Route-aware Independent MPI I/O on the Blue Gene/Q

Preeti Malakar, Venkatram Vishwanath

Argonne Leadership Computing Facility
Argonne National Laboratory

15th November 2015
I/O trends

I/O vs. FLOPS for #1 supercomputer in top500 list

Large data
Low I/O bandwidth

Expected trend

NERSC I/O trends [Credit: www.nersc.gov]
Independent I/O

• Many applications and benchmarks use independent I/O: HACC, FLASH, Energy2, Turbulence3, MADBench2, HOMME

• More than 30% of jobs running on Mira do not use collective I/O

• Performance of independent vs. collective I/O for burst buffers and NVRAM is yet to be seen
Time to write 4 GB (independent I/O) on Mira, X-axis: Number of processes, Y-axis: Time to write from 4096 MPI processes (1 rank/node) on Mira

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
<td>0.54</td>
</tr>
<tr>
<td>Max</td>
<td>24.02</td>
</tr>
<tr>
<td>Mean</td>
<td>18.02</td>
</tr>
<tr>
<td>Std. dev.</td>
<td>4.75</td>
</tr>
</tbody>
</table>
I/O node architecture on BG/Q

- **NOT SHARED**
  - BRIDGE NODES
  - 2 GB/s
  - 128:1

- **SHARED**
  - IB NETWORK
  - 4 GB/s

**Compute node rack**
- 1024 compute nodes
- 16 bridge nodes

**I/O nodes**
- 2 bridge nodes
- connect to 1 I/O node

**GPFS filesystem**
I/O node I/O time variation

Histogram of time to write (independent I/O) from 4096 MPI processes (1 rank/node) on Mira

Statistic | Time (s)  
---|---
Min | 0.13
Max | 0.72
Mean | 0.39
Std. dev. | 0.15

Time to write 4 GB to IONs (independent I/O) from 4096 MPI processes (1 rank/node), X-axis: Number of processes, Y-axis: Time to write
I/O path on BG/Q

2 bridge nodes for every 128 compute nodes, assigned at boot time

Observations
- Multiple hops
- Few congested links
- Static routing
- Varying distance from bridge node
Network contention

Path for I/O messages from some of the compute nodes, whose assigned bridge nodes are either node 3 or node 9 on Mira.

- Unbalanced load on the network
- Few heavily loaded links
- Bridge nodes also route traffic to other bridge nodes
- Not routed to the closest bridge node
Network load balance

Can we reduce the I/O time by reducing the network congestion?
Prior work

Collective I/O optimizations
• Physical data layout-aware reordering [IPDPS 2011]
• Reorganizing I/O requests [Cluster 2014]

We focus on path of independent I/O data in the interconnect

I/O optimizations for Lustre [IPCCC 2011]
• Improved routing algorithm
• Modified pairing of client to storage servers

No control over system routing or I/O node pairing
Bridge node

Default bridge node assignment

- I/O nodes and bridge nodes are decided at boot time
- Every bridge node has equal assignment of compute nodes
  - 512-node partition: 8 bridge nodes, each assigned 64 nodes
- A compute node is not always assigned its closest bridge node

Bridge node reassignment

- Assign a node to a closer bridge node, if there exists one
- Assign equal weights (new assignments) to all bridge nodes

Compute nodes to bridge node assignments
Algorithm

Construct paths to closer bridge node

Traverse and select bridge node

Tree construction

Tree traversal
Tree construction

Find neighbors

Path from node to root in the tree?

Y

Add node

P

Path from node to root in the tree?

Y

Add node

A

Routing order

B

Routing order

C

Routing order

D

Routing order

E

Routing order

(3,1,1,0,1)

(2,1,1,0,1)

(1,1,1,1,1)

(3,1,1,1,1)

(2,1,1,1,1)

(1,1,1,1,1)
Tree traversal

Traverse every node of a level in all trees

- Current distance < default distance?
  - Y
  - Weight < average weight?
    - Y
    - Assign node
  - Weight = 4
  - Average weight = 3

Distance = Number of hops
Weight = Number of new assignments

Weight = 2
Current distance = 2

Weight = 4
Current distance = 2

Weight = 2
Implementation

User-space implementation

• Facilitates easier deployment

Modified write

• Compute nodes with default assignments
  • No change
• Compute nodes with new assignments
  • Do not write directly
  • Send data to their new bridge nodes
• Bridge nodes
  • Receive data from the newly assigned compute nodes
  • Write on behalf of the new assignees

Overhead

• Memory overhead at bridge nodes
• Time overhead for sends/receives
Experimental setup

- 512 – 8192 nodes of Mira, 1 – 16 ranks per node
- Contiguous data
- 8 KB – 4 MB writes per rank
  - I/O nodes
  - GPFS file system

- Independent
- Independent optimized
  - Data per rank
  - Data combined per node (coalesced)
- Collective
Results – Strong scaling

Time for independent writes to 4 GB file from 2048 – 32768 MPI ranks.

Mitigating interconnect contention improves performance by maximum of 15x
Results – Weak scaling

Independent writes (0.25 – 2 GB) from 4096 – 32768 MPI ranks.

Our algorithm reduces independent write times (64 KB) on average by 15x and 3x with and without combining data per node.
Results – Weak scaling (Big data)

Our algorithm scales linearly with increasing number of processes.

Independent writes from 4096 – 32768 MPI ranks.

- Default independent: 51 s
- Optimized independent: 7 s
- Coalesced optimized independent: 15 s
Results – I/O node writes

Collective and Independent 2 MB writes to I/O nodes.

Improved write times over collective MPI writes to I/O nodes.
Conclusions

• Presented a network load-aware and route-aware algorithm, implemented in user-space
• Combining data per node improves performance
• Improved performance of file system writes by an average of 60% over default MPI independent I/O on BG/Q
• Improved performance of I/O node writes by an average of 20% over default MPI collective I/O
Acknowledgements

This research has been funded in part and used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357. This work was supported in part by the DOE Office of Science, Advanced Scientific Computing Research, under award number 57L38, 57L32, 57K07 and 57K50.

Philip Heidelberger, IBM T. J. Watson Research Center
Adam Scovel, Argonne National Laboratory
SUMMARY

Our network load-aware and route-aware algorithm improved performance of file system writes by an average of 60% over default MPI independent I/O, and of I/O node writes by an average of 20% over default MPI collective I/O on BG/Q.

pmalakar@anl.gov

THANK YOU

QUESTIONS?
Results – Network counter

Average number of messages routed through a bridge node while writing 64 KB and 2 MB from 2048 nodes with 4 ranks/node (columns 2 – 3) and 8 ranks/node (columns 4 – 5).

<table>
<thead>
<tr>
<th>Data size</th>
<th>4 ranks/node</th>
<th>8 ranks/node</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Default</td>
<td>Optimized</td>
</tr>
<tr>
<td>64 KB</td>
<td>12.7</td>
<td>8.5</td>
</tr>
<tr>
<td>2 MB</td>
<td>407.7</td>
<td>289.0</td>
</tr>
</tbody>
</table>
Results – Network counter

Standard deviation in number of messages sent from bridge nodes while writing 64 KB (first three rows) and 2 MB (last three rows) data from 512, 1024 and 2048 nodes with 1 rank/node.

<table>
<thead>
<tr>
<th>Data size</th>
<th>#Nodes</th>
<th>Default</th>
<th>Optimized</th>
<th>Coalesced optimized</th>
</tr>
</thead>
<tbody>
<tr>
<td>64 KB</td>
<td>512</td>
<td>0.09</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>0.05</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>0.08</td>
<td>0.002</td>
<td>0.007</td>
</tr>
<tr>
<td>2 MB</td>
<td>512</td>
<td>0.07</td>
<td>0.02</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td>1024</td>
<td>0.11</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>2048</td>
<td>0.11</td>
<td>0.002</td>
<td>0.004</td>
</tr>
</tbody>
</table>