Short introduction to MPI: Distributed memory computing

M.C. Mihoubi
cherif.mihoubi@iwes.fraunhofer.de

Fraunhofer-Institut für Windenergie und Energiesystemtechnik IWES

November 22, 2012

Fraunhofer IWES

Outlines

The big picture

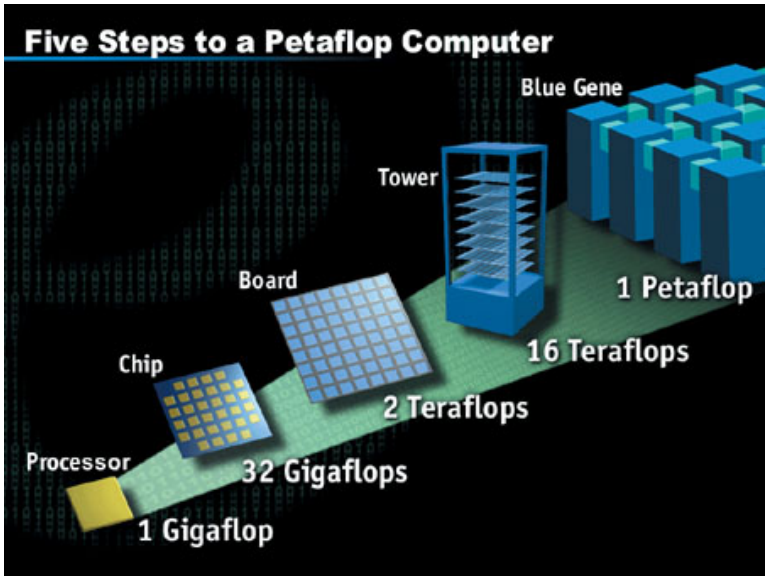


Figure: optimization flow chart

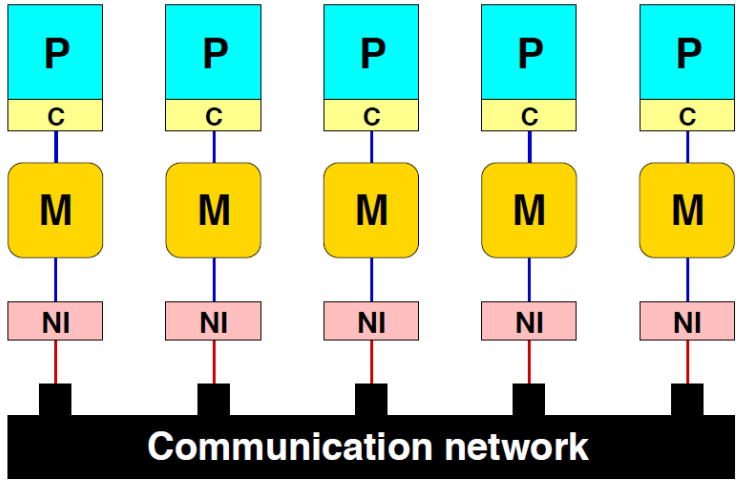


Figure: programming model of a distributed memory parallel computter

Practical Performance tuning

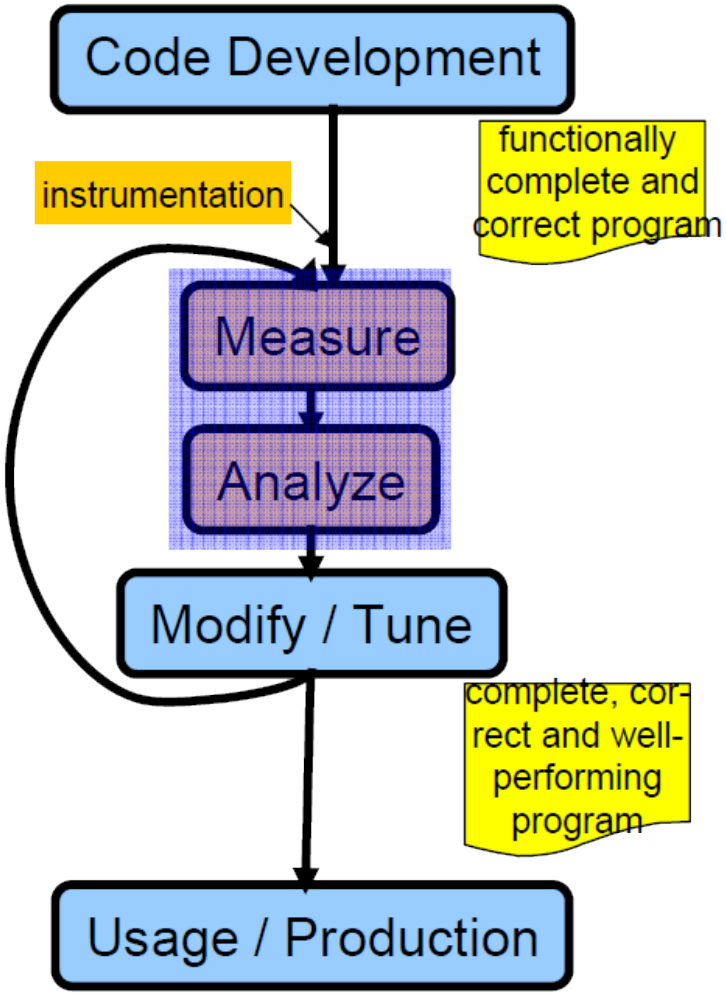
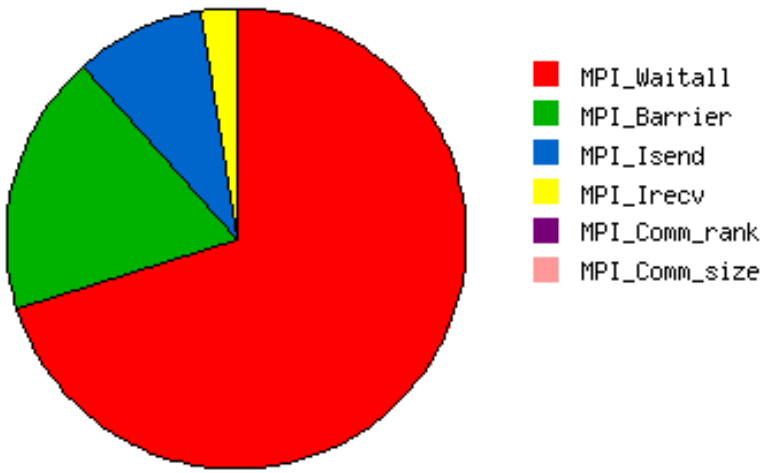


Figure: programming model of a distributed memory parallel computter

Why profiling ?

1. Determine performance problems
2. To validate tuning decisions optimizations (after each step!)
3. Optimizing MPI Performance (i.e. combining messages)

Overlapping Computation and Communication



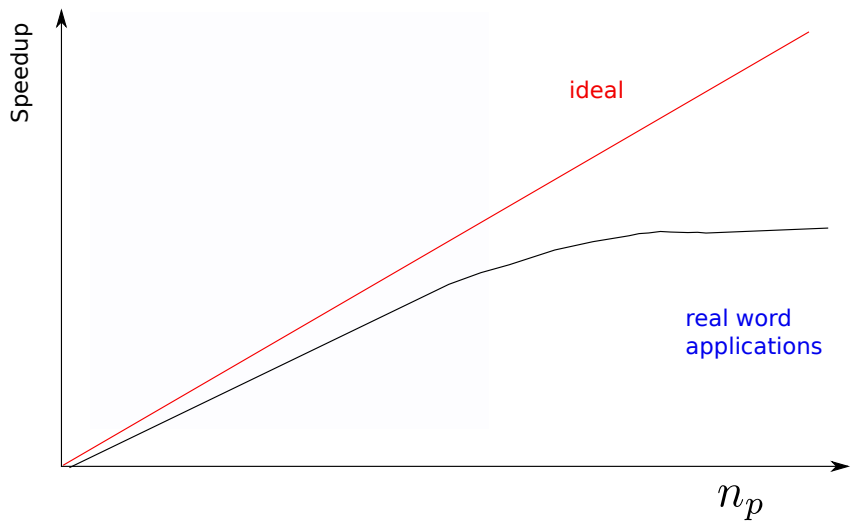
- Basic idea - make the time in MPI_Wait goto zero

```
MPI_ISend()
MPI_IRecv()
some code()
MPI_Wait()
```

- Inpracticeveryhardtoachieve

Performance

- Speed-up



- Efficiency

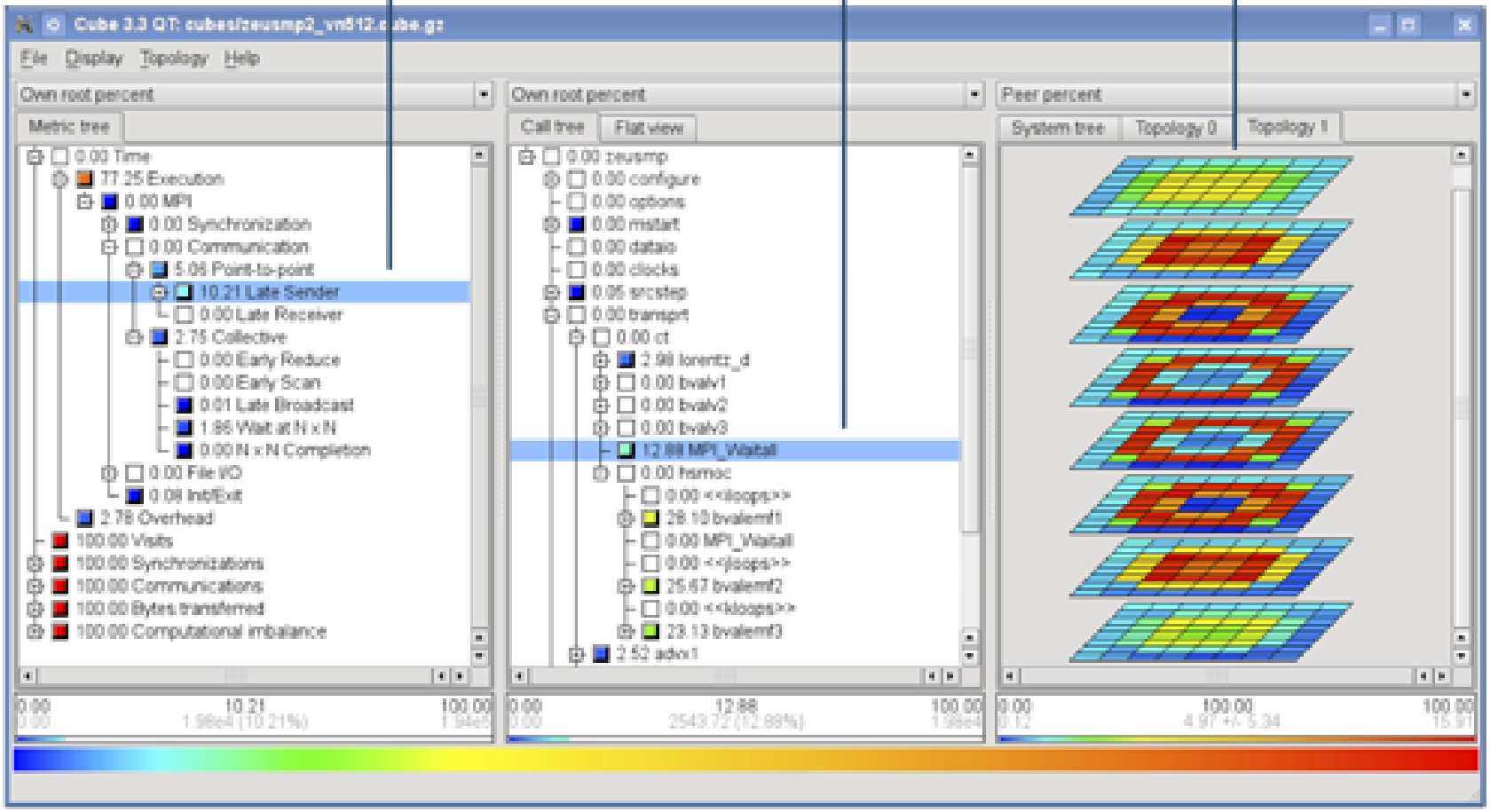
Scalasca

- Open source, New BSD license
- Portable
  - Cray XT, IBM BlueGene, IBM SP & blade clusters,
  - NEC SX, SGI Altix, SiCortex, Solaris & Linux clusters, ...
- Supports parallel programming paradigms & languages
  - MPI, OpenMP & hybrid OpenMP+MPI
  - Fortran, C, C++
- Integrated instrumentation, measurement & analysis toolset
  - Automatic and/or manual customizable instrumentation
  - Runtime summarization (aka profiling)
  - Automatic event trace analysis
  - Analysis report exploration & manipulation

Which performance problem?

Where in the program?

Where in the system?



Scalasca: three steps

- Instrumentation (or skin)

```
mpicc -c foo.c
mpicxx -o foo foo.cpp
mpif90 -openmp -o bar bar.f90
```

```
scalasca -instrument mpicc -c foo.c
scalasca -inst -pomp mpicxx -o foo foo.cpp
skin mpif90 -openmp -o bar bar.f90
```

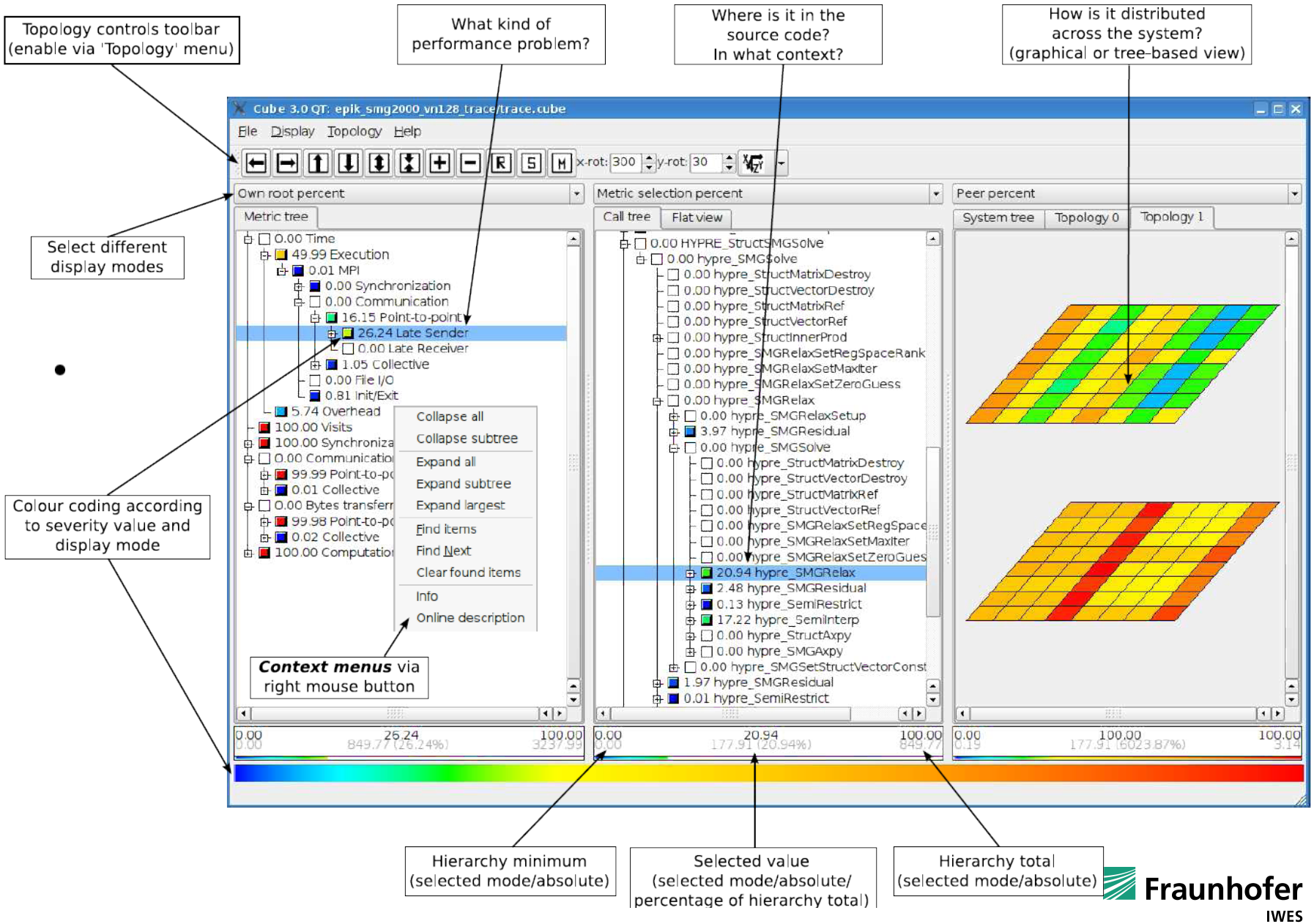
- Measurement & analyze (or scan)

```
mpiexec -np 4 foo args
OMP NUM THREADS=3 ./bar
mpiexec -np 4 foobar
```

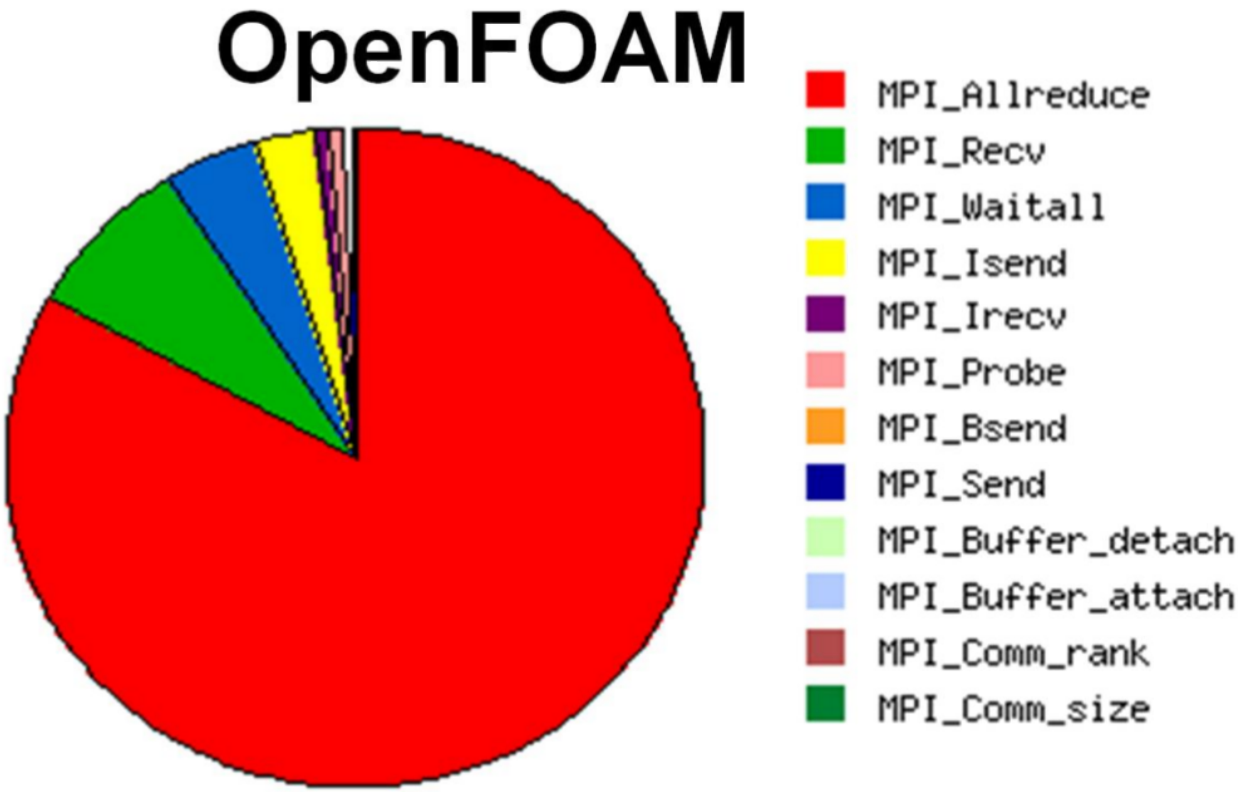
```
scalasca -analyze mpiexec -np 4 foo args
scalasca -analyze OMP NUM THREADS=3 ./bar
scan -s mpiexec -np 4 foobar
```

- Analysis report examination

```
epik foo 4 sum
epik bar 0x3 trace
epik foobar 4x3 sum
```


Topology controls toolbar (enable via 'Topology' menu)
What kind of performance problem?
Where is it in the source code? In what context?
How is it distributed across the system? (graphical or tree-based view)
Select different display modes
Colour coding according to severity value and display mode
Context menus via right mouse button
Hierarchy minimum (selected mode/absolute)
Selected value (selected mode/absolute/ percentage of hierarchy total)
Hierarchy total (selected mode/absolute)
Cube 3.0 QT: epik_smg2000_vn128_trace/trace.cube
File Display Topology Help
Own root percent
Metric selection percent
Peer percent
Metric tree
Call tree
Flat view
System tree
Topology 0
Topology 1
0.00 Time
49.99 Execution
0.01 MPI
0.00 Synchronization
0.00 Communication
16.15 Point-to-point
26.24 Late Sender
0.00 Late Receiver
1.05 Collective
0.00 File I/O
0.81 Init/Exit
5.74 Overhead
100.00 Visits
Collapse all
Collapse subtree
Expand all
Expand subtree
Expand largest
Find items
Find Next
Clear found items
Info
Online description
0.00 HYPRE_StructSMGSolve
0.00 hypre_SMGSolve
0.00 hypre_StructMatrixDestroy
0.00 hypre_StructVectorDestroy
0.00 hypre_StructMatrixRef
0.00 hypre_StructVectorRef
0.00 hypre_StructInnerProd
0.00 hypre_SMGRelaxSetRegSpaceRank
0.00 hypre_SMGRelaxSetMaxIter
0.00 hypre_SMGRelaxSetZeroGuess
0.00 hypre_SMGRelax
0.00 hypre_SMGRelaxSetup
3.97 hypre_SMGResidual
0.00 hypre_SMGSolve
0.00 hypre_StructMatrixDestroy
0.00 hypre_StructVectorDestroy
0.00 hypre_StructMatrixRef
0.00 hypre_StructVectorRef
20.94 hypre_SMGRelax
2.48 hypre_SMGResidual
0.13 hypre_SemiRestrict
17.22 hypre_SemiInterp
0.00 hypre_StructAxpy
0.00 hypre_SMGAxpy
0.00 hypre_SMGSetStructVectorConst
1.97 hypre_SMGResidual
0.01 hypre_SemiRestrict
0.00 26.24 100.00
849.77 (26.24%)
0.00 20.94 100.00
177.91 (20.94%)
0.00 100.00 100.00
177.91 (6023.87%)

OpenFOAM MPI profiling information



Thank you for your attention.

- for more documentation, please take look :
  www2.fz-juelich.de/jsc/datapool/scalasca/QuickReference.pdf