Benefits of Kernel-assisted Approaches on Scientific Applications: An Analysis Using LIMIC2 and CMA in MVAPICH2

Jerome Vienne
viennej@tacc.utexas.edu
MVAPICH User Group (MUG) meeting
August, 26th 2013
Content

- Introduction: *Why do we need LiMIC or CMA?*
- Presentation of LiMIC and CMA
- Performance evaluation
  - Intra-socket
  - Inter-socket
  - Collectives
  - Application level
- Conclusion
Why do we need LiMIC or CMA?

- MPI is the de facto standard for writing parallel applications.
- More and more cores inside nodes, it raises the importance/impact of intra-node communications.
- Most of MPI librairies use shared memory but…
Shared Memory

- Involved two memory copy operations
- Pro: portable solution, low latency
- Con: cache pollution for large messages, memory bandwidth and CPU cycle waste
LiMIC

- Linux Kernel Module for MPI Intra-Node Communication
- Need to be loaded
- Provides a kernel-assisted direct copy between MPI processes
- Simple interface.
- Used at TACC on Lonestar and Stampede
“Issues” with LiMIC

• Kernel crash: By passing an invalid LiMIC buffer identifier that will try to access a non-existing address space

• Security issue: it is possible to access the memory of all processes in the machine
CMA

- Introduced with Kernel 3.2
- Similar to LiMIC, but without the “issues”
- Support added in Mvapich2 1.9
- Two system calls:
  - process_vm_readv(): transfers data from the remote process to the local process.
  - process_vm_writv(): transfers data from the local process to the remote process.
Experimental setup

TACC Stampede Nodes:

• Dual-socket 8-core Intel Sandy Bridge with Turbo (latency/Bandwidth) + Quad-socket 8-core Intel SB (Collective, NAS)

• CentOS

• Compiler: Intel 13.1.0

3 different builds of MVAPICH2 (Default, LiMIC, CMA)

Benchmarks: OSU Benchmarks, NAS Class-D
Latency Intra-socket

Small Messages

Time (us)

Message Size (Bytes)

CMA
LiMIC
Default

Medium/Large Messages

Time (us)

Message Size (Bytes)

CMA
LiMIC
Default
Bandwidth Intra-socket

Small Messages

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>CMA</th>
<th>LiMIC</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>128</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>512</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2048</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8192</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Medium/Large Messages

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>CMA</th>
<th>LiMIC</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>16384</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>32768</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>65536</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>131072</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>262144</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>524288</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
<tr>
<td>1048576</td>
<td>12000</td>
<td>12000</td>
<td>12000</td>
</tr>
</tbody>
</table>
Latency Inter-socket

Small Messages

- CMA
- LiMIC
- Default

Medium/Large Messages

- CMA
- LiMIC
- Default
Bandwidth Inter-socket

Small Messages

- CMA
- LiMIC
- Default

Medium/Large Messages

- CMA
- LiMIC
- Default
Gather: Impact of Tuning

![Graph showing time (us) vs. message size (bytes) for Small and Medium/Large Messages with three different configurations: CMA, LiMIC, and Default.]
Gather: Impact of Tuning

**Small Messages**

- **CMA-tune**
- **Default**

**Medium/Large Messages**

- **CMA-Tune**
- **Default**
NAS Class D

FT: discrete 3D fast Fourier Transform, all-to-all communication

IS: Integer Sort, random memory access

MG: Multi-Grid on a sequence of meshes, long- and short-distance communication memory intensive

Done on large mem node (1TB) with 32 cores
Conclusion

• CMA and LiMIC bring significant throughput improvement to MPI communication

• Which one should I use?
  – On “old” system: LiMIC
  – On “new” system: CMA
Thank you!

For more information:
www.tacc.utexas.edu