11)
  - Broadcast design and impact on applications using non-blocking broadcast (Hoti ‘11)
  - Allreduce design and impact on applications using non-blocking reduce (IPDPS ‘12)
- Co-design Opportunities to exploit overlap
Switch-Based Support for Collectives (SHARP)

- Introduced by Mellanox
- Similar to the designs used in IBM Blue Gene, Quadrics, ..
- Hardware/Software logic within the switch to perform collective operations (reduce, barrier, etc.)
- Provides near-constant latency with increase in number of node
- Can be combined with InfiniBand HW mcast to provide good support for some collectives
- Can also be combined with Core-Direct Offload
  - Needs investigation
Offloading with Scalable Hierarchical Aggregation Protocol (SHArP)

- Management and execution of MPI operations in the network by using SHArP
  - Manipulation of data while it is being transferred in the switch network
- SHArP provides an abstraction to realize the reduction operation
  - Defines Aggregation Nodes (AN), Aggregation Tree, and Aggregation Groups
  - AN logic is implemented as an InfiniBand Target Channel Adapter (TCA) integrated into the switch ASIC *
  - Uses RC for communication between ANs and between AN and hosts in the Aggregation Tree *

More details in the tutorial “SHARPv2: In-Network Scalable Streaming Hierarchical Aggregation and Reduction Protocol” by Devendar Bureddy (NVIDIA/Mellanox)

* Bloch et al. Scalable Hierarchical Aggregation Protocol (SHArP): A Hardware Architecture for Efficient Data Reduction
Multi-core-aware Support for Collectives

- Increasing number of cores/socket and cores/node
- How to optimize collectives within a node and across the node
- Multiple solutions
  - Shared-memory-based
  - CMA-based (Chakraborty, Cluster ‘17)
  - XPMEM-based (Hashmi, IPDPS ‘18)
  - DPML (Bayatpour, SC ‘17)
  - SALR (Bayatpour, Cluster ‘18)
- Can also be combined with all other inter-node schemes
Optimized Collectives for GPU Clusters

- GPU Direct RDMA (GDR)
- Communication costs change with GDR
- Has impact on collectives
- Multiple solutions
  - GDR-based (Awan, EuroMPI ‘16)
  - Fallback through Host
  - Combining with Hardware multicast (Chu, ICPP ’17)
  - Kernel-based collectives, reduce (Chu, CCGrid ’16, CCGrid ‘20)
- Can also be combined with all other inter-node schemes
- Newer GPU architectures (AMD and Intel) coming up with new features. Provides new opportunities for optimizing collectives on these systems.
Collective Communication in MVAPICH2

Run-time flags:
- All shared-memory based collectives: MV2_USE_SHMEM_COLL (Default: ON)
- Hardware Mcast-based collectives: MV2_USE_MCAST (Default: OFF)
- CMA and XPMEM-based collectives are in MVAPICH2-X

Conventional (Flat)

Multi/Many-Core Aware Designs

Inter-Node Communication
- Point to Point
- Hardware Multicast
- SHARP
- RDMA

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