BSC Tools
Hands-On

Judit Giménez, Lau Mercadal (lau.mercadal@bsc.es)

Barcelona Supercomputing Center

Extrae

Extrae features

- Parallel programming models
  - MPI, OpenMP(*), pthreads, OmpSs, CUDA, CUPTI, OpenCL, Java, Python…
- Platforms: Intel, Cray, BlueGene, Fujitsu Sparc, MIC, ARM, Android…
- Performance Counters
  - Using PAPI and PMAPI interfaces
- Link to source code
  - Callstack at MPI routines
  - OpenMP outlined routines and their containers
  - Selected user functions
- And more: Sampling, IO, memory allocation…
- User events (Extrae API)

No need to recompile / relink!

(*) GNU, IBM, Intel, OMPT

Extrae overheads

	Average values	CLAIX
Event	150-200 ns	140 ns
Event + PAPI	750 ns – 1 us	600 ns
Event + callstack (1 level)	600 ns	690 ns
Event + callstack (6 levels)	1.9 us	2.6 us

How does Extrae work?

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

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

- Static link (i.e., PMPI, Extrae API)

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

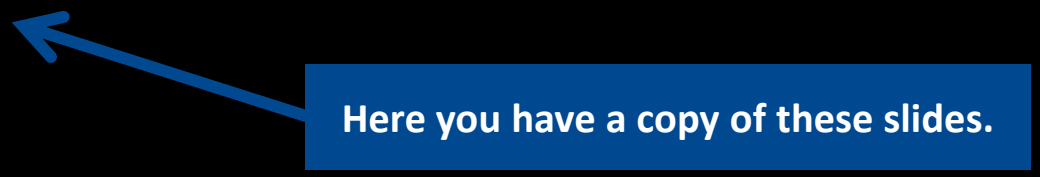
3. Run with instrumentation

- For further reference check the **Extrae User Guide:**
  - Also distributed with Extrae at $EXTRAE_HOME/share/doc

https://tools.bsc.es/tools_manuals

Log in and copy the examples to your home directory

```
> ssh -Y <USER>@login.hpc.itc.rwth-aachen.de

> cp -r /home/em440321/tools-material $HOME

> ls -l $HOME/tools-material
    ...apps
    ...extrae
    ...slides
    ...traces
```



Step 1: Adapt the job script to load Extrae with LD_PRELOAD

```
> vi $HOME/tools-material/extrae/job.lsf
```

```
#!/usr/bin/env zsh

#BSUB -J lulesh2.0
#BSUB -o lulesh2.0.out
#BSUB -e lulesh2.0.err
#BSUB -W 00:30
#BSUB -M 512
#BSUB -n 27
#BSUB -a openmpi
#BSUB -U PPCES17
#BSUB -P hpclab

export OMP_NUM_THREADS=1

# run the script
$MPIEXEC $FLAGS_MPI_BATCH
--bind-to core --map-by core ./trace.sh
lulesh2.0 ...
```

Step 1: Adapt the job script to load Extrae with LD_PRELOAD

```
> vi $HOME/tools-material/extrae/job.lsf
```

```
#!/usr/bin/env zsh

#BSUB -J lulesh2.0
#BSUB -o lulesh2.0.out
#BSUB -e lulesh2.0.err
#BSUB -W 00:30
#BSUB -M 512
#BSUB -n 27
#BSUB -a openmpi
#BSUB -U PPCES17
#BSUB -P hpclab

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

# run the script
$MPIEXEC $FLAGS_MPI_BATCH
--bind-to core --map-by core ./trace.sh
lulesh2.0 -i 10 -p -s 65
```

Step 1: Adapt the job script to load Extrae with LD_PRELOAD

```
> vi $HOME/tools-material/extrae/trace.sh
```

```
#!/usr/bin/env zsh

#BSUB -J lulesh2.0
#BSUB -o lulesh2.0.out
#BSUB -e lulesh2.0.err
#BSUB -W 00:30
#BSUB -M 512
#BSUB -n 27
#BSUB -a openmpi
#BSUB -U PPCES17
#BSUB -P hpclab

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

# run the script
$MPIEXEC $FLAGS_MPI_BATCH
--bind-to core --map-by core ./trace.sh
lulesh2.0 -i 10 -p -s 65
```

trace.sh

```
#!/bin/bash

# Configure Extrae
export EXTRAE_HOME=/rwthfs/rz/SW/unite/modules-...
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
$*
```

Pick a tracing library

Step 1: LD_PRELOAD library selection

- 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

Step 3: Run with instrumentation

- Submit your job

@ CLAIX

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

> bsub < job.lsf
```

Step 2: Extrae XML configuration: extrae.xml

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

<openmp enabled="yes">
  <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

Step 2: Extrae XML configuration: extrae.xml (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_INS
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_BR_MSP,RESOURCE_STALLS,
      PAPI_SR_INS
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_L2_DCM,PAPI_LD_INS
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_VEC_SP
    </set>
    <set enabled="yes" domain="all" changeat-time="500000us">
      ...
    </set>
  </cpu>
  <network enabled="no" />
  <resource-usage enabled="no" />
  <memory-usage enabled="no" />
</counters>
```

Define which HW counters are measured

Step 2: Extrae XML configuration: extrae.xml (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="27"
       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

Check the resulting trace

- After the execution you will get the trace (3 files):

@ CLAIX

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

- Any trouble? Traces already generated here:

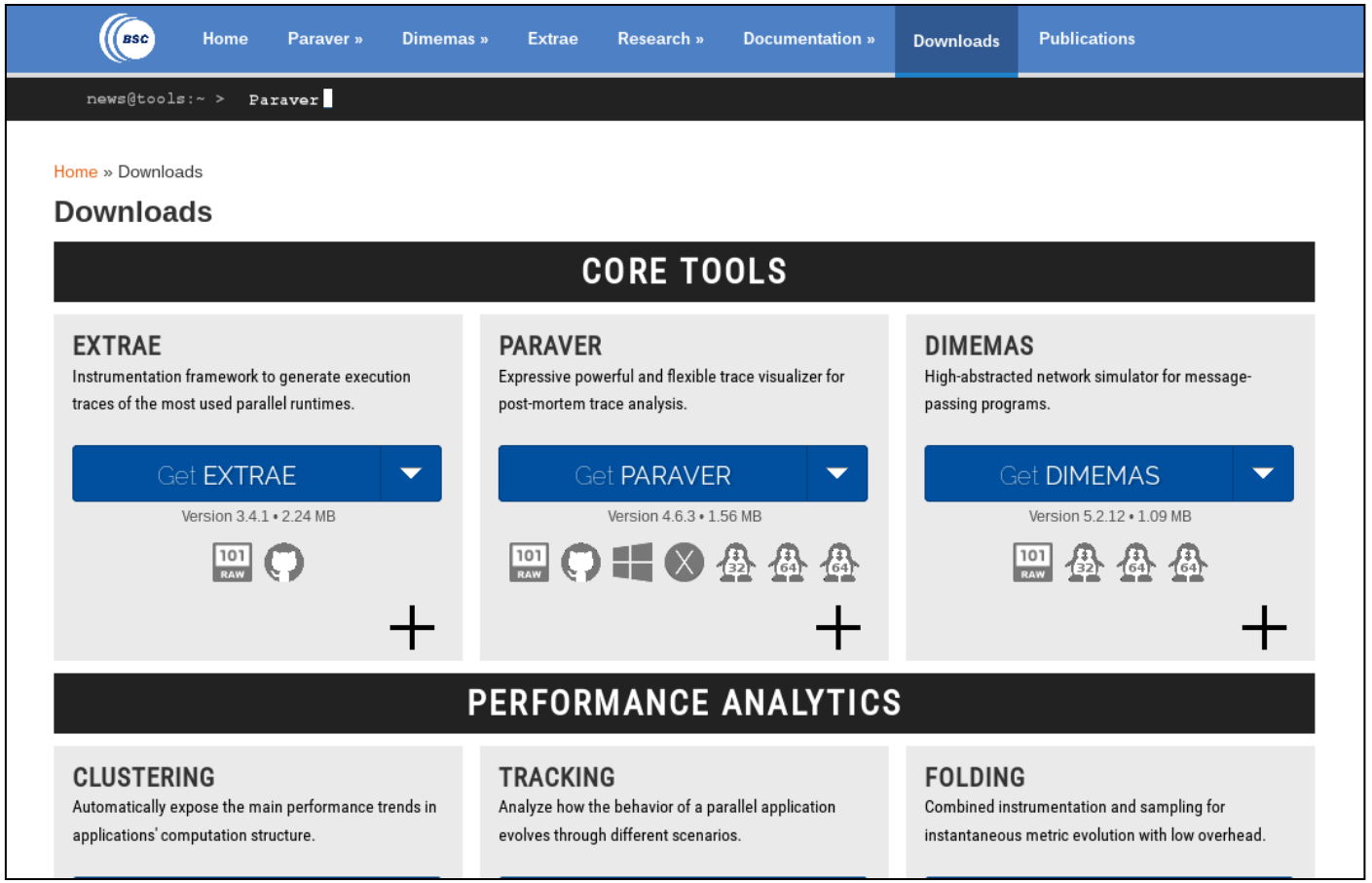
@ CLAIX

```
> cd $HOME/tools-material/traces
```

Paraver

Installing Paraver

- Download the Paraver binaries from our website



Installing Paraver

- Or copy them from CLAIX:

@ your computer

```
> scp <USER>@login.hpc.itc.rwth-aachen.de:/home/em440321/
  tools-packages/<VERSION> $HOME
```

Pick your version

Linux 64 bits	`wxparaver-4.6.3-linux-x86_64.tar.gz`
Linux 32 bits	`wxparaver-4.6.3-linux-x86.tar.gz`
Mac	`wxparaver-4.6.3-mac.zip`
Windows	`wxparaver-4.6.3-win.zip`

Installing Paraver

- 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

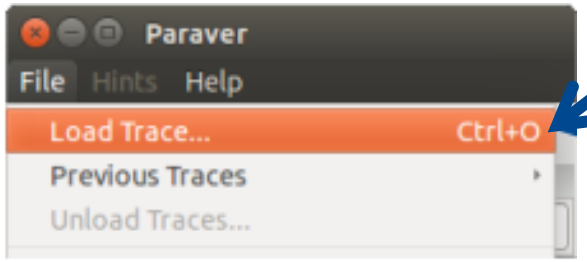
```
> tar xvfz $HOME/paraver-tutorials-20150526.tar.gz

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

Check that everything works

- Start Paraver

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

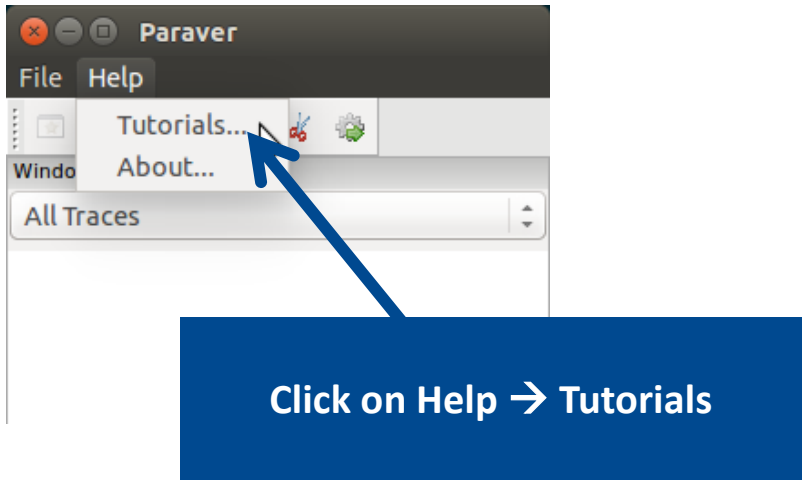
- Load the trace



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

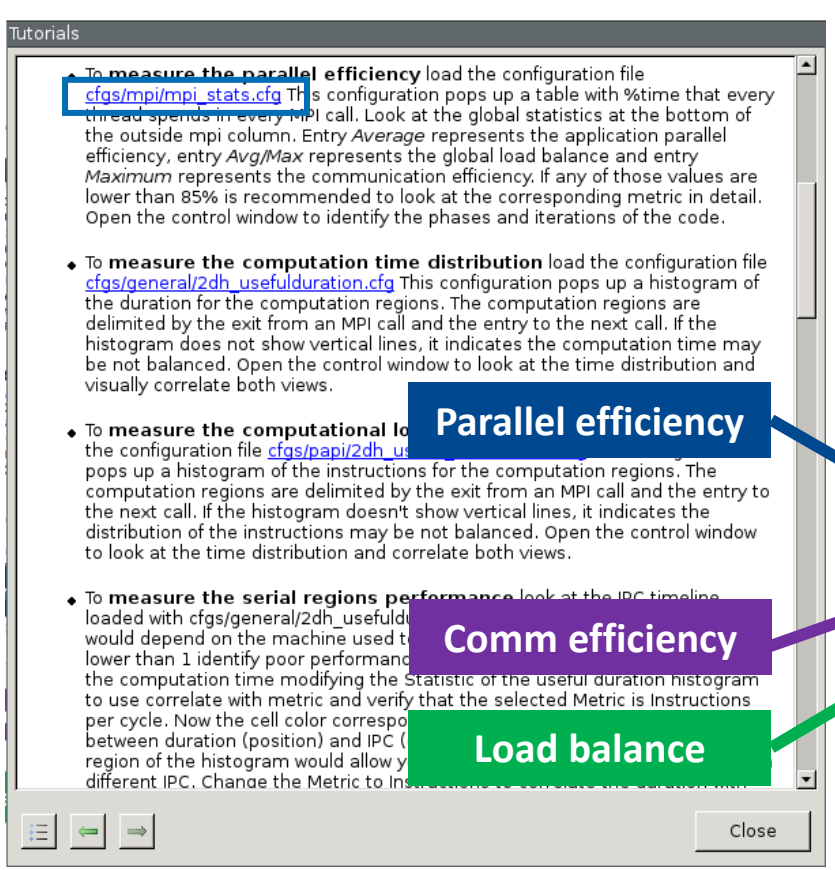
Check that everything works

- Check that tutorials are available & follow #3



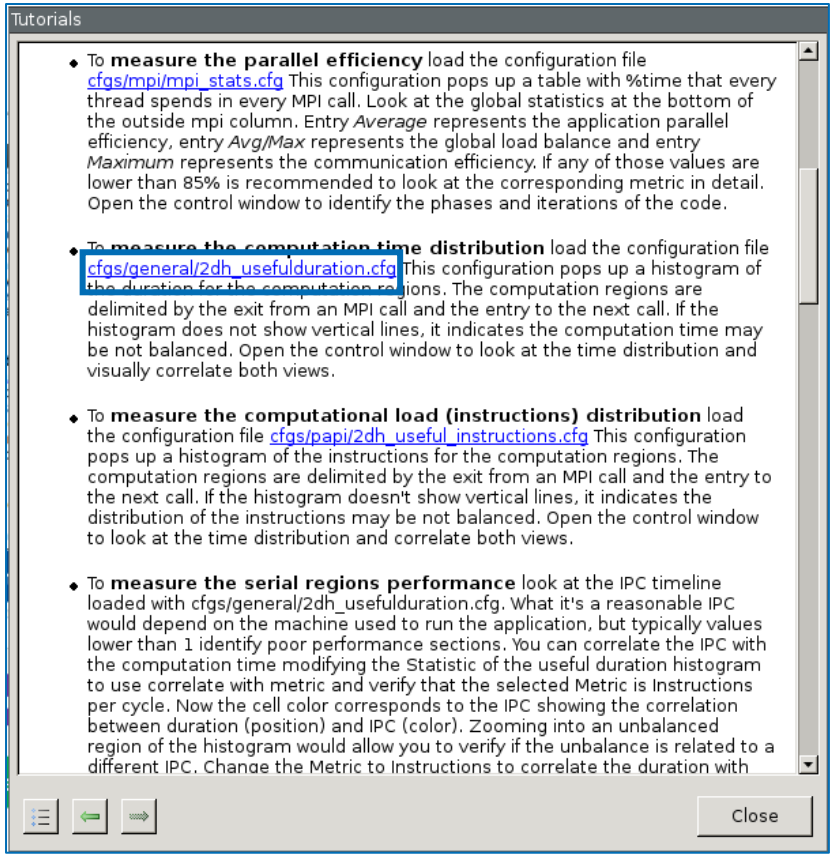
Measure the parallel efficiency

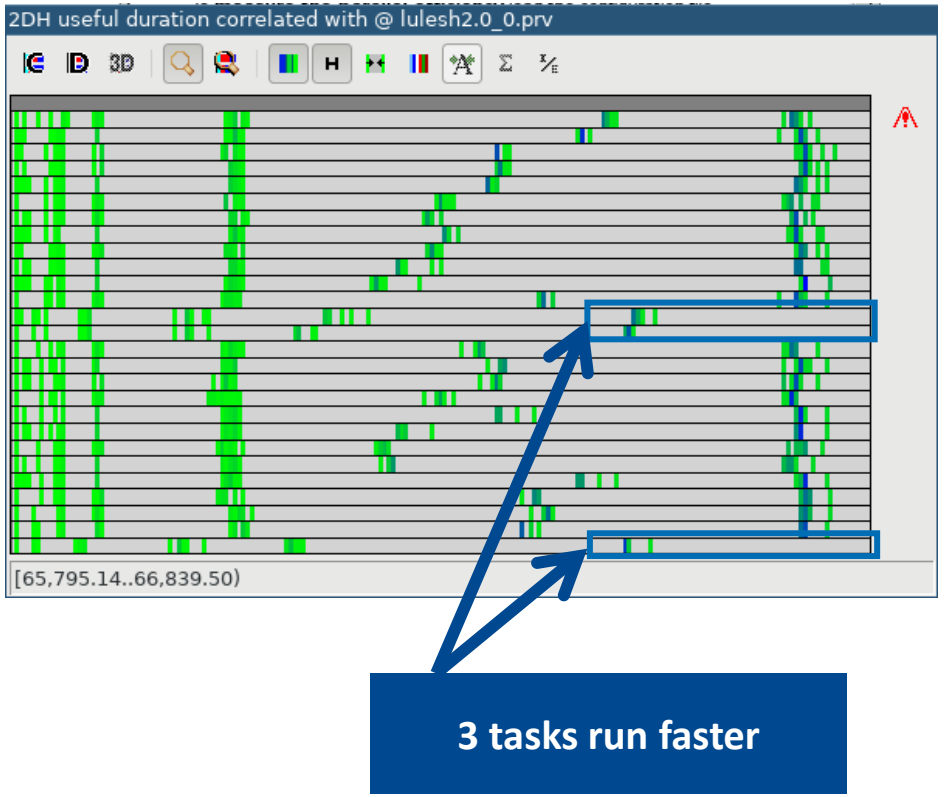
- Click on the "mpi_stats.cfg"
  - Check the Average for the column labeled "Outside MPI"



Measure the computation time distribution

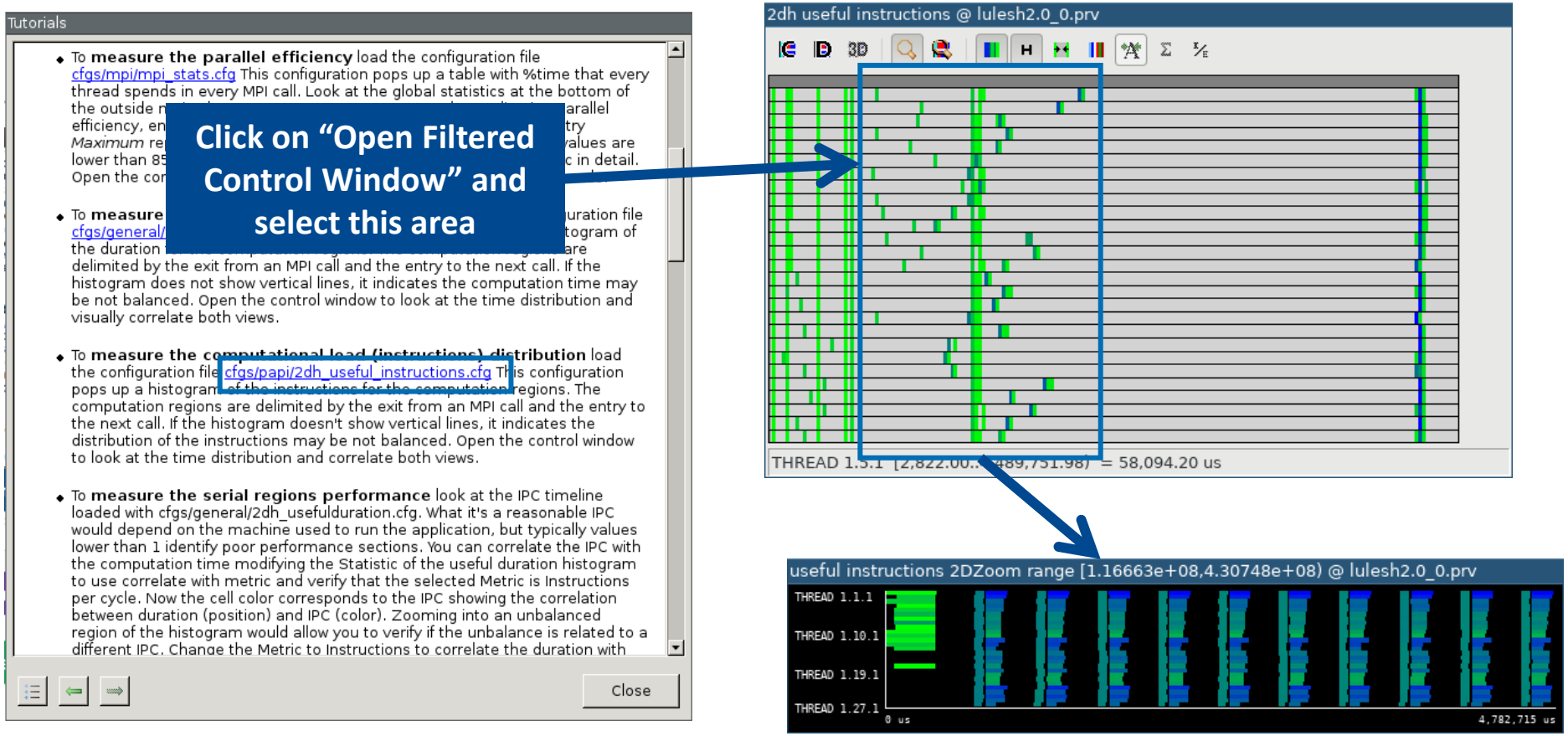
- Click on the “2dh_usefulduration.cfg”





Measure the computation time distribution

- Click on the "2dh_usefulduration.cfg"



Cluster analysis

Cluster-based analysis

- Run clustering

@ CLAIX

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

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

- If you didn’t get your own trace, you can use a previously generated one:

@ CLAIX

```
> cd $HOME/tools-material/traces
```

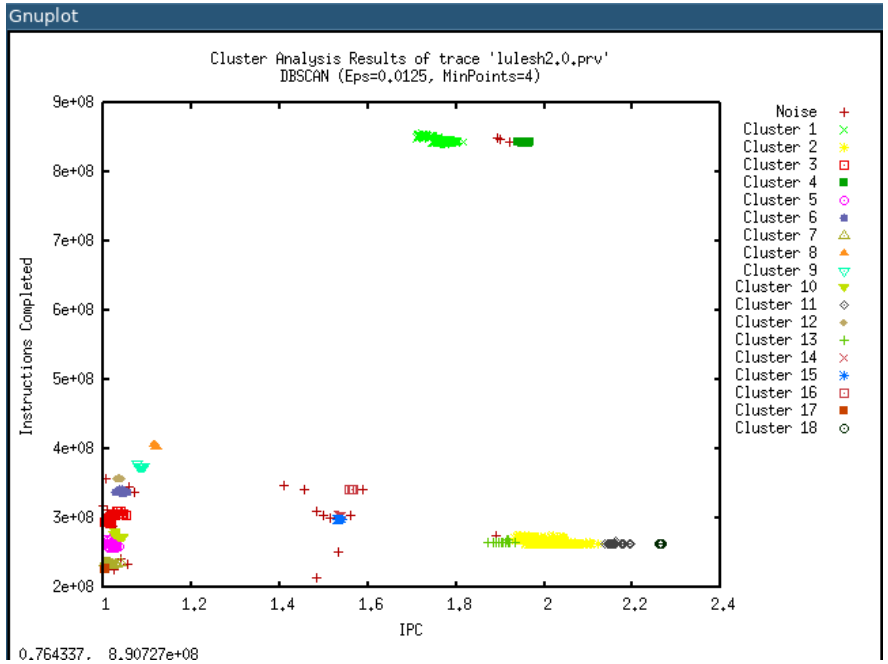
Looking at the clusters

- Check the clustering scatter plot

@ CLAIX

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

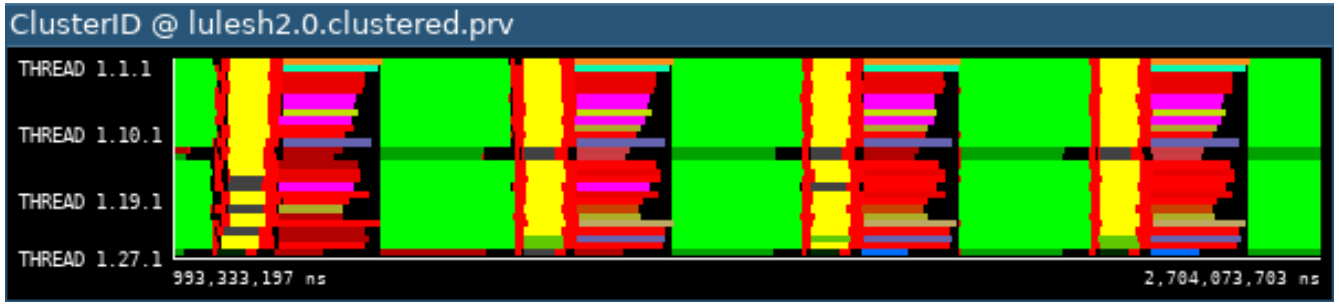
- Identify main computing trends
  - Work (Y-axis), Performance (X-axis)
- See the elongated clusters?
  - Large IPC variability
  - Wide range of instructions
  - Indicate potential imbalances



Looking at the clustered trace

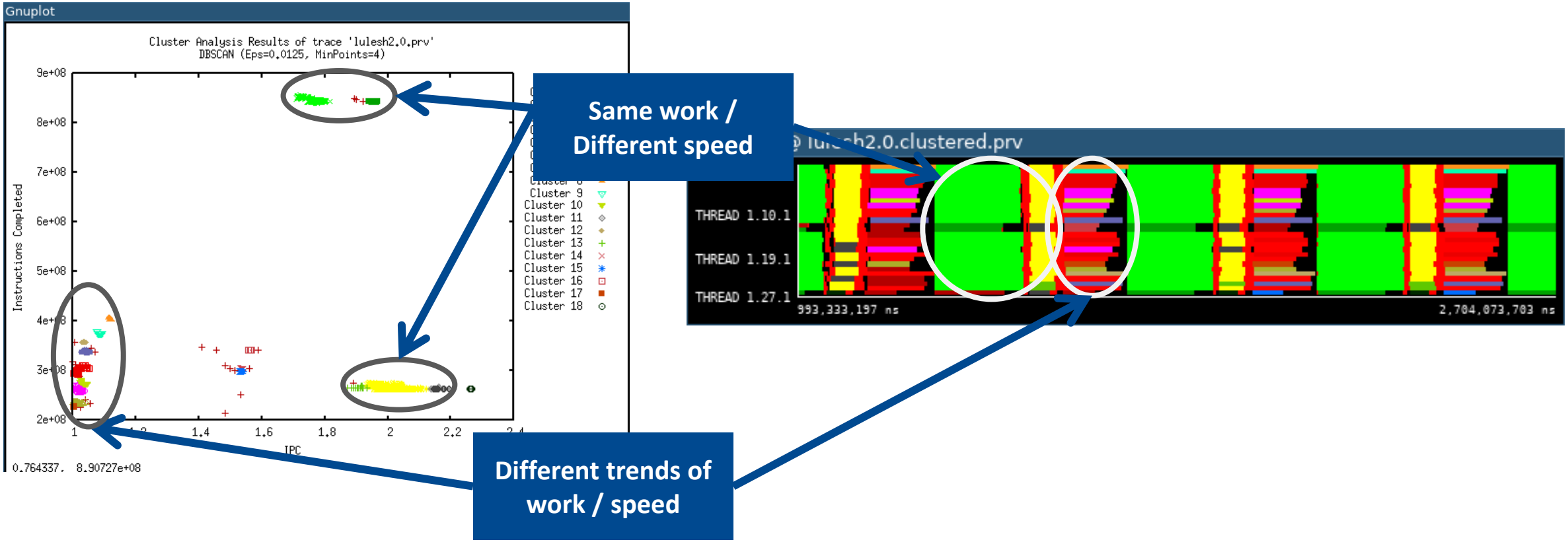
- Load the clustered trace with Paraver
  - File →Load Trace → Browse to **$HOME/tools-material/clustering/lulesh2.0.clustered.prv**

- Display the distribution of clusters over time
  - File → Load configuration → Browse to **$PARAVER_HOME/cfgs/clustering/clusterID_window.cfg**



Looking at the clustered trace

- Correlate scatter-plots and timelines to detect imbalances



BSC Tools
Hands-On

Judit Giménez, Lau Mercadal (lau.mercadal@bsc.es)

Barcelona Supercomputing Center