Virtual Institute – High Productivity Supercomputing

9-10 August 2012

Brian Wylie
Jülich Supercomputing Centre
b.wylie@fz-juelich.de
Thursday 9 August

- 09:00 (start)
- Performance analysis and tools overview (Geimer, JSC)
- Accessing & using CSCS resources
- 10:30-10:45 (break)
- Profile & automatic trace analysis with Scalasca (Wylie/Geimer, JSC)
- 12:30-13:30 (lunch)
- Trace analysis with Vampir (Knüpfer, TU Dresden)
- 15:00-15:15 (break)
- Assisted analysis & tuning of participants' codes
- 17:30 (adjourn)
We'd like to know a little about you, your application(s), and your expectations and desires from this tutorial

- What programming paradigms do you use in your app(s)?
  - only MPI, only OpenMP, mixed-mode/hybrid OpenMP/MPI, ...
  - Fortran, C, C++, multi-language, ...

- What platforms/systems must your app(s) run well on?
  - Cray XT/XE/XK, IBM BlueGene, SGI Altix, Linux cluster™, ...

- Who's already familiar with serial performance analysis?
  - Which tools have you used?
    - time, print/printf, prof/gprof, ...

- Who's already familiar with parallel performance analysis?
  - Which tools have you used?
    - time, print/printf, prof/gprof, Periscope, Scalasca, TAU, Vampir, ...
Prepare to analyse your own application codes

- Ensure your application codes build and run to completion with appropriate datasets
  - initial configuration should ideally run in less than 15 minutes with 1-4 compute nodes (up to 64 processes/threads)
    - to facilitate rapid turnaround and quick experimentation
  - larger/longer scalability configurations are also interesting
    - turnaround may be limited due to busyness of batch queues

- Compare your application performance on other systems
  - VI-HPS tools already installed on a number of HPC systems
    - if not, ask your system administrator to install them
      (or install a personal copy yourself)
**Goal**: Improve the quality and accelerate the development process of complex simulation codes running on highly-parallel computer systems

- Start-up funding (2006-2011) by Helmholtz Association of German Research Centres

- Activities
  - Development and integration of HPC programming tools
    - Correctness checking & performance analysis
  - Training workshops
  - Service
    - Support email lists
    - Application engagement
  - Academic workshops

[www.vi-hps.org](http://www.vi-hps.org)
VI-HPS partners (founders)

Forschungszentrum Jülich
- Jülich Supercomputing Centre

RWTH Aachen University
- Centre for Computing & Communication

Technical University of Dresden
- Centre for Information Services & HPC

University of Tennessee (Knoxville)
- Innovative Computing Laboratory
Barcelona Supercomputing Center
- Centro Nacional de Supercomputación

German Research School
- Laboratory of Parallel Programming

Lawrence Livermore National Lab.
- Centre for Applied Scientific Computing

Technical University of Munich
- Chair for Computer Architecture

University of Oregon
- Performance Research Laboratory

University of Stuttgart
- HPC Centre

University of Versailles St-Quentin
- LRC ITACA
VI-HPS productivity tools

- **Marmot/MUST**
  - MPI correctness checking
- **PAPI**
  - Interfacing to hardware performance counters
- **Periscope**
  - Automatic analysis driven by on-line distributed search
- **Scalasca**
  - Large-scale parallel performance analysis
- **TAU**
  - Integrated parallel performance system
- **Vampir/VampirTrace**
  - Event tracing and graphical trace visualization & analysis
- **Score-P**
  - Common instrumentation & measurement infrastructure
VI-HPS productivity tools (cont.)

- **KCacheGrind**
  - Callgraph-based cache analysis [x86 only]
- **MAQAO**
  - Assembly instrumentation & optimization [x86 only]
- **mpiP/mpiPview**
  - MPI profiling tool and analysis viewer
- **ompP**
  - OpenMP profiling tool
- **OpenMPI**
  - Memory checking
- **Open|SpeedShop**
  - Integrated parallel performance analysis environment
- **Paraver/Extrae**
  - Event tracing and graphical trace visualization & analysis
Technologies and their integration

- PAPI
- KCACHEGRIND
- Hardware monitoring
- Automatic profile & trace analysis
- Error detection
- Visual trace analysis
- Execution
- Optimization
- SCALASCA
- VAMPIR / PARAVER
- MARMOT / MUST
- PERISCOPE
- TAU
- SIONLIB / OPENMPI
- MAQAO
Tools will *not* automatically make you, your applications or computer systems more *productive*. However, they can help you understand *how* your parallel code executes and *when / where* it's necessary to work on *correctness* and *performance* issues.
VI-HPS Training & Tuning Workshops

- Goals
  - Give an overview of the programming tools suite
  - Explain the functionality of individual tools
  - Teach how to use the tools effectively
  - Offer hands-on experience and expert assistance using tools
  - Receive feedback from users to guide future development

- For best results, bring & analyse/tune your own code(s)!

- VI-HPS Tutorial series
  - SC'08, ICCS'09, SC'09, Cluster'10, SC'10, SC'11, EuroMPI'12

- VI-HPS Tuning Workshop series
  - 2008 (Aachen & Dresden), 2009 (Jülich & Bremen), 2010 (Garching & Amsterdam), 2011 (Stuttgart & Aachen)
  - 2012/04/23-27 (St-Quentin), 2012/10/15-19 (Garching)
Upcoming VI-HPS training events

- EuroMPI hands-on tutorial (23 Sep 2012, Vienna, Austria)
  - Periscope, Scalasca, TAU, Vampir (using Score-P)
- 10th VI-HPS Tuning Workshop (15-19 Oct 2012)
  - hosted by LRZ, Garching-bei-München, Germany
  - using PRACE Tier-0 SuperMUC iDataPlex system
  - Scalasca, Vampir, TAU, Periscope, KCachegrind, MAQAO, ...
- Further events to be determined
  - (one-day) tutorials
    - with guided exercises usually using Live DVD
  - (multi-day) training workshops
    - with your own applications on real HPC systems

Check www.vi-hps.org/training for announced events
- Contact us if you might be interested in hosting an event
• Bootable Linux installation ISO (on DVD or USB drive)
• Includes everything needed to try out our parallel tools on an x86-architecture notebook computer
  □ VI-HPS tools: KCachegrind, Marmot, PAPI, Periscope, Scalasca, TAU, VT/Vampir*
  □ Also: Eclipse/PTP, TotalView*, etc.
    ▶ * time/capability-limited evaluation licences provided for commercial products
□ GCC (w/ OpenMP), OpenMPI
□ Manuals/User Guides
□ Tutorial exercises & examples
• Produced by U. Oregon PRL
□ Sameer Shende
• ISO image approximately 4GB
  ■ distributed on DVD or USB drive
  ■ or download from website
• Boot directly from disk
  ■ enables hardware counter access and offers best performance
• Boot within virtual machine
  ■ faster boot time and can save/resume state, but no hardware counter access
• Boots into Linux environment
  ■ supports building and running provided MPI and/or OpenMP parallel application codes
  ■ and experimentation with VI-HPS (and other) tools
Cachegrind: cache analysis by simple cache simulation
- Captures dynamic callgraph
- Based on valgrind dynamic binary instrumentation
- Runs on x86/PowerPC/ARM unmodified binaries
  - No root access required
- ASCII reports produced

[KQ] Cachegrind GUI
- Visualization of cachegrind output

Developed by TU Munich
- Released as GPL open-source
- http://kcachegrind.sf.net/
KCacheGrind GUI

- Event cost tree map
- Source code view
- Call graph view
- Machine code annotation
Tool to check for correct MPI usage at runtime

- Checks conformance to MPI standard
  - Supports Fortran & C bindings of MPI-1.2
- Checks parameters passed to MPI
- Monitors MPI resource usage

Implementation

- C++ library gets linked to the application
- Does not require source code modifications
- Additional process used as DebugServer
- Results written in a log file (ASCII/HTML/CUBE)

Developed by HLRS & TU Dresden

- Released as open-source
- http://www.hlrs.de/organization/av/amt/projects/marmot
### Marmot Logfiles

1 (localhost.localdomain)
for MPI-Standard information see: /usr/local/packages/marmot-2.3.0/share/doc/marmot-2.3.0/MPI-STD/MARMOT_logfile_readme

3: Warning global message with Text: Processes 0 and 1 both run on localhost.

for MPI-Standard information see: /usr/local/packages/marmot-2.3.0/share/doc/marmot-2.3.0/MPI-STD/MARMOT_logfile_readme

10: Error from rank 0 (Thread: 0) with Text: ERROR: MPI_Send: datatype is not valid!
On Call: MPI_Send From: datatype c line: 53 for MPI-Standard information see: /usr/local/packages/marmot-2.3.0/share/doc/marmot-2.3.0/MPI-STD/MARMOT_logfile_readme

10: Error from rank 1 (Thread: 0) with Text: ERROR: MPI_Send: datatype is not valid!
On Call: MPI_Send From: datatype c line: 53 for MPI-Standard information see: /usr/local/packages/marmot-2.3.0/share/doc/marmot-2.3.0/MPI-STD/MARMOT_logfile_readme

<table>
<thead>
<tr>
<th>Rank</th>
<th>Global</th>
<th>Local</th>
<th>Level</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: MARMOT_MAX_TIMEOUT_ONE = 0 (maximum message time, default: 0 microseconds)</td>
</tr>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: MARMOT_MAX_TIMEOUT_TWO = 0 (maximum message time, default: 0 microseconds)</td>
</tr>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: MARMOT_LOGFILE_PATH = (path of Marmot logfile output, default: )</td>
</tr>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: MARMOT_ERRCODES_SET = (not set) (not functional yet)</td>
</tr>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: End of the environmental variables info.</td>
</tr>
<tr>
<td>0</td>
<td>Global</td>
<td>0</td>
<td>Information</td>
<td>Text: Thread Synchronisation is disabled. If you are using multiple threads errors might occur</td>
</tr>
<tr>
<td>3</td>
<td>Global</td>
<td>0</td>
<td>Warning</td>
<td>Text: Debugserver runs on same node as process 0 (localhost.localdomain)</td>
</tr>
<tr>
<td>3</td>
<td>Global</td>
<td>0</td>
<td>Warning</td>
<td>Text: Debugserver runs on same node as process 1 (localhost.localdomain)</td>
</tr>
<tr>
<td>3</td>
<td>Global</td>
<td>0</td>
<td>Warning</td>
<td>Text: Processes 0 and 1 both run on localhost.localdomain</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>Error</td>
<td>Text: ERROR: MPI_Send: datatype is not valid!</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>0</td>
<td>Error</td>
<td>Text: ERROR: MPI_Recv: datatype is not valid!</td>
</tr>
</tbody>
</table>
Next generation MPI runtime error detection tool

- Successor of the Marmot and Umpire tools
- Initial merge of Marmot's many local checks with Umpire's non-local checks
- Improved scalability expected in future
- Exploits CMake, GTI & PnMPI infrastructure

Developed by TU Dresden, LLNL & LANL

- BSD license open-source initial release in November 2011
- http://tu-dresden.de/zh/must/
Portable performance counter library & utilities

- Configures and accesses hardware/system counters
- Predefined events derived from available native counters
- Core component for CPU/processor counters
  - instructions, floating point operations, branches predicted/taken, cache accesses/misses, TLB misses, cycles, stall cycles, ...
  - performs transparent multiplexing when required
- Extensible components for off-processor counters
  - InfiniBand network, Lustre filesystem, system hardware health, ...
- Used by multi-platform performance measurement tools
  - Periscope, Scalsaca, TAU, VampirTrace, ...

Developed by UTK-ICL

- Available as open-source for most modern processors
  http://icl.cs.utk.edu/papi/
Available events and hardware information.

-----------------------------------------------
<table>
<thead>
<tr>
<th>Name</th>
<th>Code</th>
<th>Count</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAPI_L1_DCM</td>
<td>0x80000000</td>
<td>1</td>
<td>Level 1 data cache misses</td>
</tr>
<tr>
<td>PAPI_L1_ICM</td>
<td>0x800000001</td>
<td>1</td>
<td>Level 1 instruction cache misses</td>
</tr>
</tbody>
</table>

Of 107 possible events, 35 are available, of which 9 are derived.
Available native events and hardware information.

<table>
<thead>
<tr>
<th>Event Code</th>
<th>Symbol</th>
<th>Long Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x40000000</td>
<td>UNHALTED_CORE_CYCLES</td>
<td>count core clock cycles whenever the clock signal on the specific core is running (not halted). Alias to event CPU_CLK_UNHALTED:THREAD</td>
</tr>
<tr>
<td>0x40000001</td>
<td>INSTRUCTION_RETIRED</td>
<td>count the number of instructions at retirement. Alias to event INST_RETIRED:ANY_P</td>
</tr>
<tr>
<td>0x40000086</td>
<td>UNC_SNP_RESP_TO_REMOTE_HOME</td>
<td>Remote home snoop response - LLC does not have cache line</td>
</tr>
<tr>
<td>40000486</td>
<td>:I_STATE</td>
<td>Remote home snoop response - LLC does not have cache line</td>
</tr>
<tr>
<td>40000886</td>
<td>:S_STATE</td>
<td>Remote home snoop response - LLC has cache line in S state</td>
</tr>
<tr>
<td>40001086</td>
<td>:FWD_S_STATE</td>
<td>Remote home snoop response - LLC forwarding cache line in S state.</td>
</tr>
<tr>
<td>40002086</td>
<td>:FWD_I_STATE</td>
<td>Remote home snoop response - LLC has forwarded a modified cache line</td>
</tr>
<tr>
<td>40004086</td>
<td>:CONFLICT</td>
<td>Remote home conflict snoop response</td>
</tr>
<tr>
<td>40008086</td>
<td>:WB</td>
<td>Remote home snoop response - LLC has cache line in the M state</td>
</tr>
<tr>
<td>40010086</td>
<td>:HITM</td>
<td>Remote home snoop response - LLC HITM</td>
</tr>
</tbody>
</table>

Total events reported: 135
Automated profile-based performance analysis

- Iterative on-line performance analysis
  - Multiple distributed hierarchical agents
- Automatic search for bottlenecks based on properties formalizing expert knowledge
  - MPI wait states, OpenMP overheads and imbalances
  - Processor utilization hardware counters
- Clustering of processes/threads with similar properties
- Eclipse-based integrated environment

Supports

- SGI Altix Itanium2, IBM Power and x86-based architectures

Developed by TU Munich

- Released as open-source
- http://www.lrr.in.tum.de/periscope
MPI
- Excessive MPI communication time
- Excessive MPI time due to many small messages
- Excessive MPI time in receive due to late sender
- ...

OpenMP
- Load imbalance in parallel region/section
- Sequential computation in master/single/ordered region
- ...

Hardware performance counters (platform-specific)
- Cycles lost due to cache misses
  ▶ High L1/L2/L3 demand load miss rate
- Cycles lost due to no instruction to dispatch
- ...

Periscope properties & strategies (examples)
Periscope plug-in to Eclipse environment
Automatic performance analysis toolset

- Scalable performance analysis of large-scale applications
  - particularly focused on MPI & OpenMP paradigms
  - analysis of communication & synchronization overheads
- Automatic and manual instrumentation capabilities
- Runtime summarization and/or event trace analyses
- Automatic search of event traces for patterns of inefficiency
  - Scalable trace analysis based on parallel replay
- Interactive exploration GUI and algebra utilities for XML callpath profile analysis reports

Developed by JSC & GRS

- Released as open-source

http://www.scalasca.org/
Late Sender Time

Description:

Refers to the time lost waiting caused by a blocking receive operation (e.g., MPI_Recv) or MPI_Wait() that is posted earlier than the corresponding send operation.
Integrated performance toolkit

- Instrumentation, measurement, analysis & visualization
  - Highly customizable installation, API, envvars & GUI
  - Supports multiple profiling & tracing capabilities
- Performance data management & data mining
- Targets all parallel programming/execution paradigms
  - Ported to a wide range of computer systems
- Performance problem solving framework for HPC
- Extensive bridges to/from other performance tools
  - PerfSuite, Scalasca, Vampir, ...

Developed by U. Oregon/PRL

- Broadly deployed open-source software
- http://tau.uoregon.edu/
TAU ParaProf GUI displays (selected)

TAU: ParaProf Manager

File Options Windows Help

Applications
- Standard Applications
- Default App
- Default Exp

epik_bt-mz_B_4x4_trace_PAT_RT_HWPC_0ft

TAU: ParaProf: epik_bt-mz_B_4x4_trace_PAT_RT_HWPC_0ft

File Options Windows Help

Name: main => MAIN_ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
Metric Name: Time
Value: Exclusive
Units: seconds

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>node 1, thread 2</th>
<th>node 1, thread 0</th>
<th>node 1, thread 1</th>
<th>node 3, thread 0</th>
<th>node 3, thread 2</th>
<th>node 3, thread 1</th>
<th>node 2, thread 1</th>
<th>node 2, thread 2</th>
<th>node 2, thread 0</th>
<th>node 0, thread 1</th>
<th>node 0, thread 0</th>
<th>node 0, thread 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.609</td>
<td>9.547</td>
<td>9.54</td>
<td>node 1, thread 2</td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 3, thread 0</td>
<td>node 3, thread 2</td>
<td>node 3, thread 1</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 0</td>
<td>node 0, thread 2</td>
</tr>
<tr>
<td>9.118</td>
<td>9.118</td>
<td>9.104</td>
<td>node 1, thread 2</td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 3, thread 0</td>
<td>node 3, thread 2</td>
<td>node 3, thread 1</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 0</td>
<td>node 0, thread 2</td>
</tr>
<tr>
<td>9.057</td>
<td>9.037</td>
<td>9.025</td>
<td>node 1, thread 2</td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 3, thread 0</td>
<td>node 3, thread 2</td>
<td>node 3, thread 1</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 0</td>
<td>node 0, thread 2</td>
</tr>
<tr>
<td>9.019</td>
<td>8.995</td>
<td>8.977</td>
<td>node 1, thread 2</td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 3, thread 0</td>
<td>node 3, thread 2</td>
<td>node 3, thread 1</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 0</td>
<td>node 0, thread 2</td>
</tr>
<tr>
<td>8.636</td>
<td>7.477</td>
<td>6.911</td>
<td>node 1, thread 2</td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 3, thread 0</td>
<td>node 3, thread 2</td>
<td>node 3, thread 1</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 0</td>
<td>node 0, thread 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>node 1, thread 0</td>
<td>node 1, thread 1</td>
<td>node 1, thread 2</td>
<td>node 1, thread 3</td>
<td>node 2, thread 0</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 3</td>
<td>node 3, thread 0</td>
<td>node 3, thread 1</td>
<td>node 3, thread 2</td>
<td>node 3, thread 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>node 2, thread 0</td>
<td>node 2, thread 1</td>
<td>node 2, thread 2</td>
<td>node 2, thread 3</td>
<td>node 3, thread 0</td>
<td>node 3, thread 1</td>
<td>node 3, thread 2</td>
<td>node 3, thread 3</td>
<td>node 0, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 2</td>
<td>node 0, thread 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>node 3, thread 0</td>
<td>node 3, thread 1</td>
<td>node 3, thread 2</td>
<td>node 3, thread 3</td>
<td>node 0, thread 0</td>
<td>node 0, thread 1</td>
<td>node 0, thread 2</td>
<td>node 0, thread 3</td>
<td>std. dev.</td>
<td>std. dev.</td>
<td>std. dev.</td>
<td>std. dev.</td>
</tr>
</tbody>
</table>

Metric: Time
Value: Exclusive

Std. Dev.
Mean
node 0, thread 0
node 0, thread 1
node 0, thread 2
node 0, thread 3
node 1, thread 0
node 1, thread 1
node 1, thread 2
node 1, thread 3
node 2, thread 0
node 2, thread 1
node 2, thread 2
node 2, thread 3
node 3, thread 0
node 3, thread 1
node 3, thread 2
node 3, thread 3

TAU: ParaProf: Function Data Window: epik_bt-mz_B_4x4_trace

Name: main => MAIN_ => adi_ => z_solve_ => !$omp parallel @z_solve.f:43 => !$omp do @z_solve.f:52
Metric Name: Time
Value: Exclusive
Units: seconds

9.609
9.547
9.54
9.118
9.118
9.104
9.057
9.037
9.025
9.019
8.995
8.977
8.636
7.477
6.911
6.851
6.788
0.971

Exclusive Time: 9.118 seconds
Inclusive Time: 9.118 seconds
Calls: 3216.0
SubCalls: 0.0
TAU PerfExplorer data mining
Interactive event trace analysis

- Alternative & supplement to automatic trace analysis
- Visual presentation of dynamic runtime behaviour
  - event timeline chart for states & interactions of processes/threads
  - communication statistics, summaries & more
- Interactive browsing, zooming, selecting
  - linked displays & statistics adapt to selected time interval (zoom)
  - scalable server runs in parallel to handle larger traces

Developed by TU Dresden ZIH

- Open-source VampirTrace library bundled with OpenMPI 1.3
- http://www.tu-dresden.de/zih/vampirtrace/
- Vampir Server & GUI have a commercial license
- http://www.vampir.eu/
Vampir interactive trace analysis GUI
• Interactive event trace analysis
  ▪ Visual presentation of dynamic runtime behaviour
    ▪ event timeline chart for states & interactions of processes
    ▪ Interactive browsing, zooming, selecting
  ▪ Large variety of highly configurable analyses & displays
• Developed by Barcelona Supercomputing Center
  ▪ Paraver trace analyser and Extrae measurement library
  ▪ Open source available from http://www.bsc.es/paraver/
MAQAO

- Modular Assembler Quality Analyzer & Optimizer
  - Framework for binary manipulation
    - using plugins and scripting language
  - Tool exploiting framework to produce reports
    - fast prototyping and batch interface
  - STAN static performance model
  - MIL instrumentation language for dynamic analysis
    - building custom performance evaluation tools using HWCs
    - instrumentation of functions, loops, blocks & instructions

- Developed by UVSQ Exascale Computing Research lab
  - Supports Intel x86_64 microarchitecture
  - Available from www.maqao.org
Key tool components also provided as open-source

- Program development environment
  ▶ Eclipse PTP ETFw, UNITE

- Program/library instrumentation
  ▶ COBI, OPARI, PDToolkit

- Runtime measurement systems
  ▶ $P^n$MPI, Score-P, UniMCI

- Scalable I/O
  ▶ SIONlib

- Libraries & tools for handling (and converting) traces
  ▶ EPILOG, OTF, PEARL

- Analysis algebra & hierarchical/topological presentation
  ▶ CUBE
Scalable performance measurement infrastructure

- Supports instrumentation, profiling & trace collection, as well as online analysis of HPC parallel applications
- Works with Periscope, Scalasca, TAU & Vampir prototypes
- Based on updated tool components
  - CUBE4 profile data utilities & GUI
  - OA online access interface to performance measurements
  - OPARI2 OpenMP & pragma instrumenter
  - OTF2 open trace format

Created by German BMBF SILC & US DOE PRIMA projects

- JSC, RWTH, TUD, TUM, GNS, GRS, GWT & UO PRL
- Available as BSD open-source from http://www.score-p.org/
Portable native parallel I/O library & utilities

- Scalable massively-parallel I/O to task-local files
- Manages single or multiple physical files on disk
  - optimizes bandwidth available from I/O servers by matching block sizes/alignment, reduces metadata-server contention
- POSIX-I/O-compatible sequential & parallel API
  - adoption requires minimal source-code changes
- Tuned for common parallel filesystems
  - GPFS (BlueGene), Lustre (Cray), ...
- Convenient for application I/O, checkpointing,
  - Used by Scalasca tracing (when configured)

Developed by JSC

- Available as open-source from
  http://www.fz-juelich.de/jsc/sionlib/
Uniform integrated tool environment

- Manages installation & access to program development tools
  - based on software environment management “modules”
  - commonly used on most cluster and HPC systems
  - configurable for multiple MPI libraries & compiler suites
- Specifies how & where tools packages get installed
  - including integrating tools where possible
- Defines standard module names and different versions
- Supplies pre-defined module files
- Configurable to co-exist with local installations & policies

Developed by JSC, RWTH & TUD

- Available as open-source from http://www.vi-hps.org/projects/unite/