BSC Tools
Hands-On

Germán Llort, Judit Giménez

Barcelona Supercomputing Center

Getting a trace with Extrae

Extrae features

- Parallel programming model
  - MPI, OpenMP, pthreads, OmpSs, CUDA, OpenCL, Java, Python, etc.
- Platforms: Intel, Cray, BlueGene, Fujitsu Sparc, Intel 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
- Periodic samples
- User events (Extrae API)

No need to recompile / relink!

Extrae overheads

	Average values	INTI (Haswell)
Event	150-200 ns	140 ns
Event + PAPI	750 ns – 1 us	816 ns
Event + callstack (1 level)	600 ns	719 ns
Event + callstack (6 levels)	1.9 us	1.8 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)

How to use Extrae?

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

  http://www.bsc.es/computer-sciences/performance-tools/documentation

Login to INTI and copy the examples

```
> ssh -X <USER>@inti.ocre.cea.fr

> module load datadir/formation

> cp -r $FORMATION_HOME/BSC_Tools/tools-material $SCRATCHDIR

> ls $SCRATCHDIR/tools-material
    … apps/
    … clustering/
    … extrae/
    … traces/
```

Adapt the job script to load Extrae with LD_PRELOAD

```
> vi $SCRATCHDIR/tools-material/extrae/job.mpi
```

```
#!/bin/bash
#MSUB -r job.mpi
#MSUB -n 27
#MSUB -T 600
#MSUB -o job_%I.o
#MSUB -e job_%I.e
#MSUB -q haswell

set -x
cd ${BRIDGE_MSUB_PWD}

ccc_mprun -n 27
../apps/lulesh/lulesh2.0 -i 10 -p -s 65
```

Adapt the job script to load Extrae with LD_PRELOAD

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

```
#!/bin/bash
#MSUB -r job.mpi
#MSUB -n 27
#MSUB -T 600
#MSUB -o job_%I.o
#MSUB -e job_%I.e
#MSUB -q haswell

set -x
cd ${BRIDGE_MSUB_PWD}

ccc_mprun -n 27 ./trace.sh
../apps/lulesh/lulesh2.0 -i 10 -p -s 65
```

```
#!/bin/bash

export EXTRAE_HOME=<installation-path>

export EXTRAE_CONFIG_FILE=./extrae.xml

#export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitracef.so  #Fortran
export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitrace.so    #C

$@
```

Select tracing library

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

Run with instrumentation

- Submit your job

@ inti

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

> ccc_msub job.mpi -E "--reservation=vihps"
```

- Once finished the trace will be in the same folder: lulesh2.0.{pcf,prv,row} **(3 files)**

- Any issue?
  - Already generated at $SCRATCHDIR/tools-material/traces

Extrae XML configuration: extrae.xml

```
> vi $SCRATCHDIR/tools-material/extrae/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 & measure HW counters

Add call-stack events at MPI calls

Extrae XML configuration: extrae_config.xml (II)

```
<counters enabled="yes">
  <cpu enabled="yes" starting-set-distribution="cyclic">
     <set enabled="yes" changeat-time="500000us" domain="all“>
       PAPI_TOT_INS, PAPI_TOT_CYC, PAPI_L2_DCM, PAPI_L3_TCM
     </set>
    <set enabled="yes" changeat-time="500000us" domain="all">
       PAPI_TOT_INS,PAPI_TOT_CYC,RESOURCE_STALLS
    </set>
    <set … /set>
  </cpu>

  <network enabled="no" />

  <resource-usage enabled="no" />

  <memory-usage enabled="no" />
</counters>
```

Define which HW counters are measured

Extrae XML configuration: extrae_config.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="16"
       max-memory="512"
       joint-states="yes"
       keep-mpits="yes"
       sort-addresses="yes"
       overwrite="yes"
/>
```

Trace buffer size

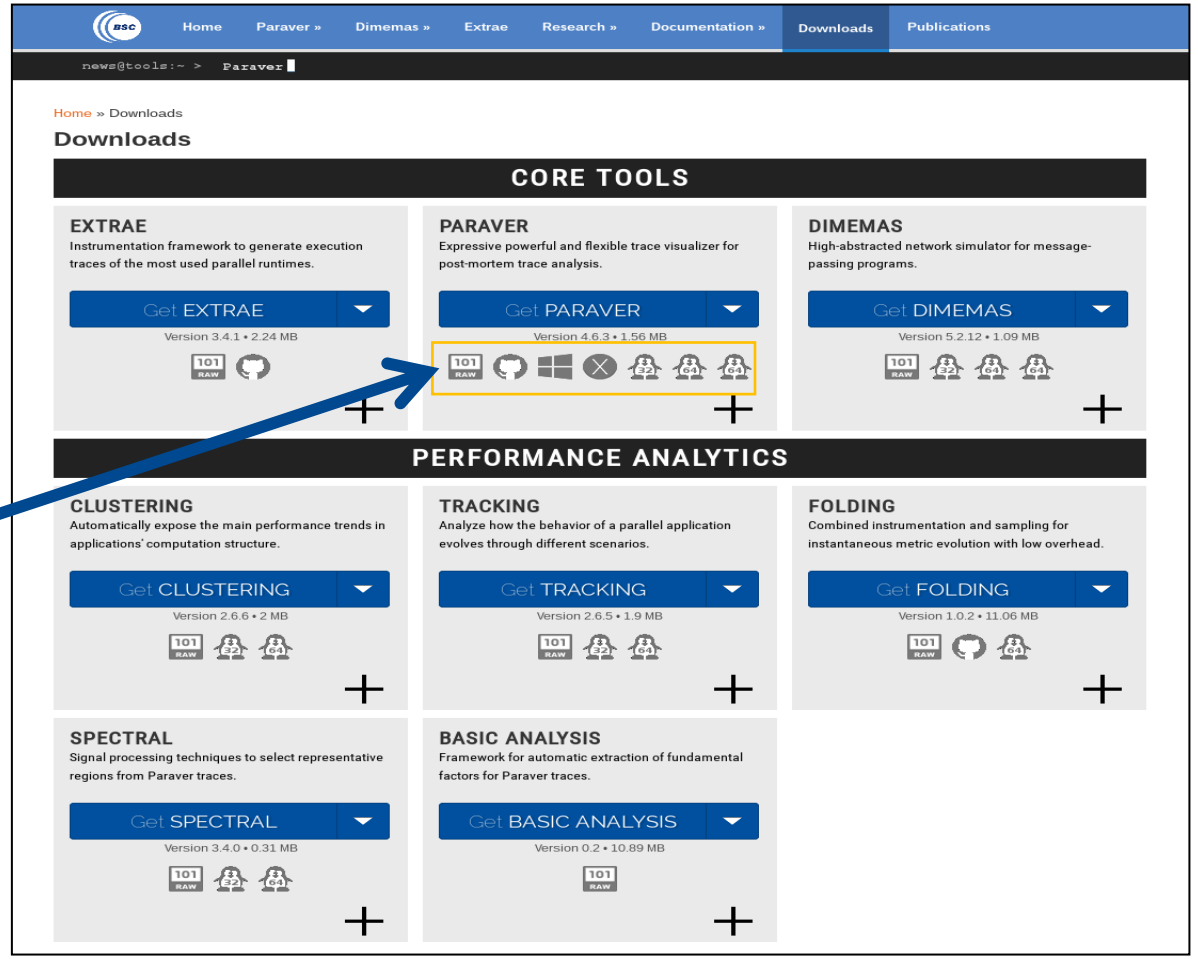
Merge intermediate files into Paraver trace

Installing Paraver
&
First analysis steps

Install Paraver in your laptop

- Download from http://tools.bsc.es

- Also available @inti
  - $FORMATION_HOME/BSC_Tools/pkgs



Install Paraver in your laptop (II)

- Uncompress into your home directory

@ your laptop

```
> tar xvfz wxparaver-4.6.4.rc1-linux-x86_64.tar.gz -C $HOME
> cd $HOME
> ln -s wxparaver-4.6.4.rc1-linux-x86_64 paraver
```

- Download Paraver tutorials and uncompress into the Paraver directory
  - From website: https://tools.bsc.es/sites/default/files/documentation/paraver-tutorials-20150526.tar.gz
  - From @inti: $FORMATION_HOME/BSC_Tools/pkgs/paraver-tutorials-20150526.tar.gz

@ your laptop

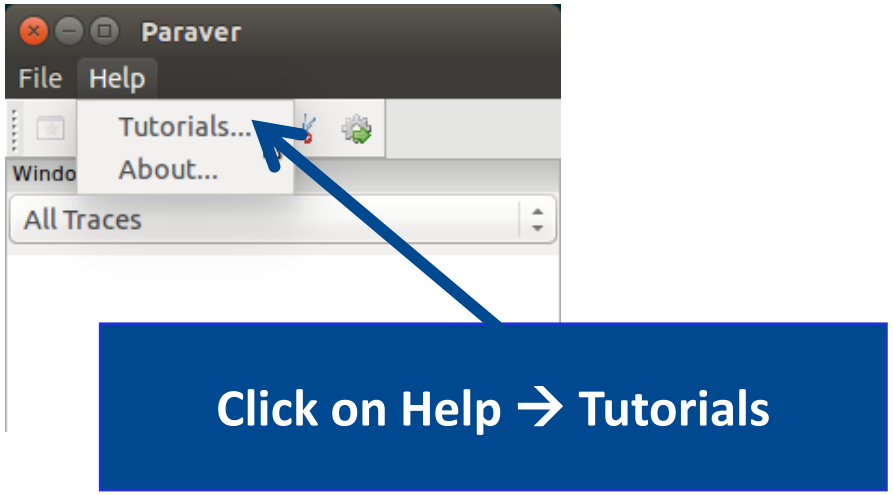
```
> tar xvfz paraver-tutorials-20150526.tar.gz -C $HOME/paraver
```

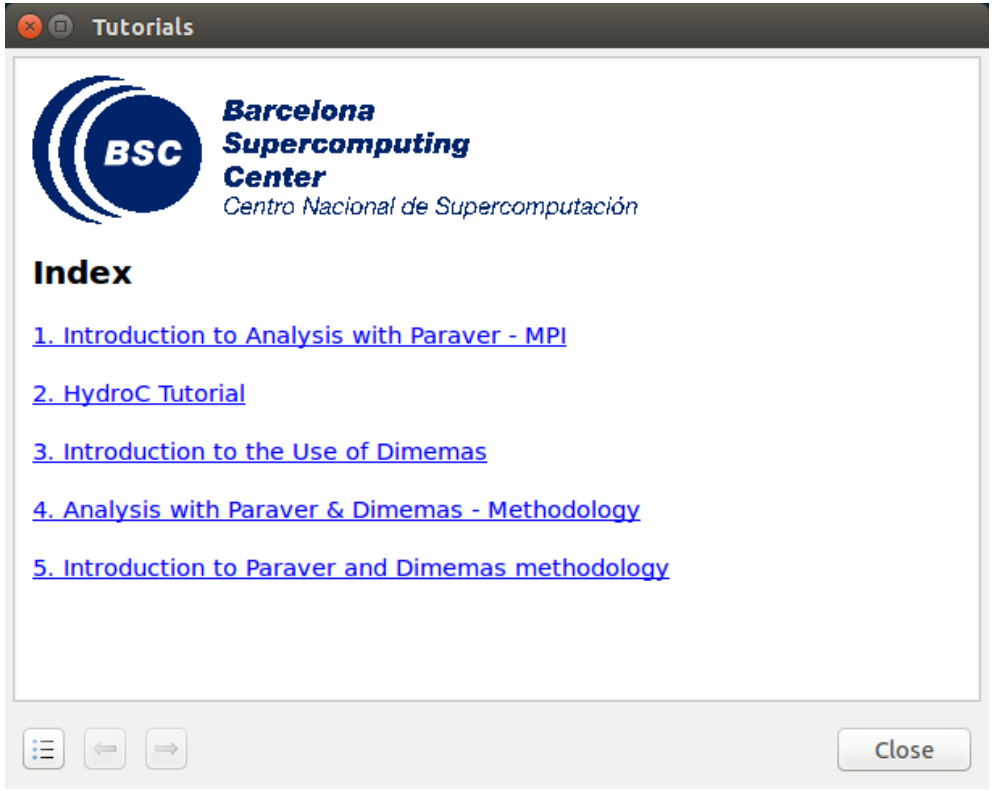
Launch Paraver

@ your laptop

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

- Check that tutorials are available





Launch Paraver (II)

- Any issues installing in your laptop? Also available @ INTI

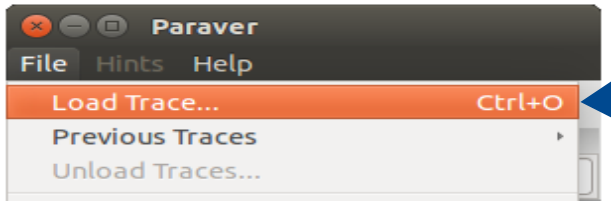
@ inti

```
> cd $FORMATION_HOME/BSC_Tools/tools/wxparaver-4.6.4/bin

> ./wxparaver $SCRATCHDIR/tools-material/extrae/lulesh2.0.prv &
```

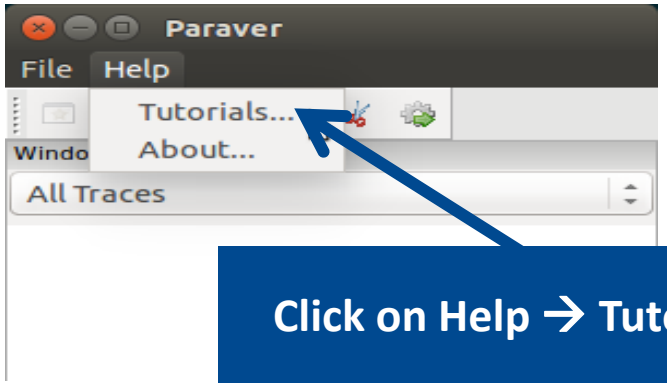
First steps of analysis

- Copy the trace to your desktop (All 3 files: *.prv, *.pcf, *.row)
- Load the trace

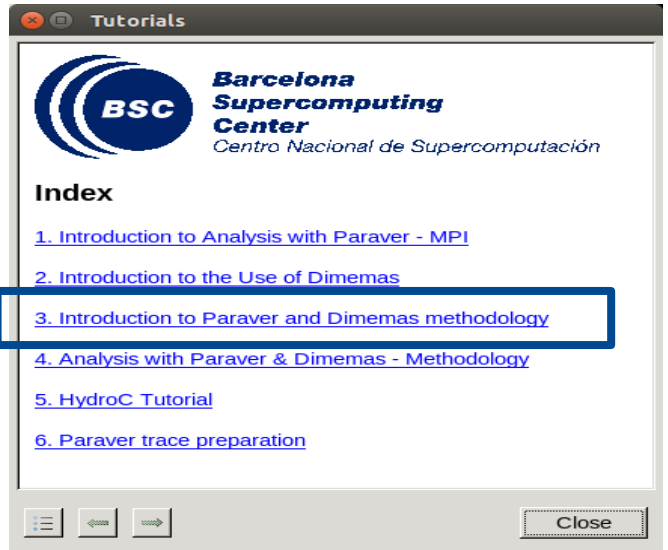


Click on File → Load Trace → Browse to the *.prv file

- Follow Tutorial #3
  - Introduction to Paraver and Dimemas methodology

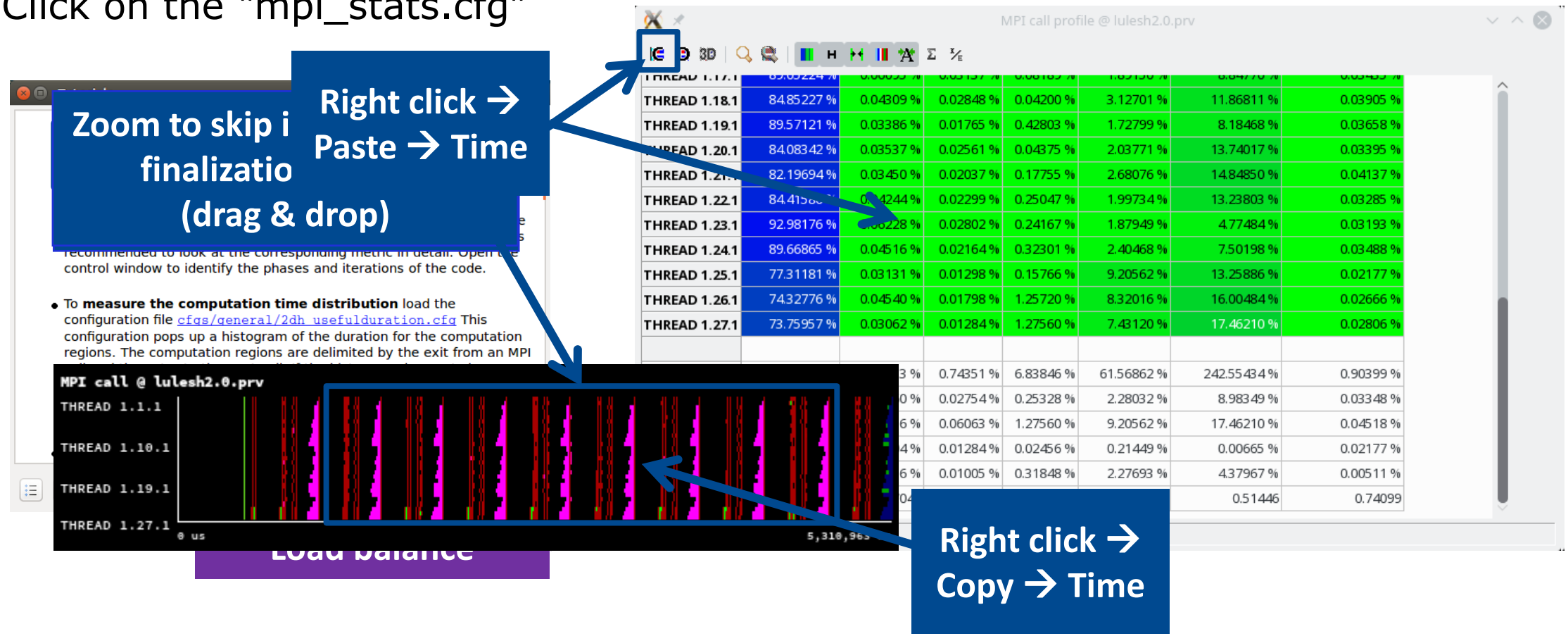


Click on Help → Tutorials



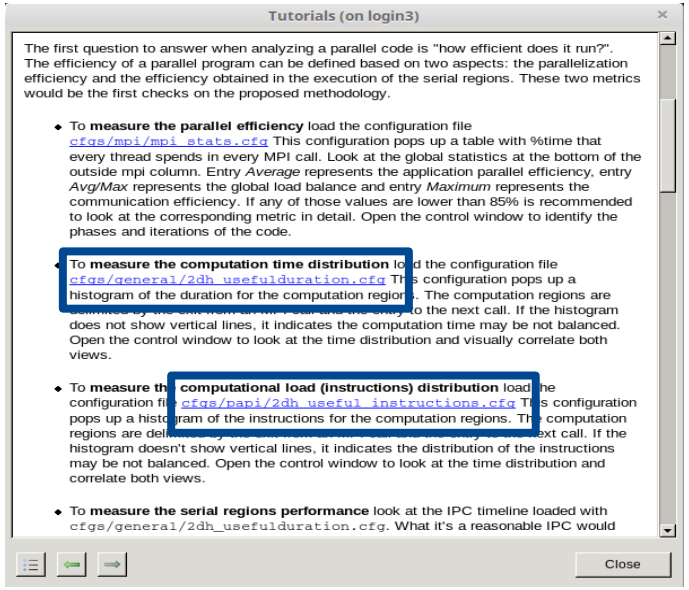
Measure the parallel efficiency

- Click on the "mpi_stats.cfg"

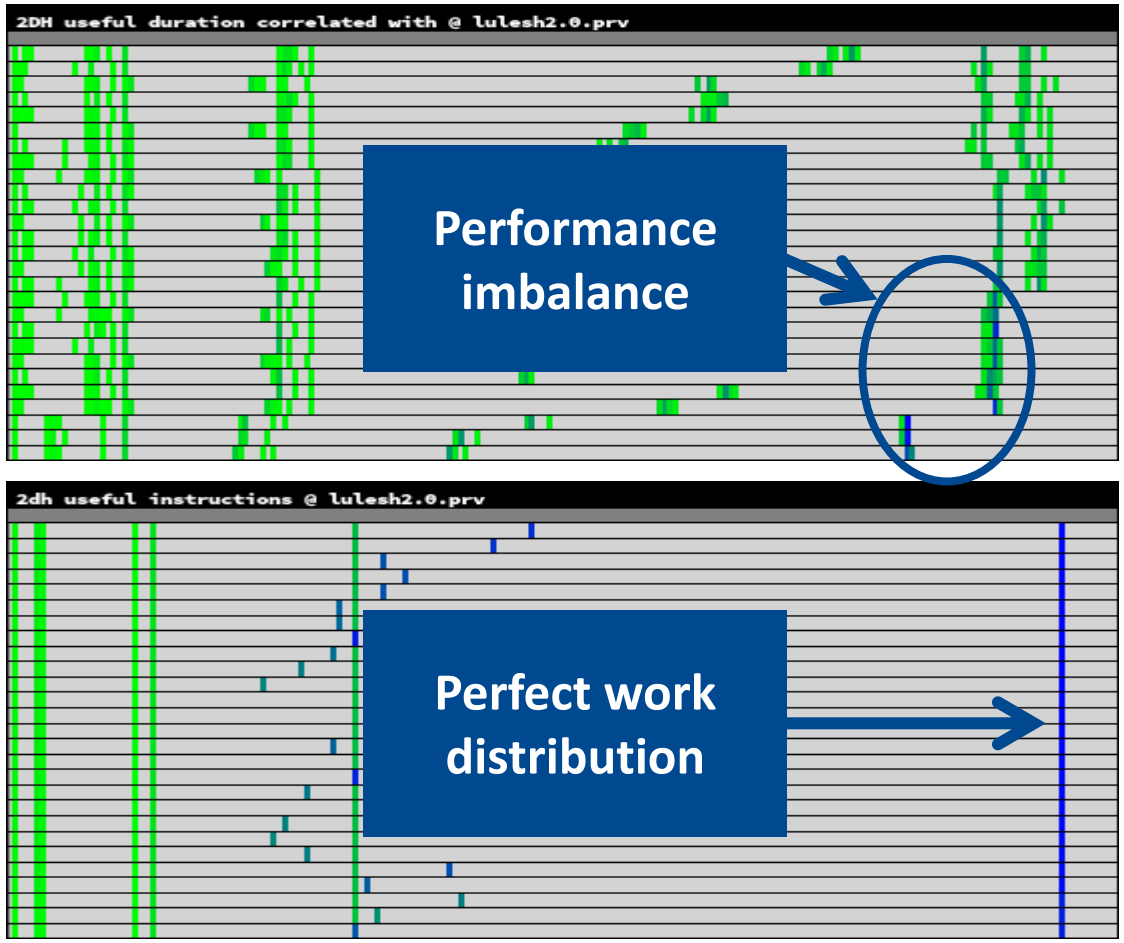


Measure the computation time distribution

- Click on “2dh_usefulduration.cfg”



- …and “2dh_useful_instructions.cfg”



Cluster-based analysis

Use clustering analysis

- Run clustering

@ inti

```
> cd $SCRATCHDIR/tools-material/clustering
> ./clusterize.sh ../extrae/lulesh2.0.prv
```

- Look at the results

@ inti

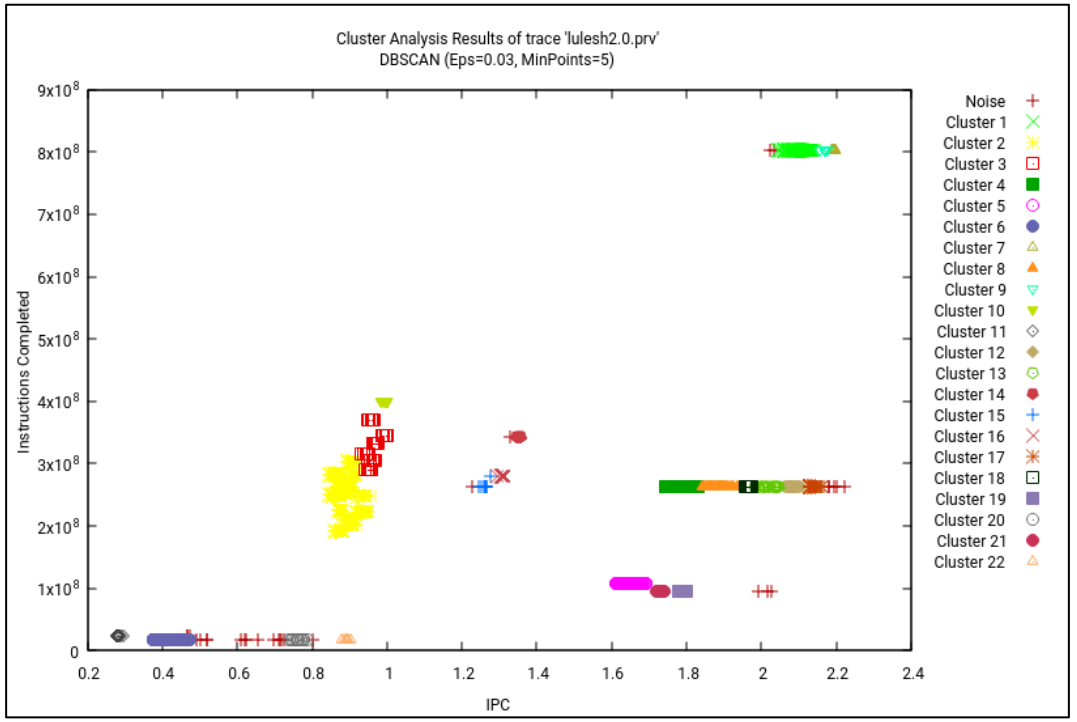
```
> gnuplot lulesh2.0_clustered.IPC.PAPI_TOT_INS.gnuplot
```

Cluster-based analysis

- Check the resulting scatter plot

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

Identify main computing trends with respect to work & performance



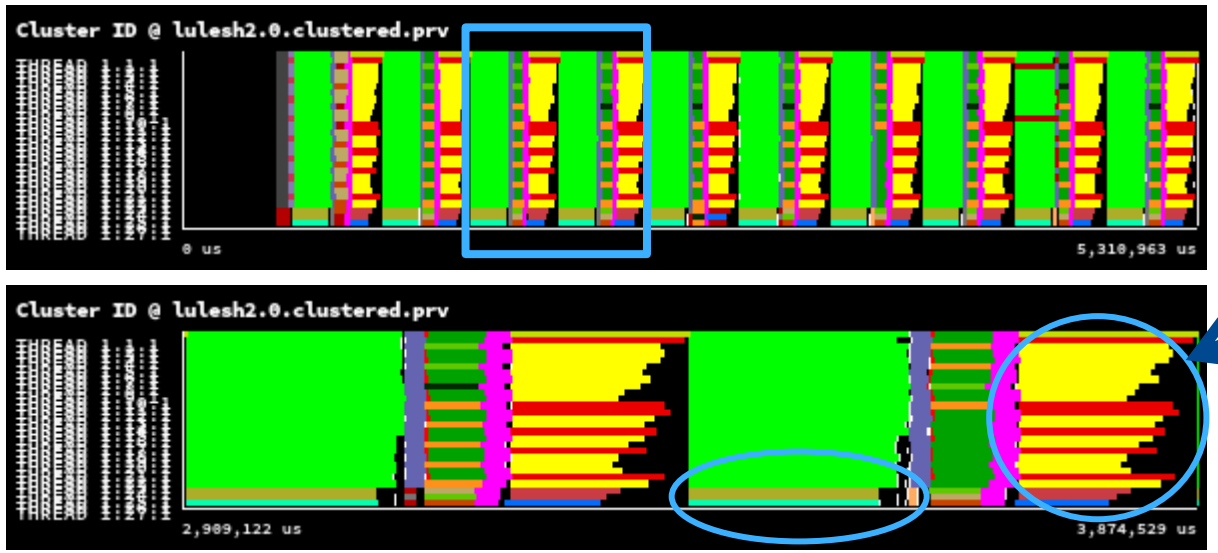
Correlating scatter plot and time distribution

- Copy the clustered trace to your laptop and look at it

@ your laptop

```
> $HOME/paraver/bin/wxparaver <path-to>/lulesh2.0_clustered.prv
```

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



Variable work / performance + Simultaneously @ different processes = Imbalances