Implementation and Analysis of Nonblocking Collective Operations on SCI Networks

Christian Kaiser

Torsten Hoefler

Boris Bierbaum, Thomas Bemmerl
Scalable Coherent Interface (SCI)

Ringlet:

- IEEE Std 1596-1992
- Memory Coupled Clusters

2D Torus:

- Data Transfer: PIO and DMA
- SISCI User-Level Interface
- 16 x Intel Pentium D, 2.8 GHz
- SCI: D352 (IB: Mellanox DDR x4)
Collective Operations: GATHER

source

Process 0

Process 1 (root)

Process 2

Process 3

A

B

C

D

destination

A

B

C

D
Collective Operations: GATHERV

<table>
<thead>
<tr>
<th>source</th>
<th>destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process 0</td>
<td>A</td>
</tr>
<tr>
<td>Process 1 (root)</td>
<td>B</td>
</tr>
<tr>
<td>Process 2</td>
<td>C</td>
</tr>
<tr>
<td>Process 3</td>
<td>D</td>
</tr>
</tbody>
</table>
### Collective Operations: ALLTOALL

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process 0</td>
<td>A0 A1 A2 A3</td>
</tr>
<tr>
<td>Process 1 (root)</td>
<td>B0 B1 B2 B3</td>
</tr>
<tr>
<td>Process 2</td>
<td>C0 C1 C2 C3</td>
</tr>
<tr>
<td>Process 3</td>
<td>D0 D1 D2 D3</td>
</tr>
</tbody>
</table>
Collective Operations: ALLTOALLV

<table>
<thead>
<tr>
<th>Process 0</th>
<th>Process 1 (root)</th>
<th>Process 2</th>
<th>Process 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0 A1 A2 A3</td>
<td>B0 B1 B2 B3</td>
<td>C0 C1 C2 C3</td>
<td>D0 D1 D2 D3</td>
</tr>
<tr>
<td>A0 B0 C0 D0</td>
<td>A1 B1 C1 D1</td>
<td>A2 B2 C2 D2</td>
<td>A3 B3 C3 D3</td>
</tr>
</tbody>
</table>
Purpose:
- Study collective communication algorithms for SCI clusters
- Support multiple MPI libraries: Open MPI, NMPI
- Support arbitrary communication libraries: LibNBC
Nonblocking Collectives (NBC)

Purpose: Overlap of Computation and Communication
NBC in MPI

MPI-2.0 JoD: Split Collectives

MPI_BCAST_BEGIN(buffer, count, datatype, root, comm)

MPI_BCAST_END(buffer, comm)

MPI-2.1:

• Implement with nonblocking Point-to-Point operations
• Blocking collectives in separate thread

MPI-3 Draft:

MPI_IBCAST(buffer, count, datatype, root, comm, request)

MPI_WAIT(request, status)
Rationale: NBC for SCI

So far:
- Promising results with NBC via LibNBC
- Research done on InfiniBand clusters

Therefore:
What about a very different network architecture?

Implementation considerations:
- Use algorithms different from blocking version?
- PIO vs DMA
- Use background thread?
Available Benchmarks for LibNBC API

**Synthetic:**

**NBCBench**: measures the communication overhead / overlap potential

**Application Kernels:**

- **CG (Alltoallv):** 3D Grid, overlaps computation with halo zone exchange
- **PC (Gatherv):** overlaps compression with gathering of previous results
- **FFT (Alltoall):** parallel matrix transpose, overlaps data exchange for z transpose with computation for x and y
Gather(v)

- **Underlying concept:** Hamiltonian Path in a 2D torus
- **Algorithms:** Binary Tree, Binomial Tree, Flat Tree, Sequential Transmission

![Diagram of Hamiltonian Path in a 2D torus](image)
Gather(v):  
- Additional progress thread: Binary Tree (PIO), Binomial Tree (PIO), Flat Tree (PIO), Sequential Transmission (PIO, DMA)  
- Single Thread with manual progress: Sequential Transmission  
- Vector Variant: Flat Tree and Sequential Transmission

Alltoall(v):  
- Additional progress thread: Bruck (PIO), Pairwise Exchange (PIO), Ring (PIO), Flat Tree (PIO)  
- Single Thread with manual progress: Pairwise Exchange (DMA)  
- Vector Variant: Pairwise Exchange, Flat Tree
Application Kernels: Algorithms

Nonblocking Collectives for SCI Networks

PC (Gatherv)
CG (Alltoallv)
FFT (Alltoall)
Communication Overhead (NBCBench)

Gather

Communication Overhead (usec)

Datasize (bytes)
Communication Overhead (NBCBench)

Alltoall

Communication Overhead (usec)

Datasize (bytes)

LibNBC/NMPL/SCI
LibNBC/Open MPI/SCI
SCIColl/SCI
LibNBC/Open MPI/IB
Application Kernels: Performance

- CG (Alltoallv)
- PC (Gatherv)
- FFT (Alltoall)

Nonblocking Collectives for SCI Networks

Chair for Operating Systems
Conclusion

**What we‘ve done:**
Implement nonblocking Gather(v) and Alltoall(v) collective operations on SCI clusters with different algorithms and implementation alternatives

**What we found out:**
- Applications can benefit from nonblocking collectives on SCI clusters in spite of inferior DMA performance
- Best implementation method: DMA in a single thread, PIO is usually used for blocking collectives
- Issues with multiple threads
Thank you for your attention!