Performance Tools Hands-On

Feb/2017

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

tools@bsc.es

Installing the tools

- Already installed in Archer:

@ Archer

```
> ls -l /work/y14/shared/bsctools
   … clustering
   … dimemas
   … extrae
   … wxparaver
```

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Copy the examples

@ Archer

```
> cp -r /work/y14/shared/bsctools/tools-material $WORK
```

Extrae

Extrae features

- Parallel programming models
  - MPI, OpenMP, pthreads, OmpSs, CUDA, OpenCL, Java, Python…
- Platforms
  - Intel, Cray, BlueGene, MIC, ARM, Android, Fujitsu Sparc…
- Performance Counters
  - Using PAPI interface
- Link to source code
  - Callstack at MPI routines
  - OpenMP outlined routines
  - Selected user functions (Dyninst)
- Periodic sampling
- User events (Extrae API)

No need to recompile / relink!

Extrae overheads

	Average values	Archer values
Event	150 – 200 ns	160 – 170 ns
Event + PAPI	750 ns – 1 us	800 ns – 950 ns
Event + callstack (1 level)	600 ns	540 ns
Event + callstack (6 levels)	1.9 us	1.5 us

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

How does Extrae work?

- Symbol substitution through LD_PRELOAD
  - Specific libraries for each combination of runtimes
    - MPI
    - OpenMP
    - OpenMP+MPI
    - …

Recommended

- Dynamic instrumentation
  - Based on Dyninst (developed by U.Wisconsin/U.Maryland)
    - Instrumentation in memory
    - Binary rewriting

- Alternatives
  - Static link (e.g., PMPI, Extrae API)

Linking in Archer

- Cray compilers link statically by default

- How to make it dynamic?
  - By adding the –dynamic compiler flag
  - Enables tracing with LD_PRELOAD method

```
cc … -dynamic
```

Using Extrae in 3 steps

1. Adapt the job submission script

2. (Optional) Tune the Extrae XML configuration file
   – Examples distributed with Extrae at $EXTRAE_HOME/share/example

3. Run it!

- For further reference check the **Extrae User Guide:**
  – Also distributed with Extrae at $EXTRAE_HOME/share/doc
  – http://www.bsc.es/computer-sciences/performance-tools/documentation

Step 1: Adapt the job script to load Extrae (LD_PRELOAD)

job.pbs

```
#!/bin/bash --login
#PBS -N lulesh2.0
#PBS -l select=2
#PBS -l walltime=0:5:0

# This shifts to the directory …
cd $PBS_O_WORKDIR

export OMP_NUM_THREADS=1

# run the script
aprun -n 27 ./lulesh2.0 …
```

Step 1: Adapt the job script to load Extrae (LD_PRELOAD)

job.pbs

```
#!/bin/bash --login
#PBS -N lulesh2.0
#PBS -l select=2
#PBS -l walltime=0:5:0

# This shifts to the directory …
cd $PBS_O_WORKDIR

export OMP_NUM_THREADS=1
export TRACE_NAME=lulesh2.0.prv

# run the script
aprun -n 27 ./trace.sh ./lulesh2.0 …
```

Step 1: Adapt the job script to load Extrae (LD_PRELOAD)

job.pbs

```
#!/bin/bash --login
#PBS -N lulesh2.0
#PBS -l select=2
#PBS -l walltime=0:5:0

# This shifts to the directory …
cd $PBS_O_WORKDIR

export OMP_NUM_THREADS=1
export TRACE_NAME=lulesh2.0.prv

# run the script
aprun -n 27 ./trace.sh ./lulesh2.0 …
```

trace.sh

```
#!/bin/bash

# Configure Extrae
export EXTRAE_HOME=/work/y14/shared/bsctools/extrae…
source $EXTRAE_HOME/etc/extrae.sh
export EXTRAE_CONFIG_FILE=./extrae.xml

# Load the tracing library (choose C/Fortran)
export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitrace.so
#export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitracef.so

# Run the program
$*
```

Select tracing library

Step 1: Which tracing library?

- Choose depending on the application type

Library	Serial	MPI	OpenMP	pthread	CUDA
libseqtrace	✓				
libmpitrace[f][1]		✓			
libomptrace			✓		
libpttrace				✓	
libcudatrace					✓
libompitrace[f] [1]		✓	✓		
libptmpitrace[f] [1]		✓		✓	
libcudampitrace[f] [1]		✓			✓

[1] include suffix “f” in Fortran codes

Having problems with dynamic linking?

- Link statically against the tracing library (+ dependencies)
  - Only supports MPI instrumentation!
  - Insert before the actual MPI library
  - Extrae will always intercept the MPI calls
  - Don’t set LD_PRELOAD

```
-L$EXTRAE_HOME/lib -lmpitrace \
-L$BSCTOOLS_HOME/deps/binutils/2.24 -lbfd -liberty \
-L$BSCTOOLS_HOME/deps/libunwind/1.1/lib -lunwind \
-L/opt/cray/papi/5.4.1.2/lib -lpapi \
-L/usr/lib64 -lxml \
-lrt -lz -ldl
```

Step 3: Run it!

- Submit your job

@ Archer

```
> cd $WORK/tools-material/extrae

> qsub job.pbs
```

- Easy! ☺

Step 2: Extrae XML configuration

```
<mpi enabled="yes">
  <counters enabled="yes" />
</mpi>

<openmp enabled="no">
  <locks enabled="no" />
  <counters enabled="yes" />
</openmp>

<pthread enabled="no">
  <locks enabled="no" />
  <counters enabled="yes" />
</pthread>

<callers enabled="yes">
  <mpi enabled="yes">1-3</mpi>
  <sampling enabled="no">1-5</sampling>
</callers>
```

Trace MPI calls + HW counters

Trace call-stack events @ MPI calls

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Step 2: Extrae XML configuration (II)

```
<counters enabled="yes">
  <cpu enabled="yes" starting-set-distribution="cyclic">
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_L1_DCM,PAPI_L3_TCM,
      PAPI_BR_MSP
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_BR_INS,PAPI_SR_INS,
      PAPI_LD_INS
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,RESOURCE_STALLS:RS,RESOURCE_STALLS:SB
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,RESOURCE_STALLS:ROB,PAPI_L2_DCM
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      ...
    </set>
  </cpu>
  <network enabled="no" />
  <resource-usage enabled="no" />
  <memory-usage enabled="no" />
</counters>
```

Select which HW counters are measured

Step 2: Extrae XML configuration (III)

```
<buffer enabled="yes">
  <size enabled="yes">5000000</size>
  <circular enabled="no" />
</buffer>

<sampling enabled="no" type="default" period="50m" variability="10m" />

<merge enabled="yes"
       synchronization="default"
       tree-fan-out="16"
       max-memory="512"
       joint-states="yes"
       keep-mpits="yes"
       sort-addresses="yes"
       overwrite="yes"
>
  $TRACE_NAME$
</merge>
```

Trace buffer size

Enable sampling

Merge intermediate files into Paraver trace

All done! Check your resulting trace

- Once finished (check with “qstat”) you will have the trace (3 files):

@ Archer

```
> ls –l $WORK/tools-material/extrae
  ...
  lulesh2.0.pcf
  lulesh2.0.prv
  lulesh2.0.row
```

- Any trouble? Traces already generated here:

@ Archer

```
> ls –l $WORK/tools-material/traces
```

- Now let’s look into it !

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Copy the trace to your laptop

@ your computer

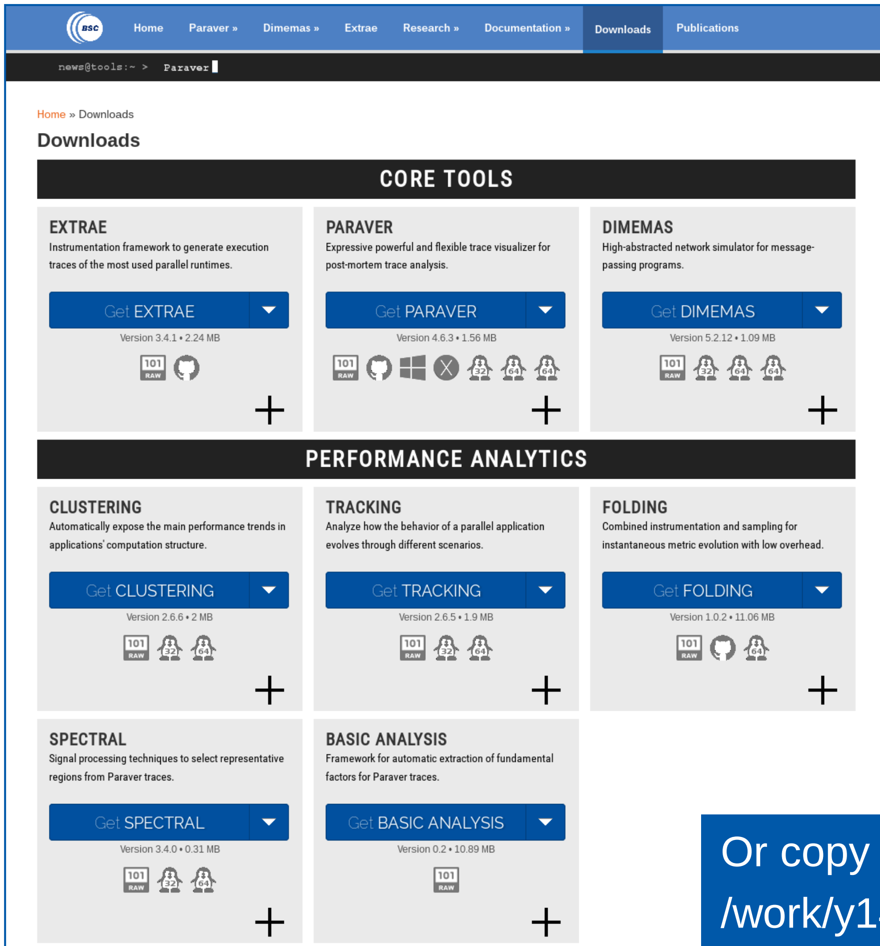
```
> scp <user>@login.archer.ac.uk:
$WORK/tools-material/extrae/lulesh2.0.* ./
```

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Paraver

Installing Paraver

- Download from https://tools.bsc.es/downloads



Or copy them from Archer @
/work/y14/shared/bsctools/tools-packages

Installing Paraver (II)

- Uncompress the package into your home directory

@ your computer

```
> tar xvfz wxparaver-4.6.3-linux-x86_64.tar.gz

> ln -s $HOME/wxparaver-4.6.3-linux-x86_64 $HOME/paraver
```

- Download Paraver tutorials and uncompress into the Paraver directory
  - https://tools.bsc.es/sites/default/files/documentation/paraver-tutorials-20150526.tar.gz

@ your computer

```
> tar xvfz $HOME/paraver-tutorials-20150526.tar.gz

> mv paraver-tutorials-20150526 $HOME/paraver/tutorials
```

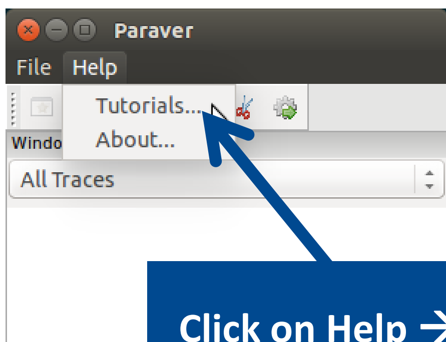
Check that everything works

Start Paraver

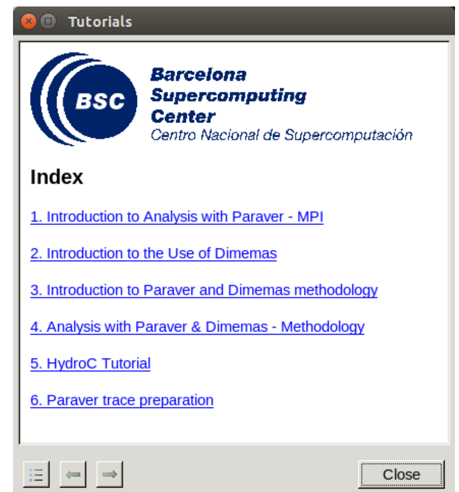
@ your computer

```
> $HOME/paraver/bin/wxparaver
```

Check that tutorials are available



Click on Help → Tutorials



Remotely available in Archer

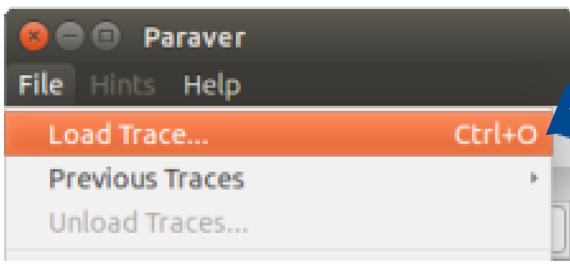
```
laptop> ssh -Y <user>@login.archer.ac.uk

archer> /work/y14/shared/bsctools/wxparaver/latest/bin/wxparaver
```

Analyze with Paraver

Start Paraver

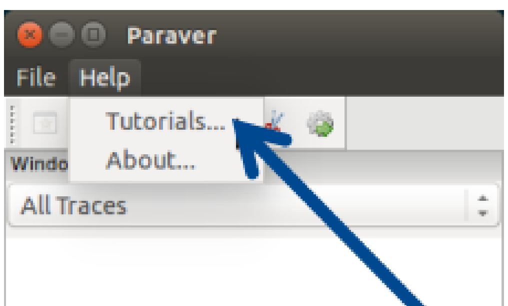
```
> $HOME/paraver/bin/wxparaver
```

Load the trace



Click on File → Load Trace → Browse to "lulesh2.0.prv"

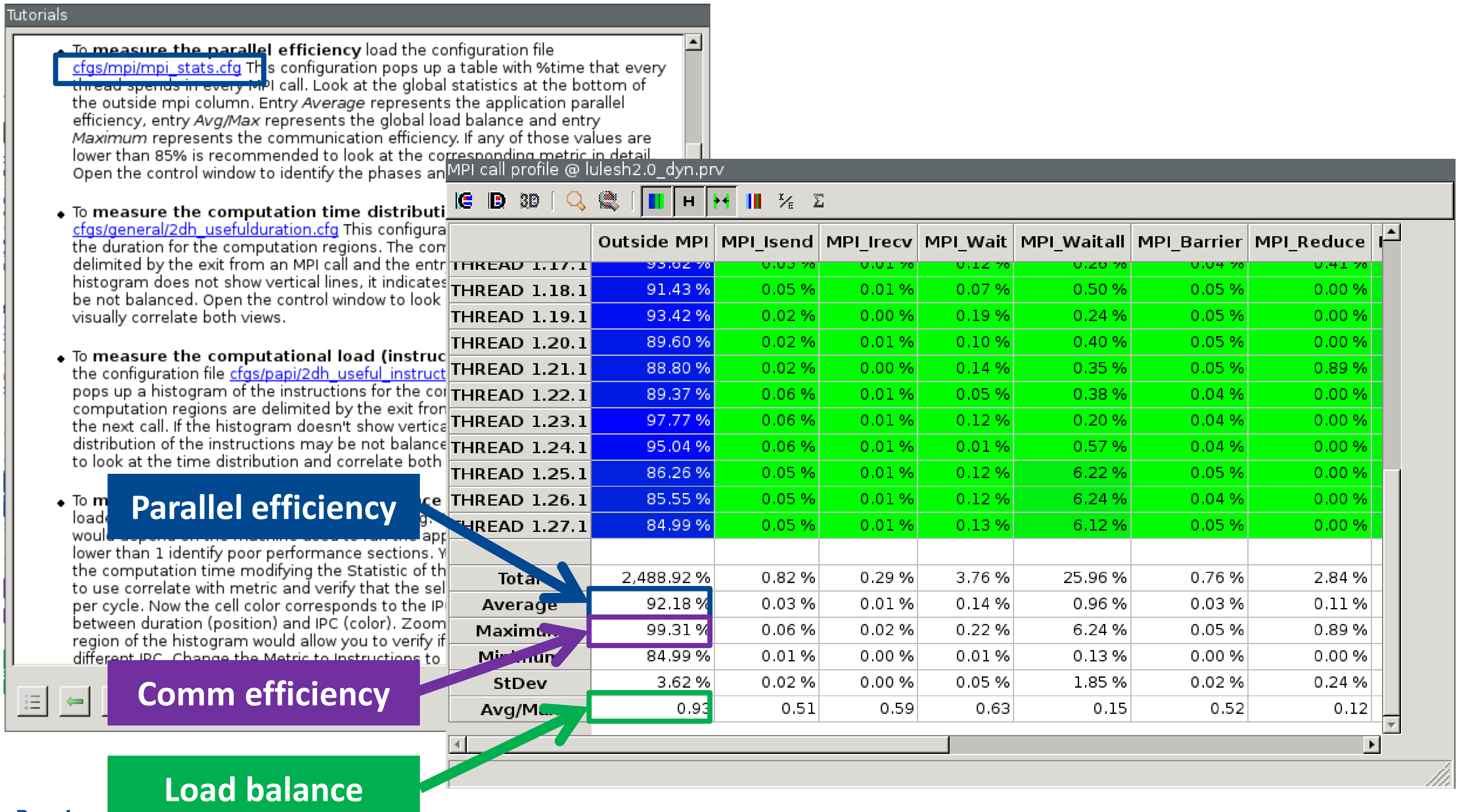
Follow tutorial #3



Click on Help → Tutorials

Measure the parallel efficiency

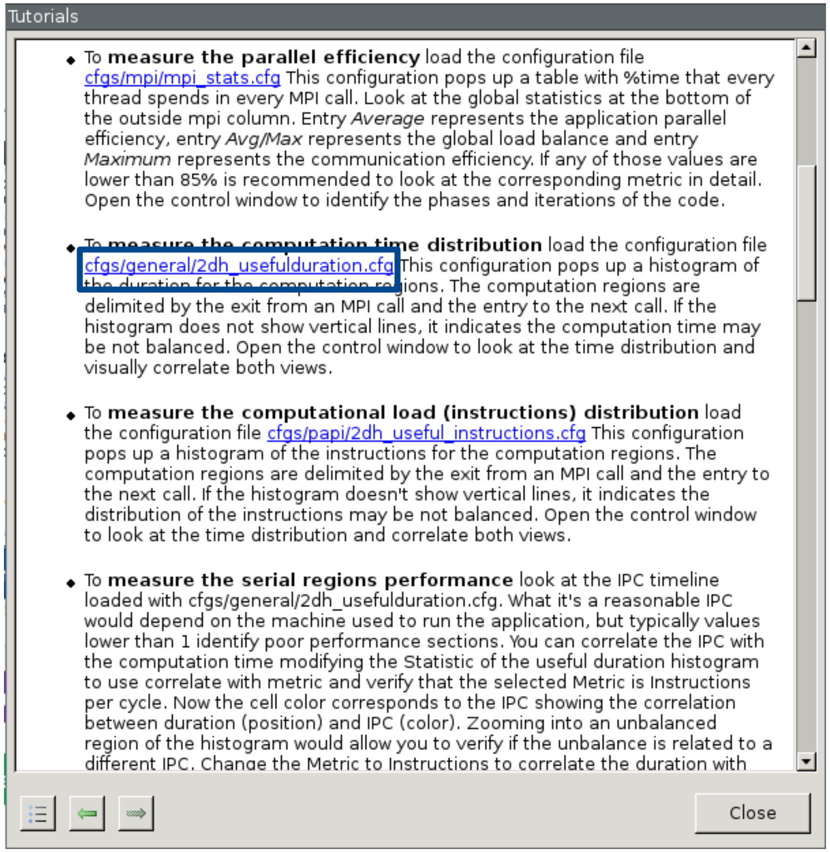
- Click on “mpi_stats.cfg”
  - Check the Average for the column labeled “Outside MPI”

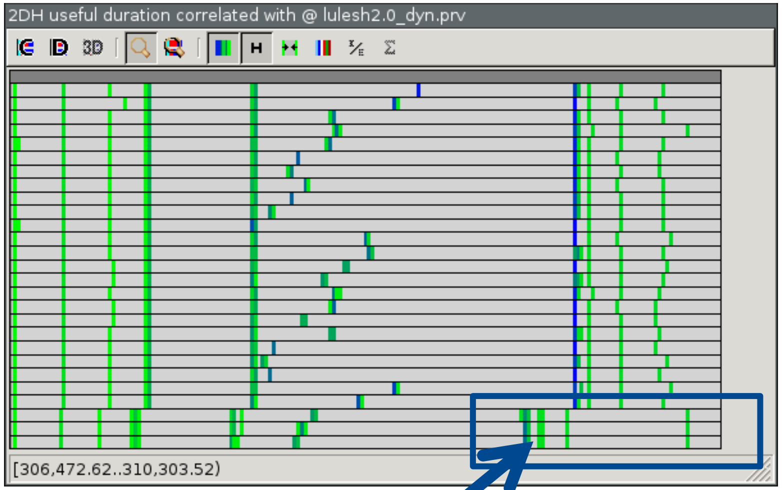


	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.18.1	91.43 %	0.05 %	0.01 %	0.07 %	0.50 %	0.05 %	0.00 %
THREAD 1.19.1	93.42 %	0.02 %	0.00 %	0.19 %	0.24 %	0.05 %	0.00 %
THREAD 1.20.1	89.60 %	0.02 %	0.01 %	0.10 %	0.40 %	0.05 %	0.00 %
THREAD 1.21.1	88.80 %	0.02 %	0.00 %	0.14 %	0.35 %	0.05 %	0.89 %
THREAD 1.22.1	89.37 %	0.06 %	0.01 %	0.05 %	0.38 %	0.04 %	0.00 %
THREAD 1.23.1	97.77 %	0.06 %	0.01 %	0.12 %	0.20 %	0.04 %	0.00 %
THREAD 1.24.1	95.04 %	0.06 %	0.01 %	0.01 %	0.57 %	0.04 %	0.00 %
THREAD 1.25.1	86.26 %	0.05 %	0.01 %	0.12 %	6.22 %	0.05 %	0.00 %
THREAD 1.26.1	85.55 %	0.05 %	0.01 %	0.12 %	6.24 %	0.04 %	0.00 %
THREAD 1.27.1	84.99 %	0.05 %	0.01 %	0.13 %	6.12 %	0.05 %	0.00 %
Tota	2,488.92 %	0.82 %	0.29 %	3.76 %	25.96 %	0.76 %	2.84 %
Average	92.18 %	0.03 %	0.01 %	0.14 %	0.96 %	0.03 %	0.11 %
Maxim	99.31 %	0.06 %	0.02 %	0.22 %	6.24 %	0.05 %	0.89 %
Min	84.99 %	0.01 %	0.00 %	0.01 %	0.13 %	0.00 %	0.00 %
StDev	3.62 %	0.02 %	0.00 %	0.05 %	1.85 %	0.02 %	0.24 %
Avg/M	0.93	0.51	0.59	0.63	0.15	0.52	0.12

Measure the computation time distribution

- Click on “2dh_usefulduration.cfg”

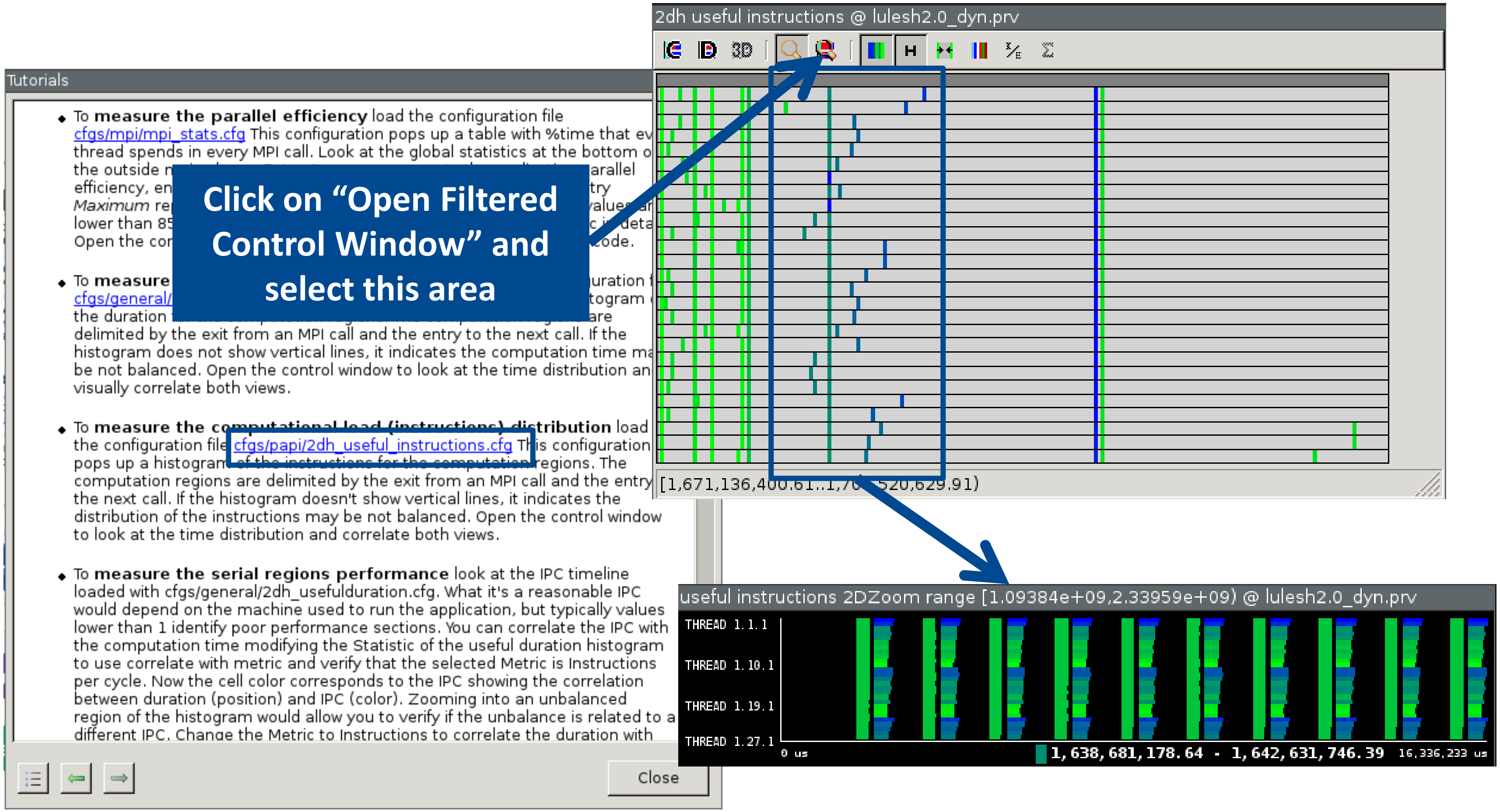




3 tasks run faster

Measure the computation time distribution

- Click on “2dh_useful_instructions.cfg”



Clustering

Cluster-based analysis

Run the clustering tool on a trace

@ Archer

```
> cd $WORK/tools-material/clustering

> ./clusterize.sh ../extrae/lulesh2.0.prv
```

– If you didn't get your own trace, you can also use a previously generated one, found it on:
  - $WORK/tools-material/traces/lulesh2.0.prv

@ your computer

```
> scp <user>@login.archer.ac.uk:$WORK/
  tools-material/clustering/*clustered* ./
```

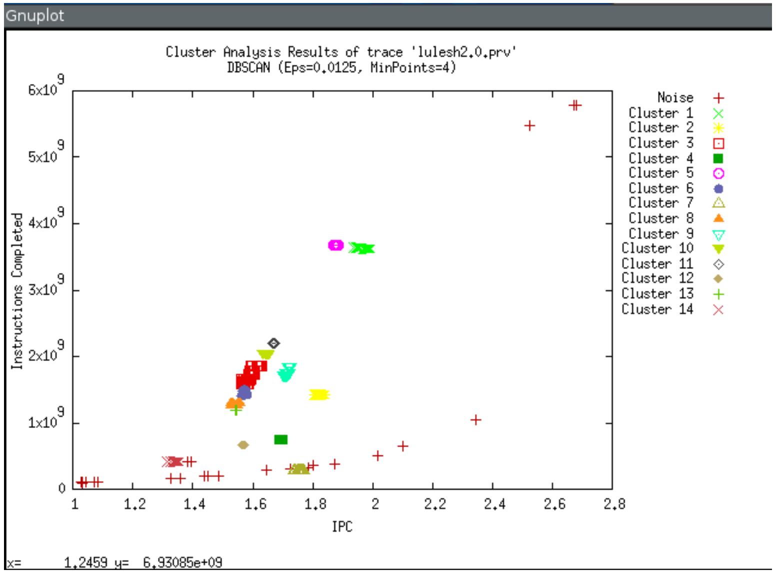
Cluster-based analysis

Check the resulting scatter plot

@ your computer

```
> gnuplot lulesh2.0.clustered.IPC.PAPI_TOT_INS.gnuplot
```

– Identify main computing trends with respect to work (Y) vs. performance (X)



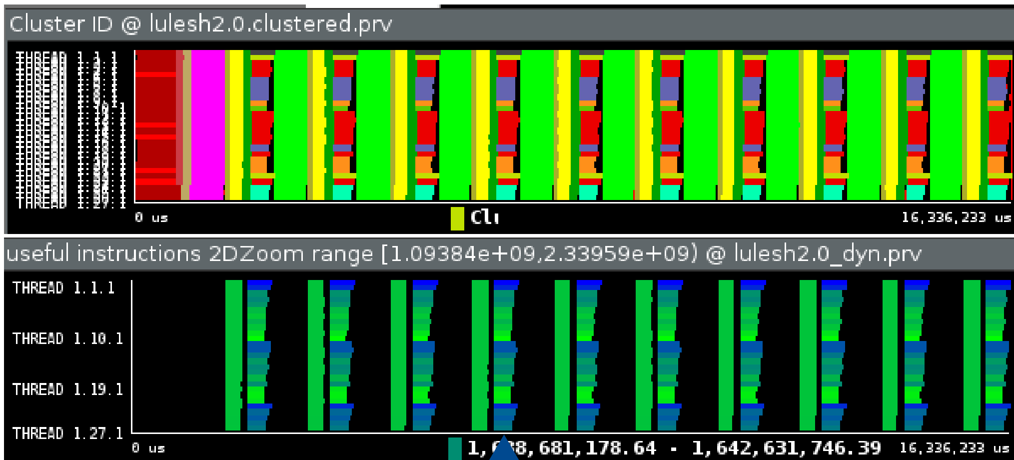
Correlate with distribution over time with Paraver

– Load clustered trace: $WORK/tools-material/clustering/lulesh2.0.clustered.prv

– File → Load configuration → $HOME/paraver/cfgs/clustering/clusterID_window.cfg

Cluster-based analysis example

- Correlate scatter plots & timelines to detect imbalances



Variable work

+

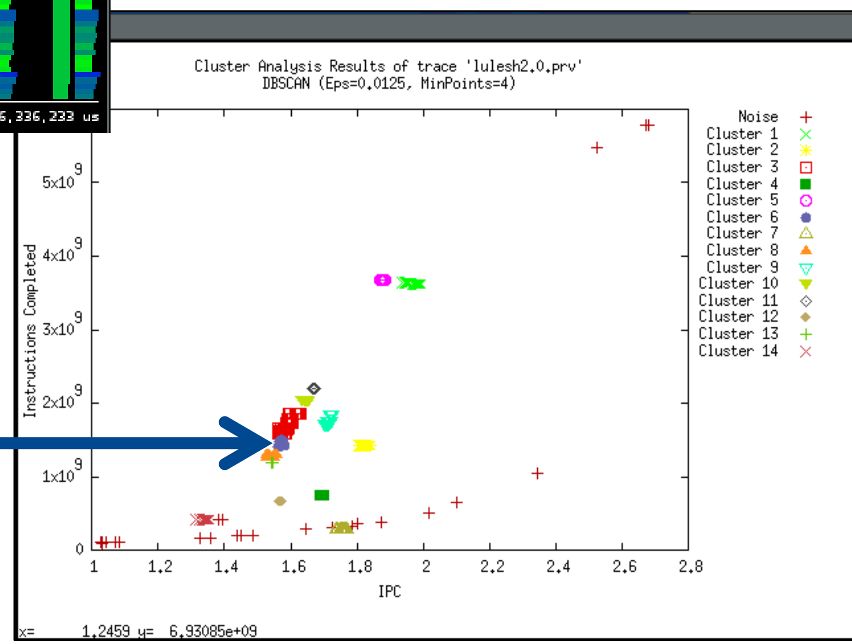
Variable performance

+

Simultaneously @ different processes

=

Imbalances



Thank you!

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

tools@bsc.es