Scalasca:
A Scalable Portable Integrated Performance Measurement and Analysis Toolset

CEA Tools 2012 | Bernd Mohr
Exascale Performance Challenges

• Exascale systems will consist of
  ▪ Complex configurations
  ▪ With a huge number of components
    ▪ Very likely heterogeneous

• Deep software hierarchies of large, complex software components will be required to make use of such systems

⇒ Sophisticated integrated performance measurement, analysis, and optimization capabilities will be required to efficiently operate an Exascale system
Cross-Cutting Considerations

- Performance-aware design, development and deployment of hardware and software necessary

- Integration with OS, compilers, middleware and runtime systems required

- Support for performance observability in HW and SW (runtime) needed

- Enable performance measurement and optimization in case of HW and SW changes due to faults or power adaptation
Technical Challenges

- Heterogeneity
- Extreme Concurrency
- Perturbation and data volume
- Drawing insight from measurements
- Quality information sources

This requires tools to be

- Portable
- Insightful
- Scalable
- Integrated
Run everywhere

PORTABILITY
Portability: Supported Platforms

• **Instrumentation and measurement only** (visual analysis on front-end or workstation)
  - Cray XT3/XT4/XT5, XE6, XK6
  - IBM BlueGene/L, BlueGene/P, BlueGene/Q
  - NEC SX8 and SX9
  - K Machine?

• **Full support** (instrumentation, measurement, and automatic analysis)
  - Linux IA32, IA64, x86_64, and PPC based clusters
  - IBM AIX Power3/4/5/6/7 based clusters
  - SGI Linux IA64 and x86_64 based clusters
  - SUN/Oracle Solaris Sparc and x86/x86_64 based clusters
### Known Installations of Scalasca

**Companies**
- Bull (France)
- Dassault Aviation (France)
- EDF (France)
- GNS (Germany)
- MAGMA (Germany)
- RECOM (Germany)
- Shell (Netherlands)
- Sun Microsystems (USA)
- Qontix (UK)

**Research / HPC Centres**
- ANL (USA)
- BSC (Spain)
- CEA (France)
- CERFACS (France)
- CINECA (Italy)
- CSC (Finland)
- CSCS (Switzerland)

**Research / HPC Centres (cont.)**
- DLR (Germany)
- DKRZ (Germany)
- EPCC (UK)
- HLRN (Germany)
- HLRS (Germany)
- ICHEC (Ireland)
- IDRIS (France)
- JSCC (Russia)
- LLNL (USA)
- LRZ (Germany)
- MSU (Russia)
- NCAR (USA)
- NCSA (USA)
- NSCC (China)
- ORNL (USA)
- PSC (USA)
- RZG (Germany)

**Research / HPC Centres (cont.)**
- SARA (Netherlands)
- SAITC (Bulgaria)
- TACC (USA)

**Universities**
- RPI (USA)
- RWTH (Germany)
- TUD (Germany)
- UOregen (USA)
- UTK (USA)

**DoD Computing Centers (USA)**
- AFRL DSRC
- ARL DSRC
- ARSC DSRC
- ERDC DSRC
- Navy DSRC
- MHPCC DSRC
- SSC-Pacific
More than numbers and diagrams

INSIGHTFULNESS
“A picture is worth 1000 words…”

- MPI ring program
- “Real world” example
“What about 1000’s of pictures?”  
(with 100’s of menu options)
Example Automatic Analysis: Late Sender
Scalasca: Example MPI Patterns

(a) Late Sender
(b) Late Receiver
(c) Late Sender / Wrong Order
(d) Wait at N x N

Legend:
- ENTER
- EXIT
- SEND
- RECV
- COLLEXIT
The Scalasca Project

- **Scalable Analysis of Large Scale Applications**
- **Approach**
  - **Instrument** C, C++, and Fortran parallel applications
    - Based on MPI, OpenMP, SHMEM, or hybrid
  - **Option 1:** scalable call-path profiling
  - **Option 2:** scalable event trace analysis
    - **Collect** event traces
    - **Search** trace for event patterns representing inefficiencies
    - **Categorize and rank** inefficiencies found
- **Supports** MPI 2.2 (P2P, collectives, RMA, IO) and OpenMP 3.0 (exception: nesting)

http://www.scalasca.org/
CEA Tools 2012

Measurement library
- Instr. target application
- HWC

Local event traces

Parallel wait-state search

Wait-state report

Summary report

Report manipulation

Optimized measurement configuration

Instrumenter
- Compiler / linker
- Source modules

Instrumented executable

Which problem?

Where in the program?

Which process?
Scalasca Example: CESM Sea Ice Module Late Sender Analysis
Scalasca Example: CESM Sea Ice Module
Late Sender Analysis + Application Topology

Cube 3.0 QT: Experiments/epik_trace_filtercice_D_T62g16.4096.120304-162028.topo.cube.gz
File Display Topology Help

Absolute

Metric tree

- 0.00 Time
  - 1.83e5 Execution
    - 0.34 MPI
    - 3.21e4 Synchronization
    - 0.00 Communication
    - 1.18e5 Point-to-point
    - 1.74e5 Late Sender
    - 0.02 Late Receiver
    - 1.06e5 Collective
    - 0.00 Init/Exit
  - 0.00 Overhead
    - 2.75e10 Visits
    - 2.17e6 Synchronizations
    - 1.87e10 Communications
    - 1.91e13 Bytes transferred
    - 9.58e4 Computational imbalance
    - 2.82e5 Short-term delay costs
    - 2.76e5 Delay costs
    - 0.00 Wait states (direct vs. indirect)
      - 1.45e5 Direct wait time
      - 1.40e5 Indirect wait time
    - 2.85e5 Wait states (propagating vs. total)
    - 201.86 Critical path profile
    - 6.08e5 Performance impact

Call tree

- 0.00 ice_comp_mct::ice_run_mct (0.00%)
  - 482.96 ice_comp_mct::ice_import_mct (0.00%)
  - 1.42e4 ice_step_mod::step_therm2 (0.00%)
  - 0.00 ice_step_mod::step_dynamics (0.00%)
    - 0.00 ice_dyn_evt::evp (0.00%)
      - 2005.90 ice_boundary::ice_halo (0.00%)
      - 3536.37 ice_grid::t2grid_vector (0.00%)
      - 1792.13 ice_boundary::ice_halo (0.00%)
      - 0.00 ice_boundary::ice_haloup (0.00%)

- 1.13e5 MPI_Waitall
  - 1533.09 ice_grid::u2grid_vector (0.00%)
  - 1.86e4 ice_transport_driver::transport (0.00%)
  - 1.09e4 ice_state::bound_state (0.00%)
  - 8529.44 cice_runmod::coupling_prepare (0.00%)
  - 0.00 ice_history::ice_write_hist (0.00%)
  - 20.82 ice_history_write::icecdf (0.00%)
  - 0.00 ice_restart::dumpfile_pio (0.00%)
  - 5.05 ice_pio::ice_pio_initdecompress (0.00%)
  - 5.57 piodarray::write_darray_4d (0.00%)
  - 0.77 piodarray::write_darray_3d (0.00%)

0.00 1.74e5 (28.44%) 6.14e5
0.00 1.13e5 (64.73%) 1.74e5
0.00 0.00 27.57 +/- 18.26 63.73

Peer distribution

Topology 0 Topology 1 Virtual topology

[Map showing data distribution]
Scalasca Root Cause Analysis

• **Root-cause analysis**
  - Wait states typically caused by load or communication imbalances earlier in the program
  - Waiting time can also propagate (e.g., indirect waiting time)
  - Goal: Enhance performance analysis to find the root cause of wait states

• **Approach**
  - Distinguish between direct and indirect waiting time
  - Identify call path/process combinations delaying other processes and causing first order waiting time
  - Identify original delay

![Diagram showing direct and indirect waits]
Scalasca Example: CESM Sea Ice Module
Direct Wait Time Analysis
Scalasca Example: CESM Sea Ice Module
Indirect Wait Time Analysis
Scalasca Example: CESM Sea Ice Module Delay Costs Analysis
EXTREME CONCURRENCY
Scaling already important **TODAY!**

- **Number of Cores share for TOP 500 June 2012**

<table>
<thead>
<tr>
<th>NCore</th>
<th>Count</th>
<th>Share</th>
<th>$\Sigma$Rmax</th>
<th>Share</th>
<th>$\Sigma$NCore</th>
</tr>
</thead>
<tbody>
<tr>
<td>1025-2048</td>
<td>1</td>
<td>0.2%</td>
<td>122 TF</td>
<td>0.1%</td>
<td>1,280</td>
</tr>
<tr>
<td>2049-4096</td>
<td>9</td>
<td>1.8%</td>
<td>623 TF</td>
<td>0.5%</td>
<td>30,520</td>
</tr>
<tr>
<td>4097-8192</td>
<td>85</td>
<td>17.0%</td>
<td>7,500 TF</td>
<td>6.1%</td>
<td>581,728</td>
</tr>
<tr>
<td>8193-16384</td>
<td>268</td>
<td>53.6%</td>
<td>24,852 TF</td>
<td>20.1%</td>
<td>3,319,798</td>
</tr>
<tr>
<td>&gt; 16384</td>
<td>137</td>
<td>27.4%</td>
<td>90,376 TF</td>
<td>73.2%</td>
<td>9,519,131</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>500</td>
<td>100%</td>
<td>123,473 TF</td>
<td>100%</td>
<td>13,452,457</td>
</tr>
</tbody>
</table>

- **Average** system size: **26,904 cores**
- **Median** system size: **13,104 cores**
Roads to Scalability

• **Scalable data collection and reduction**
  - Parallel based on MPI and parallel I/O

• **Scalable parallel data analysis**
  - Parallelized
    - pattern search
    - delay analysis
    - critical-path analysis

• **Scalable visualizations**
  - Hierarchical browsers
  - Topology displays
  - 3D displays (via TAU paraprof)
Scalasca trace analysis sweep3D@294,912 BGP

- 10 min sweep3D runtime
- 11 sec replay
- 4 min trace data write/read (576 files)
- 7.6 TB buffered trace data
- 510 billion events

CUBE: > 3D ⇒ Folding (e.g. AB-CDE-CT)
CUBE: > 3D ⇒ Folding (e.g. AB-CDE-CT)
CUBE: > 3D ⇒ Folding (e.g. ABC-DECT)
CUBE: > 3D ⇒ Selecting
e.g. A,B,C,D1,E1,C6,T0
Scalasca trace analysis bt-mz@524,288 BGQ
INTEGRATION
Integration

• Need integrated tool (environment) for all levels of parallelization
  ▪ Inter-node (MPI)
  ▪ Intra-node (OpenMP, task-based programming)
  ▪ Accelerators (CUDA, OpenCL)

• Integration with performance modeling and prediction

• No tool fits all requirements
  ▪ Interoperability of tools
  ▪ Integration via open interfaces
Scalasca ↔ TAU ↔ VAMPIR ↔ Paraver
Status End 2011

Extrae

TAU - VT

Vampir Trace

TAU - TRACE

Scalasca

TAU - EPILOG

TAU - PROFILE

TAU trace

EPILOG trace

Trace Analyzer

CUBE3 profile

PRV trace

OTF / VTF3 trace

Paraver

VAMPIR

CUBE3 Presenter

PerfDMF

PARAPROF

gprof / mpiP profile
Scalasca ↔ TAU ↔ VAMPIR ↔ Paraver

Status End 2012 ?
Tool Integration: Score-P Objectives

• Mainly funded by SILC, PRIMA, LMAC projects

• Make common part of Periscope, Scalasca, TAU, and Vampir a community effort
  ▪ Score-P measurement system

• Functional requirements
  ▪ Performance data: profiles (CUBE4), traces (OTF2)
  ▪ Initially direct instrumentation, later also sampling
  ▪ Offline and online access
  ▪ Metrics: time, communication metrics and hardware counters
  ▪ Initially MPI 2 and OpenMP 3, later also CUDA and OpenCL

• Current release: V1.1rc2 of Sep 2012 (1.1 very soon)
  ▪ http://www.score-p.org
Score-P Architecture

- **Vampir**
- **Scalasca**
- **TAU**
- **Periscope**

**Score-P measurement infrastructure**

- Event traces (OTF2)
- Call-path profiles (CUBE4)
- Online interface
- Hardware counter (PAPI)
- Memory management
- etc ...

**Instrumentation**

- Compiler
- TAU instrumentor
- COBI (binary)
- MAQAO instrumentor
- OPARI 2 (OpenMP)
- MPI wrappers

**Application (MPI, OpenMP, hybrid)**
Score-P Partners

- Forschungszentrum Jülich, Germany
- German Research School for Simulation Sciences, Aachen, Germany
- Gesellschaft für numerische Simulation mbH Braunschweig, Germany
- RWTH Aachen, Germany
- Technische Universität Dresden, Germany
- Technische Universität München, Germany
- University of Oregon, Eugene, USA
Funded Integration Projects

- **SILC (01/2009 to 12/2011)**
  - Unified measurement system (Score-P) for Vampir, Scalasca, Periscope
- **PRIMA (08/2009 to 08/2012)**
  - Integration of TAU and Scalasca
- **LMAC (08/2011 to 07/2013)**
  - Evolution of Score-P
  - Analysis of performance dynamics
- **H4H (10/2010 to 09/2013)**
  - Hybrid programming for heterogeneous platforms
- **HOPSA (02/2011 to 01/2013)**
  - Integration of system and application monitoring
Scalasca ⇒ Vampir/Paraver integration
Scalasca ⇒ Vampir/Paraver integration
Scalasca ⇒ Vampir/Paraver integration
JSC: UNITE Tools Package 1.1

- Download, build, and install many tools in ONE package
- http://apps.fz-juelich.de/unite
  - Support for more and newer packages
    - UNITE package installer and module package (⇒ 1.1)
    - OTF-1.6.5 (⇒ 1.11.1)
    - pdtoolkit-3.15 (⇒ 3.18)
    - cube-3.3 (⇒ 3.4.2)
    - extraee-2.2.1
    - paraver-4.3.4
  - Scalasca-1.3.1 (⇒ 1.4.2)
  - Vampirtrace-5.8.2 (⇒ 5.13)
  - UniMCI-1.0.1
  - Marmot-2.4
  - Vampir-5.x, 7.x
  - VampirServer-1.x, 2.x, 7.x
  - tau-2.21.3
  - Score-P-1.0.2 (incl. OTF2 + CUBE4)
- Easier to update and site-adaptable module system
- More automatically enabled tool integrations
- Local testing on JSC platforms finished
- TODO: more extensive testing by HOPSA and H4H partners
- TODO: Update of documentation
Future Work

OPEN ISSUES
Open Issues

• **How to handle asynchronous non-deterministic executions?**
  - Currently favored programming model at node-level
  - Breaks measure-analyze-optimize cycle
  - Potential solution
    - Use traditional tools only at inter-node level
    - Use auto-tuning smart runtime systems inside node

• **Further factors to non-determinism**
  - Failing components and recovery actions
  - Components operating on varying speeds to save energy
Acknowledgements

• **Scalasca team (JSC)**

Markus Geimer  
Christian Heinrich  
Michael Knubloch  
Daniel Lorenz  
Bernd Mohr  
Peter Philippen  
Christian Rössel  
David Böhme  
Marc-André Hermanns  
Alexandru Calotoiu  
Marc Schlütter  
Pavel Saviankou  
Alexandre Strube  
Brian Wylie  
Anke Visser  
Ilja Zhukov

• **Sponsors**

[Images of sponsors logos]