Analysis report examination with Cube

David Böhme
Lawrence Livermore National Laboratory
Cube

- Parallel program analysis report exploration tools
  - Libraries for XML+binary report reading & writing
  - Algebra utilities for report processing
  - GUI for interactive analysis exploration
    - Requires Qt4 ≥4.6 or Qt 5

- Originally developed as part of the Scalasca toolset

- Now available as a separate component
  - Can be installed independently of Score-P, e.g., on laptop or desktop
  - Latest release: Cube 4.3.4 (April 2016)
Analysis presentation and exploration

- Representation of values (severity matrix) on three hierarchical axes
  - Performance property (metric)
  - Call path (program location)
  - System location (process/thread)

- Three coupled tree browsers

- Cube displays severities
  - As value: for precise comparison
  - As color: for easy identification of hotspots
  - Inclusive value when closed & exclusive value when expanded
  - Customizable via display modes
Analysis presentation

- What kind of performance metric?
- Where is it in the source code? In what context?
- How is it distributed across the processes/threads?
Inclusive vs. exclusive values

- Inclusive
  - Information of all sub-elements aggregated into single value
- Exclusive
  - Information cannot be subdivided further

```c
int foo()
{
    int a;
    a = 1 + 1;
    bar();
    a = a + 1;
    return a;
}
```
Score-P analysis report exploration (opening view)
Metric selection

Selecting the “Time” metric shows total execution time
Expanding the system tree

Distribution of selected metric for call path by process/thread
Expanding the call tree

Distribution of selected metric across the call tree

Collapsed: inclusive value
 Expanded: exclusive value
Selecting a call path

Selection updates metric values shown in columns to the right
Source-code view via context menu

Right-click opens context menu
Source-code view

```
subroutine bincrhs( lhs,c,r )
C-------------------------------------------------------------
C-------------------------------------------------------------
C-------------------------------------------------------------
c
C-------------------------------------------------------------

implicit none

double precision pivot, coeff, lhs
dimension lhs(5,5)
double precision c(5,5), r(5)
C-------------------------------------------------------------
c
C-------------------------------------------------------------

pivot = 1.00d0/lhs(1,1)
lhs(1,2) = lhs(1,2)*pivot
lhs(1,3) = lhs(1,3)*pivot
lhs(1,4) = lhs(1,4)*pivot
lhs(1,5) = lhs(1,5)*pivot
c(1,1) = c(1,1)*pivot
c(1,2) = c(1,2)*pivot
c(1,3) = c(1,3)*pivot
c(1,4) = c(1,4)*pivot
```

**Note:** This feature depends on file and line number information provided by the instrumentation, i.e., it may not always be available.
Select flat view tab, expand all nodes, and sort by exclusive value.
Box plot view

Box plot shows distribution across the system; with min/max/avg/median/quartiles
Alternative display modes

Data can be shown in various percentage modes
Important display modes

- Absolute
  - Absolute value shown in seconds/bytes/counts

- Selection percent
  - Value shown as percentage w.r.t. the selected node
    “on the left” (metric/call path)

- Peer percent (system tree only)
  - Value shown as percentage relative to the maximum peer value
Multiple selection

Select multiple nodes with Ctrl-click
Context-sensitive help

Context-sensitive help available for all GUI items
Derived metrics

- Derived metrics are defined using CubePL expressions, e.g.:
  \[ \text{metric::time(i)}/\text{metric::visits(e)} \]
- Values of derived metrics are not stored, but calculated on-the-fly
- Types of derived metrics:
  - Prederived: evaluation of the CubePL expression is performed before aggregation
  - Postderived: evaluation of the CubePL expression is performed after aggregation
- Examples:
  - “Average execution time”: Postderived metric with expression
    \[ \text{metric::time(i)}/\text{metric::visits(e)} \]
  - “Number of FLOP per second”: Postderived metric with expression
    \[ \text{metric::FLOP()}/\text{metric::time()} \]
Derived metrics in Cube GUI

Collection of derived metrics

Parameters of the derived metric

CubePL expression
Example: FLOPS based on PAPI_FP_OPS and time
CUBE algebra utilities

- Extracting solver sub-tree from analysis report

```bash
% cube_cut -r '<<ITERATION>>' scorep_bt-mz_BMic15p30x4_sum/profile.cubex
Writing cut.cubex... done.
```

- Calculating difference of two reports

```bash
% cube_diff scorep_bt-mz_BMic15p30x4_sum/profile.cubex cut.cubex
Writing diff.cubex... done.
```

- Additional utilities for merging, calculating mean, etc.
- Default output of cube_utility is a new report utility.cubex
- Further utilities for report scoring & statistics
- Run utility with `-h' (or no arguments) for brief usage info
Iteration profiling

- Show time dependent behavior by “unrolling” iterations

- Preparations:
  - Mark loop body by using Score-P instrumentation API in your source code

    ```
    SCOREP_USER_REGION_DEFINE( scorep_bt_loop )
    SCOREP_USER_REGION_BEGIN( scorep_bt_loop, "<<bt_iter>>", SCOREP_USER_REGION_TYPE_DYNAMIC )
    SCOREP_USER_REGION_END( scorep_bt_loop )
    ```

- Result in the Cube profile:
  - Iterations shown as separate call trees
  - Useful for checking results for specific iterations
    - or
  - Select your user-instrumented region and mark it as loop
  - Choose “Hide iterations”
    - View the Barplot statistics or the (thread x iterations) Heatmap
## Iteration profiling: Barplot

### Cube 4.3.0: scorep-bt=mz_C.8x4_sum/summary.cubex (on froggy2)

<table>
<thead>
<tr>
<th>Metric tree</th>
<th>Complete</th>
<th>Call tree</th>
<th>Flatview</th>
</tr>
</thead>
<tbody>
<tr>
<td>Absolute</td>
<td>Absolute</td>
<td>Absolute</td>
<td>Absolute</td>
</tr>
<tr>
<td>Time (sec)</td>
<td>0.00</td>
<td>1031.37</td>
<td>0.00</td>
</tr>
<tr>
<td>Execution</td>
<td>1031.37</td>
<td>0.00</td>
<td>Overhead</td>
</tr>
<tr>
<td>Idle threads</td>
<td>0.00</td>
<td>1031.37</td>
<td></td>
</tr>
<tr>
<td>Visits</td>
<td>11767</td>
<td>0.01</td>
<td>Bcast</td>
</tr>
<tr>
<td>Synchronizations</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Communications</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Bytes transferred</td>
<td>6.24e9</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>File operations</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Computational imbalance</td>
<td>62.35</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Minimum Inclusive Time</td>
<td>38.44</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Maximum Inclusive Time</td>
<td>1190.44</td>
<td>0.00</td>
<td></td>
</tr>
</tbody>
</table>

### Aggregation selection

- **Iterations**
- **Color** Automatic

### Operation
- All (Max/Avg/Min)
- Keep on Stack
- Clean Stack

---

**SORE-P AND SCALASCA PERFORMANCE TOOLS TRAINING, NERSC, JULY 26, 2016**
Iteration profiling: Heatmap
Cube: Further information

- Parallel program analysis report exploration tools
  - Libraries for XML report reading & writing
  - Algebra utilities for report processing
  - GUI for interactive analysis exploration
- Available under 3-clause BSD open-source license
- Documentation & sources:
  - http://www.scalasca.org
- User guide also part of installation:
  - `cube-config --cube-dir`/share/doc/CubeGuide.pdf
- Contact:
  - mailto: scalasca@fz-juelich.de