Scalasca support for Intel Xeon Phi

Brian Wylie & Wolfgang Frings
Jülich Supercomputing Centre
Forschungszentrum Jülich, Germany
Overview

- **Scalasca** performance analysis toolset
  - support for MPI & OpenMP parallel applications
  - runtime summarization & automatic trace analysis
- **Intel Xeon Phi (MIC architecture)**
  - parallel programming/execution models
- **Case study**
  - TACC Stampede
  - symmetric BT-MZ execution on 4 compute nodes
- **Conclusion**
Scalasca

• Scalable performance analysis toolset for most popular parallel programming paradigms
  – supports MPI, OpenMP and hybrid OpenMP+MPI
• Specifically targeting large-scale parallel applications
  – such as those running on Blue Gene or Cray systems with thousands of processes or millions of threads
• Integrated instrumentation, measurement & analyses
  – runtime summarization (callpath profiling) and/or automatic event trace analysis
• Developed by JSC and GRS, available as open-source
Scalasca workflow

- Measurement library
  - HWC
  - Instr. target application
- Local event trace files
- Parallel wait-state search
- Wait-state report
- Summary report
- Optimized measurement configuration

Scalasca trace analysis

- Which problem?
- Where in the program?
- Which process?

- Instrumented executable
- Instrumenter+ compiler/linker
- Source modules

Scalasca support for Intel Xeon Phi | 2013-07-23
BGQ BT-MZ.F execution imbalance

Scalasca support for Intel Xeon Phi | 2013-07-23
Scalasca support for Intel Xeon Phi | 2013-07-23

- 16384 MPI processes with 64 OpenMP threads
- 512 s benchmark execution (<3% dilation)
- 2623 s extra for 312GB trace collection+analysis

IBM BGQ 8x16x8x8x2 torus network

Selected "Gomp do @z_solve.f52"

Coord: (A: 7, B: 15, C: 7, D: 7, E: 1, Core: 15, HWT: 3)
Node: R33-M1-N0f-J00 <7,15,7,7,1>
Name: Thread 63
MPI rank: 16383
Thread id: 63
Value: 0.00 (0.00%)
SIONlib

Portable native parallel file I/O library & utilities

- Scalable massively parallel I/O to task-local files
- Manages single or multiple physical files on disk
  - reduces meta-data server contention, optimizes bandwidth available by matching block sizes/alignment
- POSIX I/O compatible sequential & parallel API
- Tuned for common parallel filesystems (GPFS, Lustre)
- Convenient for application I/O, checkpointing
  - can be used by Scalasca tracing
- Developed by JSC, available as open-source
  - http://www.fz-juelich.de/jsc/sionlib/
Intel Xeon Phi compute nodes

Shared Virtual Memory

Twin Xeon host system

PCI Express

Compute node

Scalasca support for Intel Xeon Phi | 2013-07-23
# Intel Xeon Phi (MIC)

<table>
<thead>
<tr>
<th>Host</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xeon E5-2680 (processor)</td>
<td>Xeon Phi SE10P (coprocessor)</td>
</tr>
<tr>
<td>2x sockets</td>
<td>2x PCI Express cards</td>
</tr>
<tr>
<td>8x 2.7 GHz cores, each 2-way HT x86_64 (cpu)</td>
<td>61x 1.1 GHz cores, each 4-way HT mic (native)</td>
</tr>
<tr>
<td>32 GB shared virtual memory</td>
<td>8 GB local memory per card</td>
</tr>
<tr>
<td>Intel Composer XE 2013 (13.1)</td>
<td>N/A</td>
</tr>
<tr>
<td>• normal compile &amp; link</td>
<td>• -mmic cross-compile &amp; link</td>
</tr>
<tr>
<td>Intel MPI 4.1</td>
<td>Intel MPI 4.1</td>
</tr>
</tbody>
</table>

**MIC** = Many Integrated Core architecture (aka *manycore*)

**Xeon Phi** = Intel's first product (aka *Knights Corner*)
### MIC parallel programming models

<table>
<thead>
<tr>
<th>Host</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI</td>
<td>MPI</td>
</tr>
<tr>
<td>+ OpenMP</td>
<td>+ OpenMP</td>
</tr>
<tr>
<td>+ OpenCL</td>
<td>+ OpenCL</td>
</tr>
<tr>
<td>+ pthreads</td>
<td>+ pthreads</td>
</tr>
<tr>
<td>+ F2008</td>
<td>+ F2008</td>
</tr>
<tr>
<td>+ TBB [C/C++]</td>
<td>+ TBB [C/C++]</td>
</tr>
<tr>
<td>+ Cilk Plus [C++]</td>
<td>+ Cilk Plus [C++]</td>
</tr>
</tbody>
</table>

**symmetric** = host MPI[+MT] + device MPI[+MT]

**offload** = automatic or compiler-assisted kernel execution on device or host

**LEO** = language extension for offload (pragmas/directives similar to OpenMP)
Scalasca usage (basic)

<table>
<thead>
<tr>
<th>Host</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>skin mpiifort -O -openmp *.f</td>
<td>skin mpiifort -O -openmp -mmic *.f</td>
</tr>
<tr>
<td>scan mpiexec -n 2 a.out.cpu</td>
<td>ssh mic0 \</td>
</tr>
<tr>
<td></td>
<td>-c &quot;scan mpiexec -n 61 a.out.mic&quot;</td>
</tr>
<tr>
<td></td>
<td>scan mpiexec.hydra \</td>
</tr>
<tr>
<td></td>
<td>-host node0 -n 1 a.out.cpu \</td>
</tr>
<tr>
<td></td>
<td>: -host node1 -n 1 a.out.cpu</td>
</tr>
<tr>
<td>scan mpiexec.hydra \</td>
<td>scan mpiexec.hydra \</td>
</tr>
<tr>
<td></td>
<td>-host mic0 -n 30 a.out.mic \</td>
</tr>
<tr>
<td></td>
<td>: -host mic1 -n 31 a.out.mic</td>
</tr>
<tr>
<td>square epik_a_2x16_sum</td>
<td>square epik_a_mic61x4_sum</td>
</tr>
</tbody>
</table>

Symmetric execution:
scan mpiexec.hydra -host node0 -n 2 a.out.cpu : -host mic0 -n 61 a.out.mic
square epik_a_2x16+mic61x4_sum
Scalasca case study configuration

TACC Stampede Dell PowerEdge C8220
• 6,400 compute nodes (2x Xeon + [012]x Xeon Phi)
• 4 compute node partition with 1x MIC for experiment

NPB3.3-MZ-MPI class D: bt-mz_D.68[.cpu|.mic]

SLURM_TASKS_PER_NODE=2(x4)
MIC_PPN=15
MIC_OMP_NUM_THREADS=OMP_NUM_THREADS=16
scan -t ibrun.symm -c bt-mz_D.68.cpu -m bt-mz_D.68.mic

 ➔ epik_bt-mz_D_2p8x16+mic15p60x16_trace
Scalasca usage (stampede ibrun.symm)

ELG_SION_FILES=-1  SCAN_ANALYZE_OPTS="-s -i"
scan  -t  ibrun.symm -c bt-mz_D.68.cpu -m bt-mz_D.68.mic
S=C=A=N: Scalasca 1.4.3 trace collection and analysis
S=C=A=N: ./epik_bt-mz_D_2p8x16+mic15p60x16_trace experiment archive
ibrun.symm -c "bt-mz_D.68.cpu" -m "bt-mz_D.68.mic"
[00000.0]EPIK: Created archive ./epik_bt-mz_D_2p8x16+mic15p60x16_trace
[00000.0]EPIK: ELG_SION_FILES=63 determined automatically
...
[00000.0]EPIK: Closed experiment ./epik_bt-mz_D_2p8x16+mic15p60x16_trace
S=C=A=N: Collect done
S=C=A=N: Analyze start
ibrun.symm -c "$SCALASCA_DIR/bin/fe/scout.hyb -s -i"
   -m "$SCALASCA_DIR/bin/be/scout.hyb -s -i"
Analyzing experiment archive ./epik_bt-mz_D_2p8x16+mic15p60x16_trace
...
S=C=A=N: ./epik_bt-mz_D_2p8x16+mic15p60x16_trace experiment complete
Scalasca experiment characteristics

<table>
<thead>
<tr>
<th>Activity</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference (uninstrumented) execution</td>
<td>325.41 s</td>
</tr>
<tr>
<td>Summary measurement execution</td>
<td>330.47 s [+1.6%]</td>
</tr>
<tr>
<td>Summary report generation</td>
<td>0.49 s</td>
</tr>
<tr>
<td>Tracing measurement execution</td>
<td>330.98 s [+1.7%]</td>
</tr>
<tr>
<td>Event trace generation</td>
<td>44.55 s</td>
</tr>
<tr>
<td>Event trace analysis (complete)</td>
<td>113.16 s</td>
</tr>
</tbody>
</table>

Symmetric execution of BT-MZ class D configured with 68 MPI processes and 16 OpenMP threads per process on four *Stampede* Xeon Phi compute nodes. Custom instrumentation filter for Intel compiler (from preliminary score report). 5.08 GB event trace data (buffered). One SION event trace file per process.
Scalasca BT-MZ case study insight

- 75% of total allocation time is actual computation
- 15% of time considered to be *Idle threads*
  - unused compute resources outside of parallel regions
- 1% of total time in MPI
  - similarly done outside of parallel regions
- 9% of total time in OpenMP parallelization overheads
  - 6.5% *OpenMP implicit barrier synchronization time* at end of parallel regions
    - 70% of this within *Z_solve* routine (lines 43-428)
    - unevenly distributed across threads *on devices*, though threads on hosts apparently well balanced
    - maximum 43 seconds, mean $15.4 \pm 11.0$ seconds
Current and future work

- Extend support for symmetric/heterogeneous MPMD measurement collection and analysis
  - mpiexec.hydra
  - consistency checks
  - additional metrics?
- Migration to new Score-P measurement infrastructure
  - support for additional threading models
  - support for OpenMP target offload
Conclusions

- Port of Scalasca toolset to MIC was straightforward
  - configuration currently somewhat complicated due to evolving/maturing environment
  - provides familiar usage model
- Measurement and analysis of symmetric execution exploits combined host+device installation
  - smart mode determination
- Facilitates performance analysis for tuning/optimization of MPI and/or OpenMP
Acknowledgments

- XSEDE (Jay Alameda)
- TACC (Stampede)
- Intel/JSC ExaCluster Laboratory
- DEEP Project (EU FP7)
- Scalasca development team
Scalable performance analysis of large-scale parallel applications

- toolset for scalable performance measurement & analysis of MPI, OpenMP and MPI+OpenMP parallel applications
- supporting most popular HPC computer systems
- available under New BSD open-source license
- sources, documentation & publications:
  - [http://www.scalasca.org](http://www.scalasca.org)
  - [mailto: scalasca@fz-juelich.de](mailto:scalasca@fz-juelich.de)