ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

The HPCToolkit

Introduction to Rosa Course
Scientific Computing Group

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

HPCToolkit: Motivation

- HPCToolkit's measurement tools collect performance data on each process and thread of an MPI program
  - Can be used with MPI or hybrid programs (MPI with OpenMP or pthreads)
- Supports C, C++ and Fortran
- The *hpcviewer* tool aids analysis of the application performance data

HPCToolkit: Motivation

- HPCToolkit can provide ***accurate fine-grain*** measurements of production applications running at scale, through
  - *low measurement overhead*
  - *call path profiling*
  - mapping between *fully optimised object code* and *associated source code*

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

HPCToolkit: Motivation

- Requirement: low measurement overhead
  - ***Avoid* instrumentation**
    - Tools such as *Tau* use source code instrumentation to insert profiling code into the source before compilation
      - Instrumentation can interfere with compiler optimisations such as inlining and loop transformations
      - Source instrumentation is not useful when using libraries
    - Binary instrumentors such as *gprof* may prevent full optimisation of code

HPCToolkit: Motivation

- Requirement: low measurement overhead
  - Instead of instrumentation use ***statistical sampling***
    - Sample events triggered by periodic interrupts from an interval timer or from the overflow of hardware performance counters
    - One can sample metrics that reflect work (e.g. floating point operations performed), resource consumption (e.g. cycles, memory bus transactions) or inefficiency (e.g. stalls)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

HPCToolkit: Motivation

- Call path profiling
  - Important to associate the costs incurred by a procedure with the context in which it is called
  - Particularly important for codes based on application frameworks and libraries

HPCToolkit: Motivation

- Recovering static program structure
  - Since measurements are made with reference to executables, it is necessary to map measurements back to program source code
  - HPCToolkit constructs this mapping using binary analysis
    - Can thus accurately attribute performance metrics to calling contexts, procedures, loops, and inlined instances of procedures

HPCToolkit: How to use on Rosa

- Load the hpct module
  - **module load hpct**
- Compile code with debugging support
  - *for PGI*: **-gopt**; *for PathScale*: **-g1**; *for GNU*: **-g**
    - Remember to add optimisation flags (eg. –O3) *after* –g to make sure compiler optimisation is not reduced
- Link your application using hpclink
  - **hpclink ftn –o code.exe code.o code1.o …**
  - **hpclink cc –o code.exe code.o code1.o …**

HPCToolkit: How to use on Rosa

- In your PBS batch job script set the CSPROF_OPT_EVENT environment variable according to which events you wish to monitor, e.g.
  - **export CSPROF_OPT_EVENT="PAPI_TOT_CYC@4000000 PAPI_L2_TCM@400000"**
    - This would sample your executable every 4 million cycles and every 400k level 2 cache misses
  - Note: use "papi_avail" program to see which hardware counters are available on the XT5

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

HPCToolkit: How to use on Rosa

- Launch code with **aprun** as normal
  - You will get a directory called hpctoolkit-<exe>-measurements-<pbs_jobid>
    - This will contain files with suffix .hpcrun (one for each MPI task) and logfiles (.log)
- Do a one-time binary analysis of your code using the hpcstruct command (interactively)
  - **hpcstruct <exe>**
    - This generates a file called <exe>.hpcstruct
    - May take some minutes!

HPCToolkit: How to use on Rosa

- Finally, invoke “hpcprof” to join the information from the run to the information from “hpcstruct”
  - **hpcprof –S <exe>.hpcstruct –I <path-to-source> hpctoolkit-<exe>-measurements-<pbs_jobid>/*.hpcrun**
    - This produces a directory hpctoolkit-<exe>-database, which can be shipped elsewhere if you wish to analyse the results on another machine

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

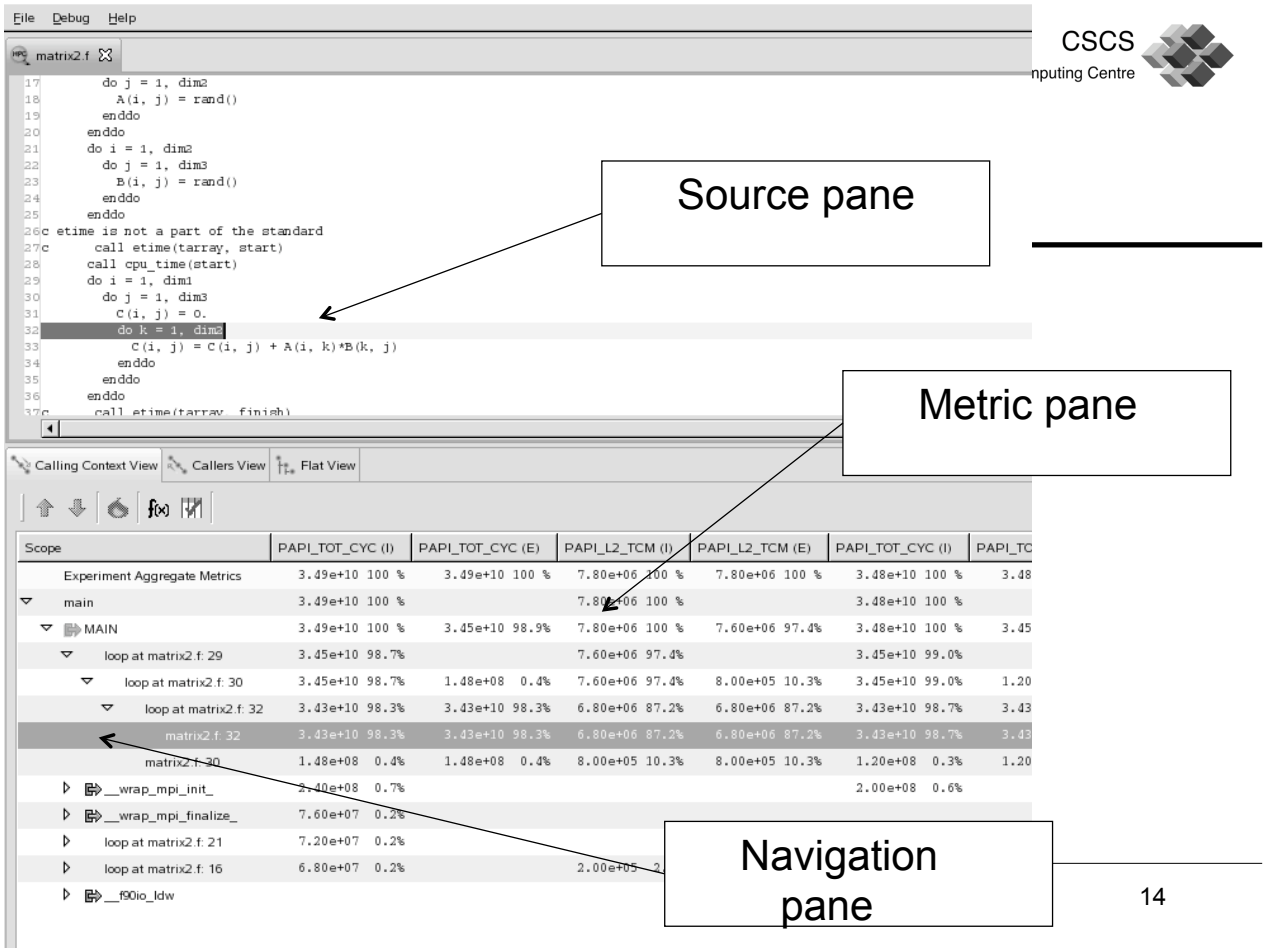
CSCS
Swiss National Supercomputing Centre

HPCToolkit: How to use on Rosa

- Java-based viewer “hpcviewer” provides a top-down visualisation of the experiment database
  - **module load hpct**
  - **hpcviewer hpctoolkit-<exe>-database**
- Viewer can also be installed and launched on your workstation
- Help can be found in a help pane
- Documentation on how to use the viewer at http://hpctoolkit.org/documentation.html

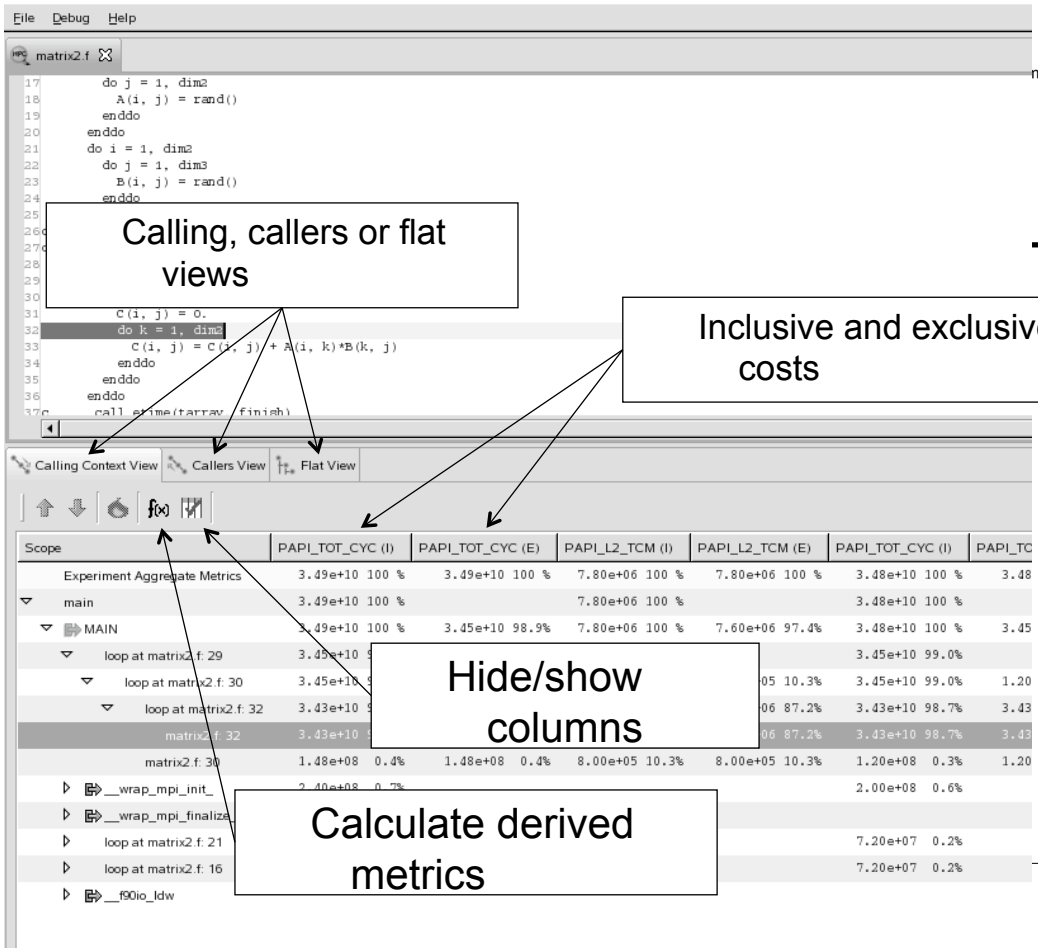
HPCToolkit: hpcviewer

- Browser window divided in three panes
  - Source pane
    - Select an entity in the navigation pane and the source pane will load appropriate file and highlight corresponding line
  - Navigation pane
    - Hierarchical tree-based structure used to organise performance data. Display depends on whether a context, callers, or flat view is being shown
  - Metric pane
    - Displays performance metrics



HPCToolkit: hpcviewer

- Supports three main views of performance data
  - Calling context view
    - Can see costs incurred by calls to a procedure in a particular calling context (call path)
  - Callers view
    - Bottom-up view enables you to look upward along call paths. Useful for understanding performance of procedures that are used in multiple contexts
  - Flat view
    - Costs incurred by a procedure in any calling context are aggregated together



HPCToolkit: How to analyse

- Values of individual metrics are of limited use by themselves
  - e.g. number of cache misses for a loop or routine is of little value in isolation – only useful when compared to the number of instructions executed or the number of cache accesses
  - hpcviewer can be used to calculate derived metrics via spreadsheet-like formulae (e.g. *cycles/instruction* from PAPI_TOT_CYC/PAPI_TOT_INS for a particular scope)

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

HPCToolkit: How to analyse

- Derived metrics can be used to pinpoint and quantify inefficiencies
- Scalability bottlenecks can be pinpointed by differential analysis of two profiles run using different numbers of processes
- See *Effective Strategies for Analysis Program Performance with HPCToolkit* for more details
- http://hpctoolkit.org