What's new in Allinea's tools
From easy batch script integration & remote access to energy profiling
Introduction Agenda

- Overview of HPC current and future needs
- What’s new in Allinea’s tools
  - Transitioning from CPU Hours to Power Usage
  - Enabling easy remote application development
- What is coming next
  - Reducing trapped capacity on next-gen processors
  - Coming Allinea “good stuff”
HPC Ultimate target

COMPUTER SIMULATIONS ➔ PREDICTIVE SCIENCE
Example: Weather and Forecasting models
Demands are evolving

<table>
<thead>
<tr>
<th>Scalability</th>
<th>Efficiency</th>
<th>Simplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Enable multi-physics simulations</td>
<td>• Maximize science output per $</td>
<td>• Readiness of applications on HPC platforms</td>
</tr>
<tr>
<td>• Run larger, more accurate models</td>
<td>• Minimize time to result</td>
<td>• Minimize learning curve for HPC users</td>
</tr>
<tr>
<td>• Resolve ground-breaking scientific problems</td>
<td>• Monitor and reduce wasted resources (energy..)</td>
<td>• Facilitate dialogue with scientific communities</td>
</tr>
</tbody>
</table>
Allinea’s vision

• Helping maximize HPC production
  • Reduce HPC systems operating costs
  • Resolve cutting-edge challenges
  • Promote Efficiency (as opposed to Utilization)
  • Transfer knowledge to HPC communities

• Helping the HPC community design the best applications
  • Reach highest levels of performance and scalability
  • Improve scientific code quality and accuracy
Development process: tools-centric view

Demand for software efficiency

FORGE

ANALYZE
(Allinea Performance Reports)

Open Interfaces (e.g. JSON APIs)

Demand for developer efficiency

FORGE

PERF OPTIMIZATION
(Allinea MAP)

Continuous Integration (e.g. Jenkins, etc.)

Debug/optimize, edit, commit, build, repeat

Version Control (e.g. CVS, etc…)

DEBUGGING
(Allinea DDT)

New Version

DB
Allinea MAP and CrayPAT: a great synergy

Simple optimization with Allinea MAP
- Characterize performance at-scale with a simple GUI tool
- See which lines of code are hotspots
- Identify common problems at once

Prepare optimization strategy with Allinea MAP
- Document performance issues to communicate to CrayPAT experts
- Identify loop(s) to instrument with CrayPAT API
- Identify CrayPAT performance counter(s) to record

Fine tune the code with CrayPAT
- Retrieve low-level details with CrayPAT
- Fix up CPU usage to make the code fly
Demands are evolving

- Scalability
- Efficiency
- Simplicity
Allinea Remote Client: fast interactive debugging
Reverse connect: the end of template files

[patrick@allinea-demo 3_fix]$ ddt&
[1] 3761
[patrick@allinea-demo 3_fix]$ ls
job.sub  mmult3.c  mmult3.f90  mmult3.sub
Makefile  mmult3_c.exe  mmult3_f90.exe
[patrick@allinea-demo 3_fix]$ cat job.sub
#!/bin/bash
#PBS -l walltime=8:00:00,nodes=1:ppn=8,pmem=1000mb
#PBS -N test
module load openmpi/1.6.5

ddt --connect mpirun -n 8 mmult3.exe
[patrick@allinea-demo 3_fix]$ qsub job.sub
Job 3248 has been submitted.
[patrick@allinea-demo 3_fix]$
Towards better efficiency with Allinea’s tools

- Scalability
  - Enable multi-physics simulations
  - Run larger, more accurate models
  - Resolve ground-breaking scientific problems
- Efficiency
- Simplicity
  - Readiness of applications on HPC platforms
  - Minimize learning curve for HPC users
  - Facilitate dialogue with scientific communities
Energy efficiency with Allinea’s tools

Energy

A breakdown of how the 3.6 Wh was used:

- CPU: 62.9%
- System: 37.1%
- Mean node power: 92.4 W
- Peak node power: 94 W

Significant energy is wasted during MPI communications. It may be more efficient to use fewer nodes with more data on each node.

Significant time is spent waiting for memory accesses. Reducing the CPU clock frequency could reduce overall energy usage.
Quantify gains immediately

### CPU RUN

**Energy**
A breakdown of how the 3.6 Wh was used:
- CPU: 62.9%
- System: 37.1%
- Mean node power: 92.4 W
- Peak node power: 94 W

Significant energy is wasted during MPI communications. It may be more efficient to use fewer nodes with more data on each node.

**Accelerators**
A breakdown of how accelerators were used:
- GPU utilization: 92.5%
- Global memory accesses: 40.4%
- Mean GPU memory usage: 9.6%
- Peak GPU memory usage: 15.2%

Significant time is spent in global memory accesses. Try modifying kernels to use shared memory instead and check for bad striding patterns.

The peak device memory usage is low, it may be more efficient to offload a larger portion of the dataset to each device.

### GPU RUN

**Energy**
A breakdown of how the 2.84 Wh was used:
- CPU: 28.4%
- System: 71.6%
- Mean node power: 175.8 W
- Peak node power: 163 W

Most of the time is spent waiting for accelerators. Use asynchronous transfers. The per-core performance is memory-bound. Use a profiler to identify time-consuming memory accesses. No time is spent in vectorization advisory.

**Accelerators**
A breakdown of how accelerators were used:
- GPU utilization: 0.0%
- Global memory accesses: 0.0%
- Mean GPU memory usage: 0.0%
- Peak GPU memory usage: 0.0%

GPUs are available but are not used. Identify suitable hot loops with a profiler and try offloading them to the accelerator.

The peak device memory usage is low, it may be more efficient to offload a larger portion of the dataset to each device.
Custom metrics with Allinea MAP

Hardware (system + CPU) provides energy-related data (Currently: IPMI-based power sensors). API extracts this data and feeds Allinea's tools (Currently: Intel Energy Checker SDK, RAPL, data in memory, our in-house daemon).

Allinea's tools process data at runtime to bring unique perspective.
Reducing trapped capacity on next-gen processors

<table>
<thead>
<tr>
<th>Scalability</th>
<th>Efficiency</th>
<th>Simplicity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximize science output per $</td>
<td>Minimize time to result</td>
<td>Monitor and reduce wasted resources (energy..)</td>
</tr>
<tr>
<td>Readiness of applications on HPC platforms</td>
<td>Minimize learning curve for HPC users</td>
<td>Facilitate dialogue with scientific communities</td>
</tr>
</tbody>
</table>
Towards more vertical & horizontal scalability

INTEL KNIGHTS LANDING

NVIDIA GPUS

FPGAs

NEXT-GEN INTEL XEON

ARM v8

allinea
Towards more vertical & horizontal scalability

INTEL KNIGHTS LANDING
NVIDIA GPUS
FPGAs

NEXT-GEN INTEL XEON
ARM v8

PORTABLE PERFORMANCE REQUIRED
Support for next generation systems

• **ARMv8 Support**
  – Allinea Forge already available
  – Allinea Performance Reports for ARMv8 scheduled (H2020 ExaNest)

• **Nvidia GPUs Support (currently supported: CUDA 7.5)**
  – CUDA 8.0 expected around GTC 2016

• **Intel KNL Support**
  – Will be supported by Allinea at release
What is coming next? (more under NDA)

**Allinea DDT**
- Advanced memory debugging for Intel Knight’s Landing
- New HTML crash and leak reports
- Export debugging data to continuous integration tools (Jenkins, Bamboo etc)

**Allinea MAP**
- Profile selected ranks only
- Toggle view between absolute times & percentages
- Export function-level performance to continuous integration tools

**Allinea Performance Reports**
- Profile selected ranks only
- Toggle view between absolute times & percentages
- Add custom section to your own reports
- Export metrics to continuous integration tools
Summary: Allinea provides...

- Leadership in HPC software tools
  - The only scalable tool suite for HPC

- Active development and investment
  - Innovative, extremely scalable tools with unique visual insight

- Rapid innovation and user focus
  - Providing what science needs, first.
Thank you!

Technical Support team: support@allinea.com
Sales team: sales@allinea.com