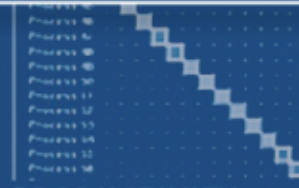




SOFTWARE

+  19.56 updatex  
+  399.70 updateien  
+  0.00 gene  
-  0.00 <<iteration loop>>  
+  447.52 genbc

PRODUCTIVITY



FAST SOLUTIONS

- PAPI\_L1\_ICM
- PAPI\_L2\_DCM
- PAPI\_L2\_ICM
- PAPI\_L1\_TCM

# scalasca

## Tutorial Exercise

### NPB-MPI/BT

Brian Wylie & Markus Geimer  
Jülich Supercomputing Centre  
[scalasca@fz-juelich.de](mailto:scalasca@fz-juelich.de)  
March 2010

0. Reference preparation for validation
  1. Program instrumentation: `skin`
  2. Summary measurement collection & analysis: `scan [-s]`
  3. Summary analysis report examination: `square`
  4. Summary experiment scoring: `square -s`
  5. Event trace collection & analysis: `scan -t`
  6. Event trace analysis report examination: `square`
- Configuration & customization
    - Instrumentation, Measurement, Analysis, Presentation

- Intermediate-level tutorial example
- Available in MPI, OpenMP, hybrid OpenMP/MPI variants
  - also MPI File I/O variants (collective & individual)
- Summary measurement collection & analysis
  - Automatic instrumentation
    - ▶ OpenMP, MPI & application functions
  - Summary analysis report examination
  - PAPI hardware counter metrics
- Trace measurement collection & analysis
  - Filter determination, specification & configuration
  - Automatic trace analysis report patterns
- Manual instrumentation
- Measurement configuration
- Analysis report algebra

- Load the module

```
% module load UNITE
UNITE loaded
% module load scalasca
scalasca/1.3 loaded
```

- ... and run `scalasca` for brief usage information

```
% scalasca
Scalasca 1.3
Toolset for scalable performance analysis of large-scale applications
usage: scalasca [-v][-n] {action}
  1. prepare application objects and executable for measurement:
     scalasca -instrument <compile-or-link-command>      # skin
  2. run application under control of measurement system:
     scalasca -analyze <application-launch-command>      # scan
  3. interactively explore measurement analysis report:
     scalasca -examine <experiment-archive|report>      # square

-v: enable verbose commentary
-n: show actions without taking them
-h: show quick reference guide (only)
```

- Prefix compile/link commands in Makefile definitions (config/make.def) with the Scalasca instrumenter

```
MPIF77 = scalasca -instrument mpif77
FLINK = $(MPIF77)
FFLAGS = -O

mpi-bt: $(OBJECTS)
        $(FLINK) $(FFLAGS) -o mpi-bt $(OBJECTS)

.f.o:
        $(MPIF77) $(FFLAGS) -c $<
```

- or use PREP macro as customizable preparation preposition

```
MPIF77 = $(PREP) mpif77
```

- By default, PREP macro is not set and no instrumentation is performed for a regular “production” build
- Specifying a PREP value in the Makefile or on the make command line uses it as a preposition, e.g., for instrumentation
  - ▶ % make **PREP=“scalasca -instrument”** ...  
**scalasca -instrument mpif77 -O -c bt.f**

- Return to root directory and clean-up

```
% make clean
```

- Re-build specifying Scalasca instrumenter as PREP

```
% make bt NPROCS=16 CLASS=W PREP="scalasca -instrument"
cd BT; make NPROCS=16 CLASS=W SUBTYPE= VERSION=
gmake: Entering directory 'BT'
cd ../sys; cc -o setparams setparams.c
../sys/setparams bt 16 W
scalasca -instrument mpif77 -c -o bt.f
...
scalasca -instrument mpif77 -c -o setup_mpi.f
cd ../common; scalasca -instrument mpif77 -c -o print_results.f
cd ../common; scalasca -instrument mpif77 -c -o timers.f
scalasca -instrument mpif77 -c -o btio.f
scalasca -instrument mpif77 -o -o ../bin.scalasca/bt_W.16 \
bt.o make_set.o initialize.o exact_solution.o exact_rhs.o \
set_constants.o adi.o define.o copy_faces.o rhs.o solve_subs.o \
x_solve.o y_solve.o z_solve.o add.o error.o verify.o setup_mpi.o \
../common/print_results.o ../common/timers.o btio.o
INFO: Instrumented executable for MPI measurement
gmake: Leaving directory 'BT'
```

- Run the application using the Scalasca measurement collection & analysis nexus prefixed to launch command

```
% cd bin.scalasca; scalasca -analyze mpiexec -np 16 ./bt_W.16
S=C=A=N: Scalasca 1.1 runtime summarization
S=C=A=N: ./epik_bt_W_16_sum experiment archive
S=C=A=N: Sun Mar 29 16:36:31 2009: Collect start
mpiexec -np 16 ./bt_W.16
[00000]EPIK: Created new measurement archive ./epik_bt_W_16_sum
[00000]EPIK: Activated ./epik_bt_W_16_sum [NO TRACE] (0.006s)

    [... Application output ...]

[00000]EPIK: Closing experiment ./epik_bt_W_16_sum
[00000]EPIK: 102 unique paths (102 max paths, 4 max frames, 0 unknown)
[00000]EPIK: Unifying... done (0.023s)
[00000]EPIK: Collating... done (0.049s)
[00000]EPIK: Closed experiment ./epik_bt_W_16_sum (0.073s)
S=C=A=N: Sun Mar 29 16:36:45 2009: Collect done (status=0) 14s
S=C=A=N: ./epik_bt_W_16_sum complete.
```

- Produces experiment directory ./epik\_bt\_W\_16\_sum

- Interactive exploration with Scalasca GUI

```
% scalasca -examine epik_bt_W_16_sum  
INFO: Post-processing runtime summarization result...  
INFO: Displaying ./epik_bt_W_16_sum/summary.cube...  
  
[GUI showing summary analysis report]
```

- The measurement archive directory ultimately contains
  - a copy of the execution output (epik.log)
  - a record of the measurement configuration (epik.conf)
  - the basic analysis report that was collated after measurement (epitome.cube)
  - the complete analysis report produced during post-processing (summary.cube.gz)



# Analysis report exploration (opening view)



The screenshot displays the Cube 3.0 QT software interface. The main window title is "Cube 3.0 QT: epik\_bt\_16\_sum/summary.cube.gz". The interface is divided into three main panels, each with a dropdown menu set to "Absolute".

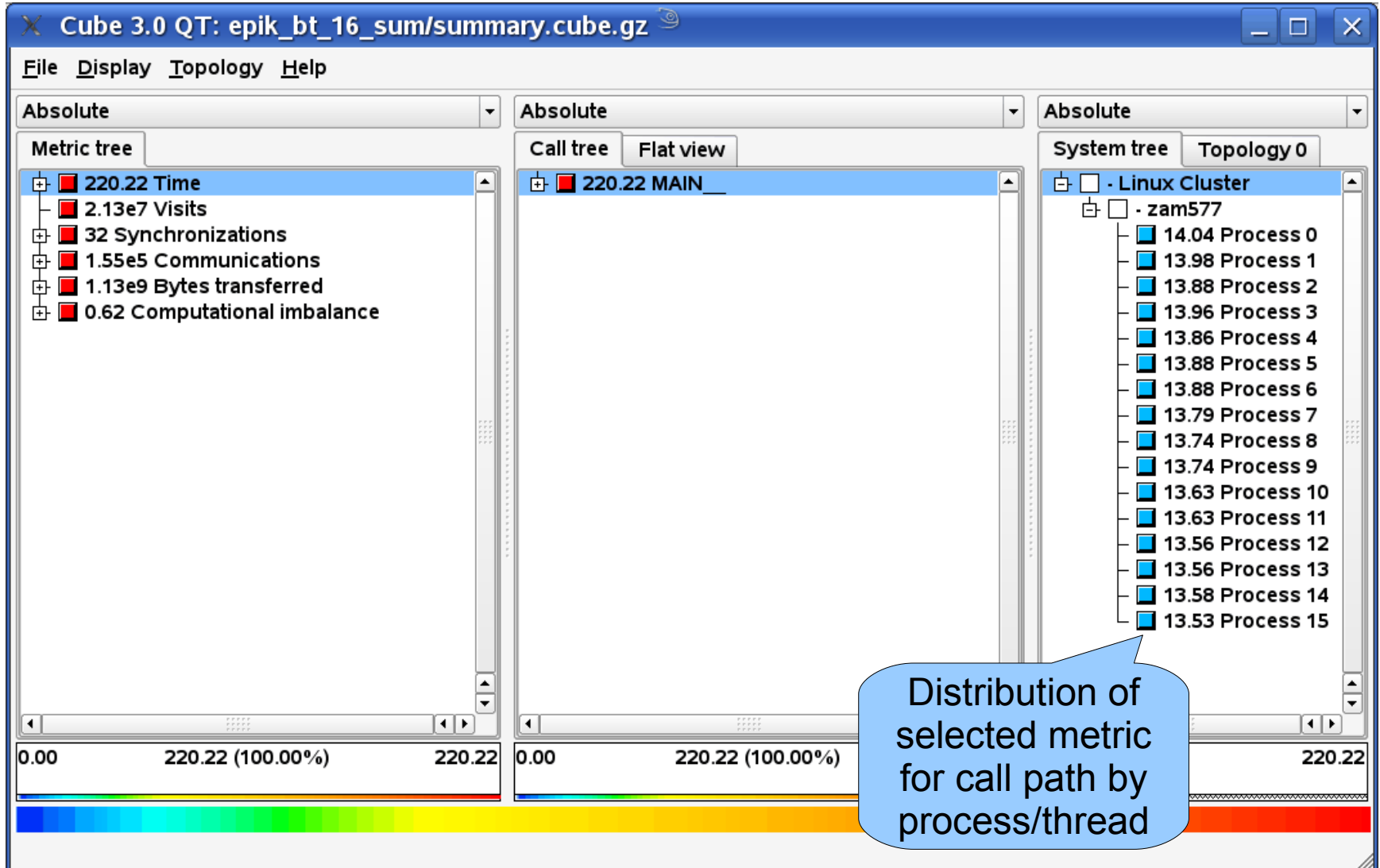
- Left Panel (Metric tree):** Shows a tree view of metrics for "220.22 Time". The metrics listed are:
  - 2.13e7 Visits
  - 32 Synchronizations
  - 1.55e5 Communications
  - 1.13e9 Bytes transferred
  - 0.62 Computational imbalance
- Middle Panel (Call tree / Flat view):** Shows a tree view for "220.22 MAIN\_".
- Right Panel (System tree / Topology 0):** Shows a tree view for "220.22 Linux Cluster".

An "About Cube 3.0 QT" dialog box is open in the center, containing the following information:

- This is Cube 3.0 QT.
- (c) 2009 Juelich Supercomputing Centre, Forschungszentrum Juelich GmbH
- Home page: [www.scalasca.org](http://www.scalasca.org)
- Technical support: [scalasca@fz-juelich.de](mailto:scalasca@fz-juelich.de)

At the bottom of the interface, there are three progress bars, each showing "0.00" on the left, "220.22 (100.00%)" in the middle, and "220.22" on the right. Below these bars is a horizontal color gradient bar ranging from blue to red.

# Analysis report exploration (system tree)



# Analysis report exploration (call tree)



Cube 3.0 QT: epik\_bt\_16\_sum/summary.cube.gz

File Display Topology Help

Absolute

Metric tree

- 220.22 Time
- 2.13e7 Visits
- 32 Synchronizations
- 1.55e5 Communications
- 1.13e9 Bytes transferred
- 0.62 Computational imbalance

Absolute

Call tree Flat view

- 0.01 MAIN\_
  - 4.16 setup\_mpi\_
    - 0.73 MPI\_Bcast
    - 0.00 make\_set\_
    - 0.00 set\_constants\_
  - 0.17 initialize\_
    - 0.00 setup\_btio\_
    - 0.00 lhsinit\_
  - 0.05 exact\_rhs\_
    - 0.00 compute\_buffer\_size\_
  - 0.03 adi\_
    - 17.40 copy\_faces\_
    - 69.62 x\_solve\_
    - 65.64 y\_solve\_
    - 0.10 z\_solve\_
      - 7.29 z\_solve\_cell\_
        - 0.07 z\_send\_solve\_info\_
        - 0.02 z\_receive\_solve\_info\_
        - 53.80 MPI\_Wait
        - 0.07 z\_unpack\_solve\_info\_
        - 0.35 z\_backsubstitute\_

Absolute

System tree Topology 0

- Linux Cluster
  - zam577
    - 3.42 Process 0
    - 3.61 Process 1
    - 3.21 Process 2
    - 3.20 Process 3
    - 3.34 Process 4
    - 3.68 Process 5
    - 3.17 Process 6
    - 3.75 Process 7
    - 3.37 Process 8
    - 3.19 Process 9
    - 3.29 Process 10
    - 3.15 Process 11
    - 3.35 Process 12
    - 3.45 Process 13
    - 3.43 Process 14
    - 3.19 Process 15

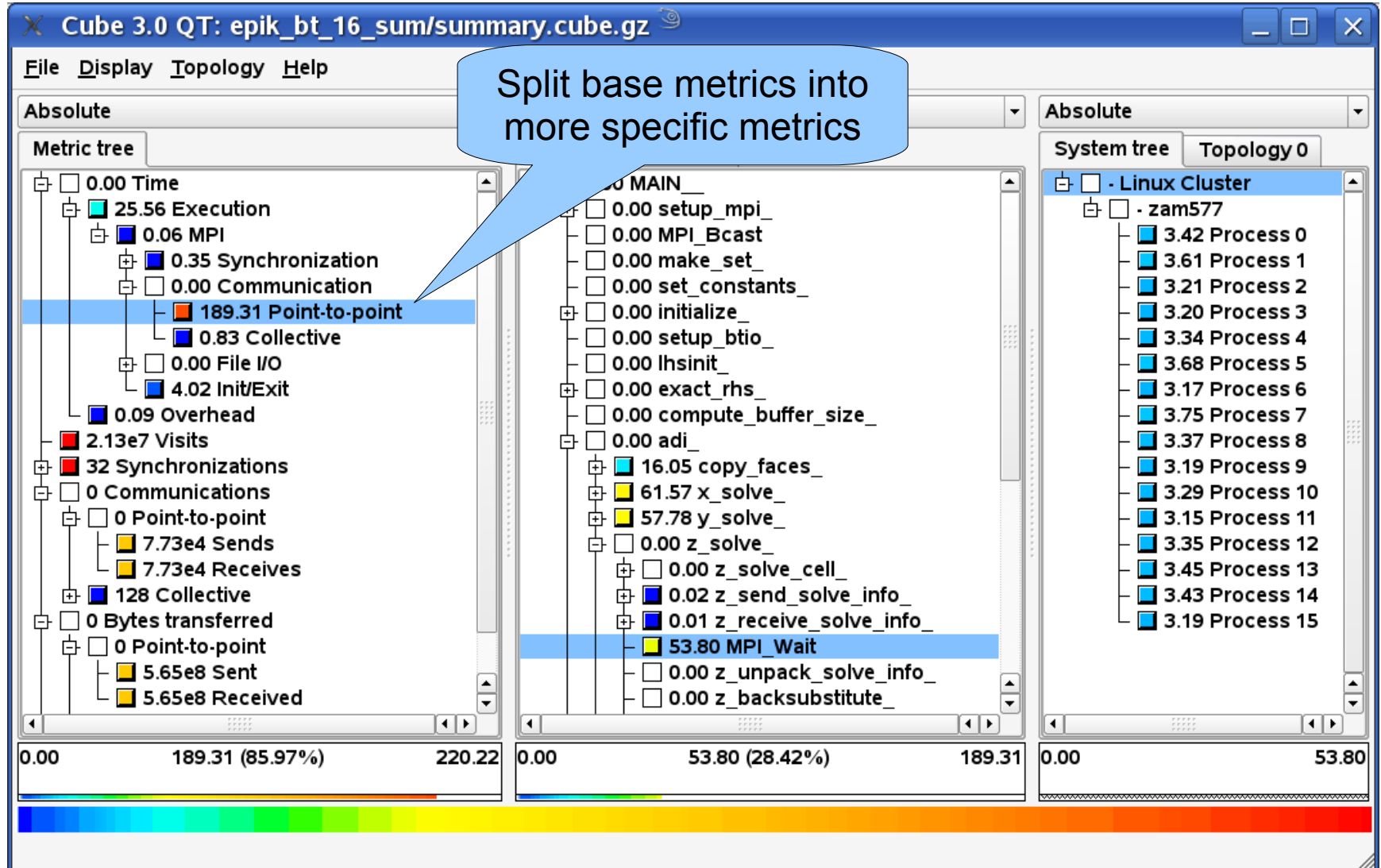
0.00 220.22 (100.00%) 220.22

0.00 53.80 (24.43%) 53.80

Selection updates metrics shown in columns to right

Distribution of selected metric across the call tree

# Analysis report exploration (metric tree)



Split base metrics into more specific metrics

# Analysis report exploration (source browser)



```

/home/wylie/test/NPB3.3/NPB3.3-MPI/BT/z_solve.f
c
c  in our terminology stage is the number of the cell in the y-dir
c  i.e. stage = 1 means the start of the line stage=ncells means e
c
c-----
c  do stage = 1,ncells
c    c = slice(3,stage)
c    isize = cell_size(1,c) - 1
c    jsize = cell_size(2,c) - 1
c    ksize = cell_size(3,c) - 1
c-----
c  set last-cell flag
c
c    if (stage .eq. ncells) then
c      last = 1
c    else
c      last = 0
c    endif
c
c    if (stage .eq. 1) then
c-----
c    This is the first cell, so solve without receiving data
c-----
c      first = 1
c      call lhsz(c)
c      call z_solve_cell(first,last,c)
c    else
c-----
c    Not the first cell of this line, so receive info from
c    processor working on preceeding cell
c-----
c      first = 0
c      call z_receive_solve_info(recv_id,c)
c-----
c    overlap computations and communications
c-----
c      call lhsz(c)
c-----
c    wait for completion
c-----
c    call mpi_wait(send_id,r_status,error)
c    call mpi_wait(recv_id,r_status,error)
c-----
c    install C'(kstart+1) and rhs'(kstart+1) to be used in this cell
c-----
c      call z_unpack_solve_info(c)
c      call z_solve_cell(first,last,c)
c    endif
c
c    if (last .eq. 0) call z_send_solve_info(send_id,c)
c  enddo

```

be.gz

ite Absolute

Flat view

System tree Topology 0

- Linux Cluster
  - zam577
    - 3.42 Process 0
    - 3.61 Process 1
    - 3.21 Process 2
    - 3.20 Process 3
    - 3.34 Process 4
    - 3.68 Process 5
    - 3.17 Process 6
    - 3.75 Process 7
    - 3.37 Process 8
    - 3.19 Process 9
    - 3.29 Process 10
    - 3.15 Process 11
    - 3.35 Process 12
    - 3.45 Process 13
    - 3.43 Process 14
    - 3.19 Process 15

0.00 MAIN\_  
0.00 setup\_mpi\_  
0.00 MPI\_Bcast  
0.00 make\_set\_  
0.00 set\_constants\_  
0.00 initialize\_  
0.00 setup\_btio\_  
0.00 lhsinit\_  
0.00 exact\_rhs\_  
0.00 compute\_buffer\_size\_  
0.00 adi\_  
16.05 copy\_faces\_  
61.57 x\_solve\_  
57.78 y\_solve\_  
0.00 z\_solve\_  
0.00 z\_solve\_cell\_  
0.02 z\_send\_solve\_info\_  
0.01 z\_receive\_solve\_info\_  
53.80 MPI Wait  
0.00 z\_unpack\_solve\_info\_  
0.00 z\_backsubstitute\_

53.80 (28.42%) 189.31 0.00 53.80

Read only Save Save as Font... Close

- If you made it this far, you successfully used Scalasca to
  - *instrument* the application
  - *analyze* its execution with a summary measurement, and
  - *examine* it with the interactive analysis report explorer GUI
- ... revealing the call-path profile annotated with
  - Time metrics (including MPI time, P2P & Collective times)
  - Visit counts
  - MPI message statistics (sends/receives, bytes sent/received)
  - Computational imbalance
- ... but how **good** was the measurement?
  - The measured execution produced the desired valid result
  - however, the execution took rather longer than expected!
    - ▶ even when ignoring measurement start-up/completion, therefore
    - ▶ it was probably dilated by instrumentation/measurement overhead

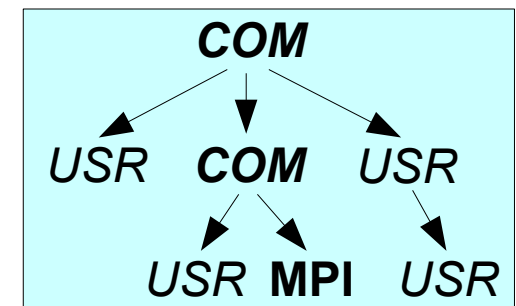
- Report scoring as textual output

```
% scalasca -examine -s epik_bt_W_16_sum
cube3_score ./epik_bt_W_16_sum/summary.cube
Reading ./epik_bt_W_16_sum/summary.cube... done.
Estimated aggregate size of event trace (total_tbc): 513,823,960 bytes
Estimated size of largest process trace (max_tbc): 32,528,680 bytes
(When tracing set ELG_BUFFER_SIZE to avoid intermediate flushes or
reduce requirements using filter file listing names of USR regions.)
```

flt	type	max_tbc	time	% region
	ANY	32528680	220.22	100.00 (summary) ALL
	MPI	642712	194.57	88.35 (summary) MPI
	COM	197928	1.03	0.47 (summary) COM
	USR	31688040	24.62	11.18 (summary) USR

- Region/callpath classification

- MPI (pure MPI library functions)
- OMP (pure OpenMP functions/regions)
- USR (user-level source local computation)
- COM (“combined” USR + OpenMP/MPI)
- ANY/ALL (aggregate of all region types)





- Summary measurement analysis score reveals
  - Total size of event trace would be over 500MB
  - Maximum trace buffer size would be over 30MB per process
    - ▶ smaller buffer would require flushes to disk during measurement resulting in substantial perturbation
  - 97% of the trace requirements are for USR regions
    - ▶ purely computational routines never found on COM call-paths common to communication routines
  - These USR regions contribute around 10% of total time
    - ▶ however, much of that is very likely to be measurement overhead for frequently-executed small routines
- Advisable to tune measurement configuration
  - Specify an adequate trace buffer size
  - Specify a filter file listing (USR) regions not to be measured

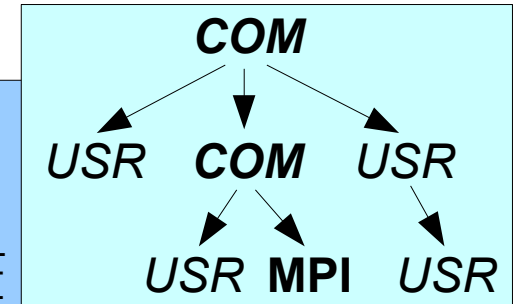


- Report scoring with region breakdown

```
% cube3_score -r ./epik_bt_w_16_sum/summary.cube
```

flt	type	max_tbc	time	% region	
	ANY	32528680	220.22	100.00	(summary) ALL
	MPI	642712	194.57	88.35	(summary) MPI
	COM	197928	1.03	0.47	(summary) COM
	USR	31688040	24.62	11.18	(summary) USR
	USR	10231704	4.44	2.02	binvcrhs_
	USR	10231704	3.06	1.39	matvec_sub_
	USR	10231704	3.53	1.60	matmul_sub_
	USR	492048	0.16	0.07	binvrhs_
	USR	360576	0.12	0.05	exact_solution_
	MPI	241500	0.27	0.12	MPI_Isend
	MPI	222180	0.12	0.06	MPI_Irecv
	MPI	173664	173.02	78.57	MPI_Wait
	USR	57888	0.06	0.03	lhsabinit_
	USR	19296	3.53	1.60	y_solve_cell_
	USR	19296	3.45	1.56	x_solve_cell_
	USR	19296	3.54	1.61	z_solve_cell_
	USR	19296	0.35	0.16	z_backsubstitute_

...



- Report scoring with prospective filter listing 6 USR regions

```
% cube3_score -r -f npb.filt ./epik_bt_w_16_sum/summary.cube
Applying filter "./npb.filt":
Estimated aggregate size of event trace (total_tbc): 16,852,888 bytes
Estimated size of largest process trace (max_tbc): 1,053,304 bytes
```

flt	type	max_tbc	time	% region
+	<b>FLT</b>	<b>31475376</b>	<b>11.37</b>	<b>5.16 (summary) FLT</b>
*	ANY	1053328	208.85	94.84 (summary) ALL-FLT
-	MPI	642712	194.57	88.35 (summary) MPI-FLT
*	COM	197928	1.03	0.47 (summary) COM-FLT
*	USR	212688	13.25	6.02 (summary) USR-FLT
+	<b>USR</b>	<b>10231704</b>	<b>4.44</b>	<b>2.02 binvcrhs_</b>
+	<b>USR</b>	<b>10231704</b>	<b>3.06</b>	<b>1.39 matvec_sub_</b>
+	<b>USR</b>	<b>10231704</b>	<b>3.53</b>	<b>1.60 matmul_sub_</b>
+	<b>USR</b>	<b>492048</b>	<b>0.16</b>	<b>0.07 binvrhs_</b>
+	<b>USR</b>	<b>360576</b>	<b>0.12</b>	<b>0.05 exact_solution_</b>
-	MPI	241500	0.27	0.12 MPI_Isend
-	MPI	222180	0.12	0.06 MPI_Irecv
-	MPI	173664	173.02	78.57 MPI_Wait
+	<b>USR</b>	<b>57888</b>	<b>0.06</b>	<b>0.03 lhsabinit_</b>
...				

```
% cat npb.filt
# filter for bt
binvcrhs_
matvec_sub_
matmul_sub_
binvrhs_
exact_solution_
lhsabinit_
```

Filtered routines marked with '+'

- Rename former measurement archive directory, set new filter configuration and re-run the measurement

```
% mv epik_bt_W_16_sum epik_bt_W_16_sum.nofilt
% export EPK_FILTER=npb.filt
% scalasca -analyze mpiexec -np 16 ./bt_W.16
S=C=A=N: Scalasca 1.1 runtime summarization
S=C=A=N: ./epik_bt_W_16_sum experiment archive
S=C=A=N: Sun Mar 29 16:58:34 2009: Collect start
mpiexec -np 16 ./bt_W.16
[00000]EPIK: Created new measurement archive ./epik_bt_W_16_sum
[00000]EPIK: EPK_FILTER "npb.filt" filtered 6 of 96 functions
[00000]EPIK: Activated ./epik_bt_W_16_sum [NO TRACE] (0.071s)

    [... Application output ...]

[00000]EPIK: Closing experiment ./epik_bt_W_16_sum
[00000]EPIK: 84 unique paths (84 max paths, 4 max frames, 0 unknowns)
[00000]EPIK: Unifying... done (0.014s)
[00000]EPIK: Collating... done (0.059s)
[00000]EPIK: Closed experiment ./epik_bt_W_16_sum (0.075s)
S=C=A=N: Sun Mar 29 16:58:41 2009: Collect done (status=0) 7s
S=C=A=N: ./epik_bt_W_16_sum complete.
```

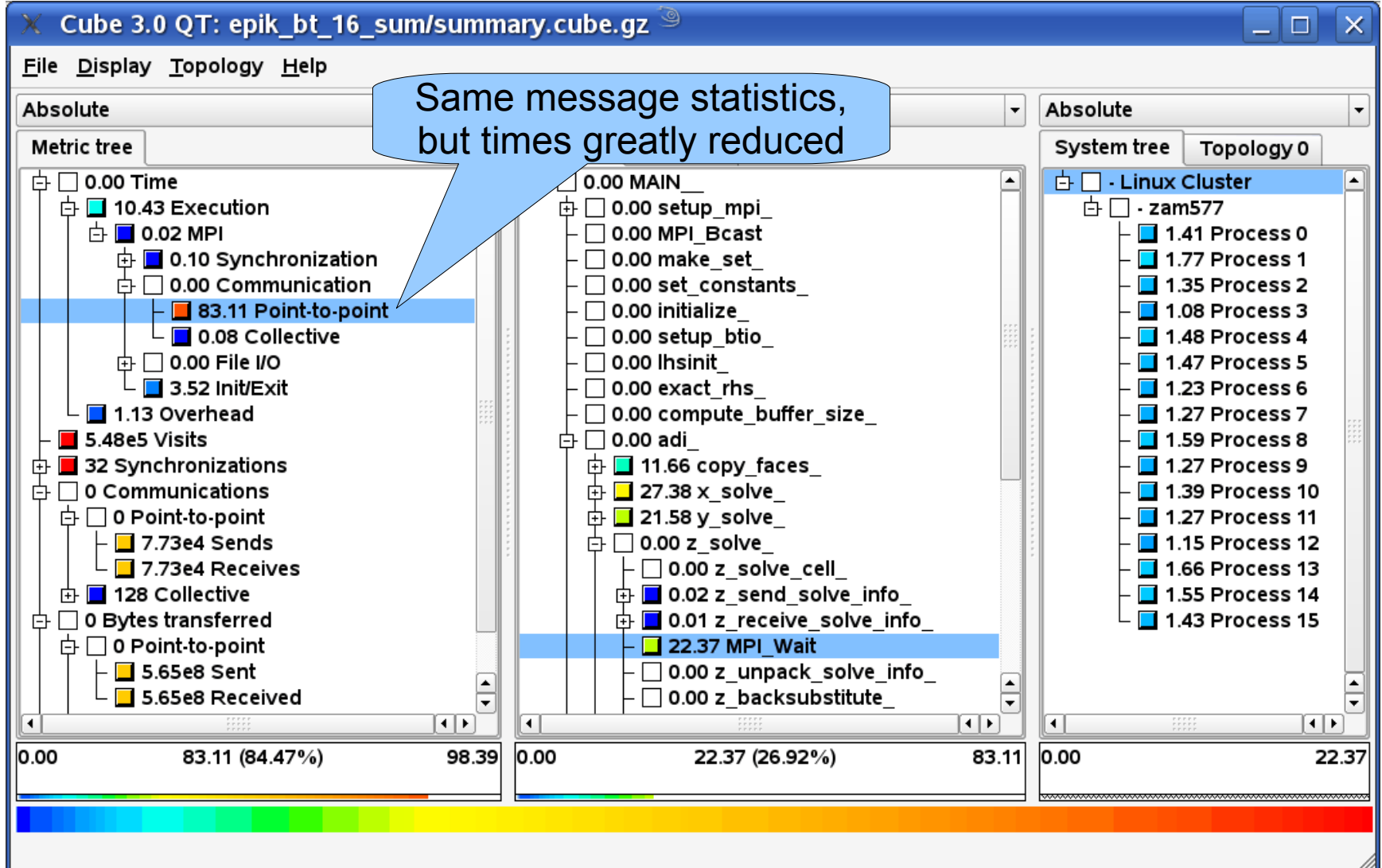
- Scoring of new analysis report as textual output

```
% scalasca -examine -s epik_bt_W_16_sum
INFO: Post-processing runtime summarization result...
cube3_score ./epik_bt_W_16_sum/summary.cube
Estimated aggregate size of event trace (total_tbc): 16,852,888 bytes
Estimated size of largest process trace (max_tbc): 1,053,328 bytes
(When tracing set ELG_BUFFER_SIZE to avoid intermediate flushes or
 reduce requirements using filter file listing names of USR regions.)

flt  type          max_tbc      time      % region
     ANY          1053328     98.39    100.00 (summary) ALL
     MPI           642712     86.83     88.25 (summary) MPI
     COM           197928      1.68      1.71 (summary) COM
     USR           212688      9.88     10.04 (summary) USR
```

- Significant reduction in runtime (measurement overhead)
  - Not only reduced time for USR regions, but MPI reduced too!
- Further measurement tuning (filtering) may be appropriate
  - e.g., use “timer\_\*” to filter timer\_start\_, timer\_read\_, etc.

# Summary analysis report exploration (filtered)



- Re-run the application using Scalasca nexus with “-t” flag

```
% scalasca -analyze -t mpiexec -np 16 ./bt_W.16
S=C=A=N: Scalasca trace collection and analysis
S=C=A=N: ./epik_bt_W_16_trace experiment archive
S=C=A=N: Sun Apr  5 18:50:57 2009: Collect start
mpiexec -np 16 ./bt_W.16
[00000]EPIK: Created new measurement archive ./epik_bt_W_16_trace
[00000]EPIK: EPK_FILTER "npb.filt" filtered 6 of 96 functions
[00000]EPIK: Activated ./epik_bt_W_16_trace [10000000 bytes] (0.051s)

    [... Application output ...]

[00000]EPIK: Closing experiment ./epik_bt_W_16_trace [0.016GB] (max 1053310)
[00000]EPIK: Flushed 1053330 bytes to file ./epik_bt_W_16_trace/ELG/00000
[00000]EPIK: 84 unique paths (84 max paths, 4 max frames, 0 unknowns)
[00000]EPIK: Unifying... done (0.021s)
[00013]EPIK: Flushed 1053306 bytes to file ./epik_bt_W_16_trace/ELG/00013
...
[00001]EPIK: Flushed 1053306 bytes to file ./epik_bt_W_16_trace/ELG/00001
[00000]EPIK: lflush=0.001GB@2.582MB/s, Pflush=0.015GB@35.458MB/s
[00000]EPIK: Closed experiment ./epik_bt_W_16_trace (0.178s)
S=C=A=N: Sun Apr  5 18:51:05 2009: Collect done (status=0) 8s
[... continued ...]
```

- Separate trace file per MPI rank written straight into new experiment directory ./epik\_bt\_W\_16\_trace

- Continues with automatic (parallel) analysis of trace files

```
S=C=A=N: Sun Apr  5 18:51:05 2009: Analyze start
mpiexec -np 16 scout.mpi ./epik_bt_W_16_trace
SCOUT  Copyright (c) 1998-2009 Forschungszentrum Juelich GmbH

Analyzing experiment archive ./epik_bt_W_16_trace

Reading definitions file ... done (0.563s).
Reading event trace files ... done (0.495s).
Preprocessing ... done (0.134s).
Analyzing event traces ... done (2.186s).
Writing CUBE report ... done (0.160s).

Total processing time      : 3.737s
Max. memory usage         : 7.000MB

S=C=A=N: Sun Apr  5 18:51:09 2009: Analyze done (status=0) 4s
S=C=A=N: ./epik_bt_W_16_trace complete.
```

- Produces trace analysis report in experiment directory

```
% scalasca -examine epik_bt_W_16_trace
INFO: Post-processing runtime summarization result...
INFO: Post-processing trace analysis report ...
INFO: Displaying ./epik_bt_W_16_sum/trace.cube...
```

[GUI showing trace analysis report]



**Cube 3.0 QT: epik\_bt\_16\_trace/trace.cube.gz**

File Display Topology Help

Absolute Absolute

Metric tree Call tree

- 0.00 Time
  - 9.60 Execution
    - 0.06 MPI
      - 0.00 Synchronization
        - 0.00 Collective
          - 0.07 Wait at Barrier
          - 0.02 Barrier Completion
        - 0.00 Communication
          - 64.98 Point-to-point
            - 18.50 Late Sender**
            - 0.00 Late Receiver
          - 0.16 Collective
            - 0.00 Early Reduce
            - 0.00 Early Scan
            - 0.00 Late Broadcast
            - 0.05 Wait at N x N
            - 0.01 N x N Completion
        - 0.00 File I/O
        - 3.89 Init/Exit
        - 0.83 Overhead
        - 5.48e5 Visits

0.00 18.50 (18.84%) 98 4.14 (22.38%) 18.50 0.00 4.14

Additional trace-based metrics in metric hierarchy

**Online description**

**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.

location

time

Send

Recv

- 0.10 Process 6
- 0.21 Process 7
- 0.23 Process 8
- 0.48 Process 9
- 0.22 Process 10
- 0.14 Process 11
- 0.59 Process 12
- 0.24 Process 13
- 0.31 Process 14
- 0.26 Process 15



- Consult quick reference guide for further information

```
% scalasca -h
Scalasca 1.3 – quick reference guide
pdfview /UNITE/packages/scalasca/1.3/doc/manuals/QuickReference.pdf

[PDF viewer showing quick reference guide]
```

- CUBE GUI provides context sensitive help and on-line metric descriptions (including problem diagnosis hints)
- EPIK archive directories contain analysis report(s), measurement collection & analysis logs, etc.
- Instrumentation, measurement, analysis & presentation can all be extensively customized
  - covered in more detailed presentation
- Visit [www.scalasca.org](http://www.scalasca.org) or mail [scalasca@fz-juelich.de](mailto:scalasca@fz-juelich.de)

0. Reference preparation for validation
  1. Program instrumentation: `skin`
  2. Summary measurement collection & analysis: `scan [-s]`
  3. Summary analysis report examination: `square`
  4. Summary experiment scoring: `square -s`
  5. Event trace collection & analysis: `scan -t`
  6. Event trace analysis report examination: `square`
- General usage/help: `scalasca [-h]`
  - Instrumentation, measurement, analysis & presentation can all be extensively customized
    - covered in more detailed presentation
  - Visit [www.scalasca.org](http://www.scalasca.org) or mail [scalasca@fz-juelich.de](mailto:scalasca@fz-juelich.de)

- Prepares application objects & executables for measurement
  - **skin = scalasca -instrument**
  - **skin [options] <compile-or-link-command>**
    - ▶ defaults to automatic instrumentation of USR routines by compiler
      - available for most compilers, but not all
      - when not desired, disable with -comp=none
    - ▶ for OpenMP, includes source-level pre-processing of directives
    - ▶ for MPI, links wrappers to PMPI library routines
  - [-pdt] pre-process sources with PDTToolkit (when available)
    - ▶ configurable instrumentation of specified routines (or all by default)
  - Manual instrumentation activation
    - ▶ offers complementary program structure information for analyses via user-provided annotations (e.g., phases, loops, ...)
    - ▶ [-user] enable EPIK user instrumentation API macros
    - ▶ [-pomp] enable processing of POMP region pragmas/directives

- Automatic source instrumentation using PDTToolkit [-pdt]
  - only available if configured when Scalasca installed
- By default, instruments all routines in source file
  - source routines are automatically instrumented by compiler, therefore use **-comp=none** to avoid duplicate instrumentation
- Selective instrumentation of specified routines
  - -optTauSelectFile=<filename>
  - TAU-format plain text specification file
    - ▶ list names of source files and routines to exclude from instrumentation, using wildcard patterns
      - unsupported TAU instrumentation features are silently ignored
    - ▶ refer to TAU/PDTToolkit documentation for details
    - ▶ refer to Scalasca User Guide for known limitations

- EPIK user instrumentation API
  - `#include "epik_user.h"`
  - `EPIK_USER_REG(epik_solve, "<<Solve>>")`
  - `EPIK_USER_START(epik_solve)`
  - `EPIK_USER_END(epik_solve)`
- Can be used to mark initialization, solver & other phases
  - Annotation macros ignored by default
  - Instrumentation enabled with "**-user**" flag to instrumenter
  - Also available for Fortran
    - ▶ `#include "epik_user.inc"` and use C preprocessor
- Appear as additional regions in analyses
  - Distinguishes performance of important phase from rest

- Runs application under control of measurement system to collect and analyze an execution experiment
  - **scan = scalasca -analyze**
  - **scan [options] <application-launch-command>**
    - ▶ e.g., scan [options] [\$MPIEXEC [mpiexec-options]] [target [args]]
  - **[-s]** collect summarization experiment [default]
  - **[-t]** collect event traces and then analyze them automatically
  - Additional options
    - ▶ **[-e title]** specify experiment archive (directory) name: *epik\_title*
    - ▶ **[-f filter]** specify file listing routines to ignore during measurement
    - ▶ **[-m metric1:metric2:...]** include hardware counter metrics
    - ▶ **[-n]** preview scan and perform checks but don't execute
    - ▶ **[-q]** quiesce (disable most) measurement collection
    - ▶ **[-a]** (re-)analyze a previously collected trace experiment

- Via `./EPIK.CONF` file

```
EPK_FILTER=smg2000.filt  
EPK_MPI_ENABLED=CG:COLL:ENV:IO:P2P:RMA:TOPO  
ELG_BUFFER_SIZE=40000000
```

- Via environment variables

```
% export EPK_FILTER=smg2000.filt  
% export EPK_MPI_ENABLED=CG:COLL:ENV:IO:P2P:RMA:TOPO  
% export ELG_BUFFER_SIZE=40000000
```

- Via command-line flags (partially)

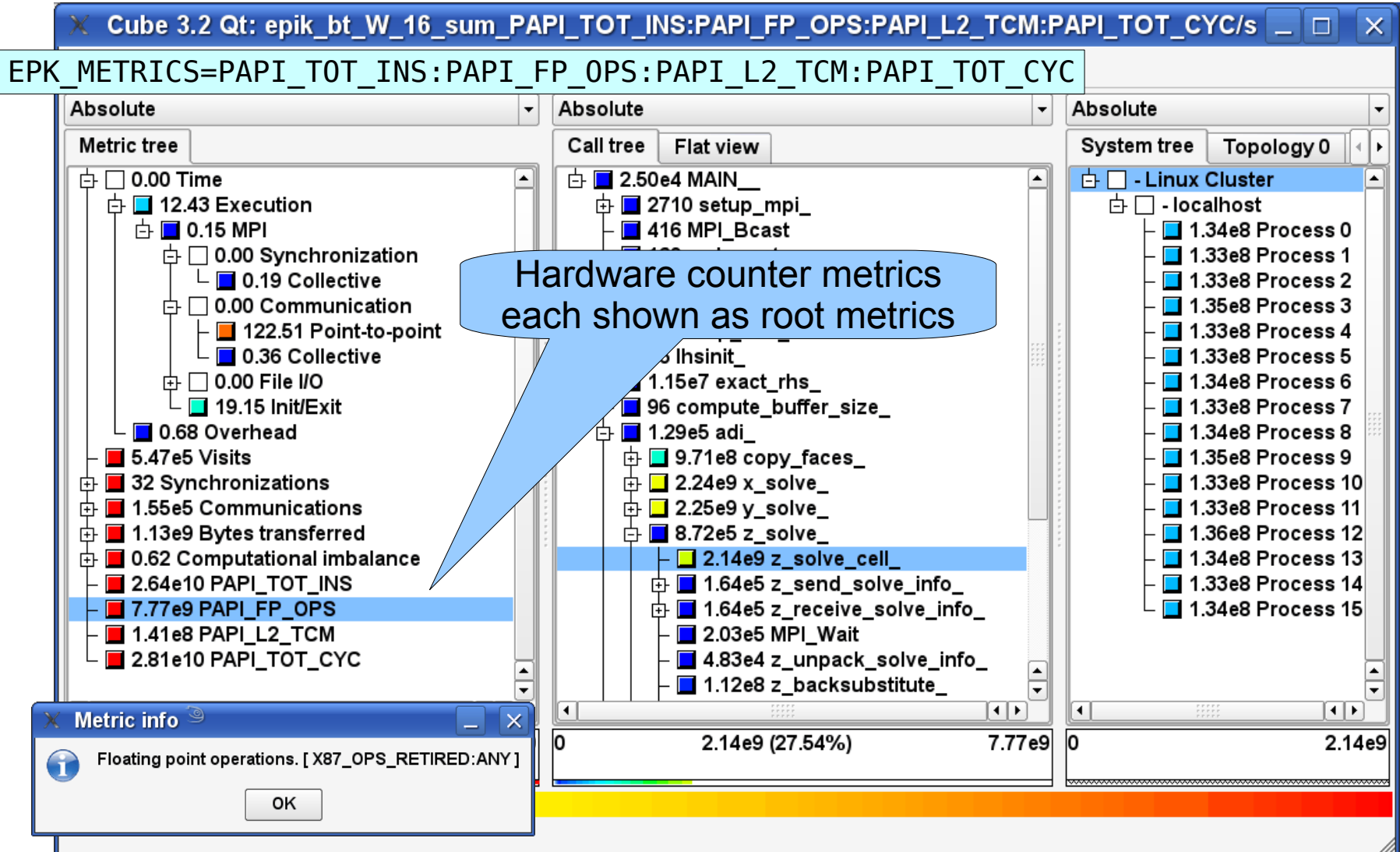
```
% scalasca -analyze -f smg2000.filt ...
```

- To show current/default configuration

```
% epik_conf
```

- Actual Scalasca measurement configuration saved in experiment archive as `epik.conf`

# Summary analysis report with HWC metrics





- Generally, the SCAN nexus will correctly parse execution command lines, but occasionally you may need to help it
- MPI launcher arguments may need to be explicitly separated from the target application with a double-dash

```
% scalasca -analyze mpirun -np 16 -- a.exe arg
```

- Unusual MPI launcher options may need to be quoted

```
% scalasca -analyze mpirun -np 16 "-verbose 2" a.exe arg
```

- (On most systems `-verbose` doesn't take an argument)

- Explicitly specify the instrumented target executable name when using imposter commands/scripts

```
% export SCAN_TARGET=a.exe  
% scalasca -analyze imposter.sh i.arg a.exe arg  
% scan -t mpirun -np 16 imposter.sh i.arg a.exe arg
```

- (*dplace*, *omplace* and *numactl* are common imposters)

- Prepares and presents measurement analysis report(s) for scoring and/or interactive exploration
  - **square = scalasca -examine**
  - **square [options] <experiment-archive|report>**
    - ▶ e.g., square epik\_*title*
  - Post-processes intermediate measurement analysis reports
  - Launches GUI and presents default analysis report (if multiple reports available)
    - ▶ trace analysis given precedence over summary analysis
    - ▶ select other reports via File/Open menu
  - [-s] skip display and output textual score report
    - ▶ estimate total trace size and maximum rank trace size
    - ▶ breakdown of USR vs. MPI/OMP vs. COM region requirements

- Extracting a sub-tree from an analysis report

```
% cube3_cut -r '<<SMG.Solve>>' epik_smg2000_12_trace/trace.cube  
Writing cut.cube... done.
```

- Calculating difference of two analysis reports

```
% cube3_diff epik_bt_9_trace/trace.cube epik_bt_16_trace/trace.cube  
Writing diff.cube... done.
```

- Combining two or more related analysis reports

```
% cube3_merge trace/trace.cube HWC1/summary.cube HWC2/summary.cube  
Writing merge.cube... done.
```

- Additional algebra utilities for calculating mean, etc.
  - Default output of `cube3_utility` is a new report `utility.cube`
- Further utilities for report scoring & statistics
- Run utility with “-h” (or no arguments) for brief usage info

# DON'T PANIC!

- Remember the Scaldasca User Guide is your friend
- And if you need more advice, <mailto:scaldasca@fz-juelich.de>