BSC Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA
SEVERO
OCHOA

Extrae & Paraver Hands-On

tools@bsc.es

2018

Copy files for the hands-on

- You can download the material for most of the hands on from the web site https://tools.bsc.es/tools-hands-on.
- No binaries are provided, but you can follow the Extrae part with your own code.

```
> ls -l tools-material
    … clustering/
    … dimemas/
    … extrae/
    … traces/
```

Using Extrae in **3 steps**

1. **Adapt** your job submission scripts

2. **Configure** what to trace (optional)
   - XML configuration file
   - Example configurations at $EXTRAE_HOME/share/example

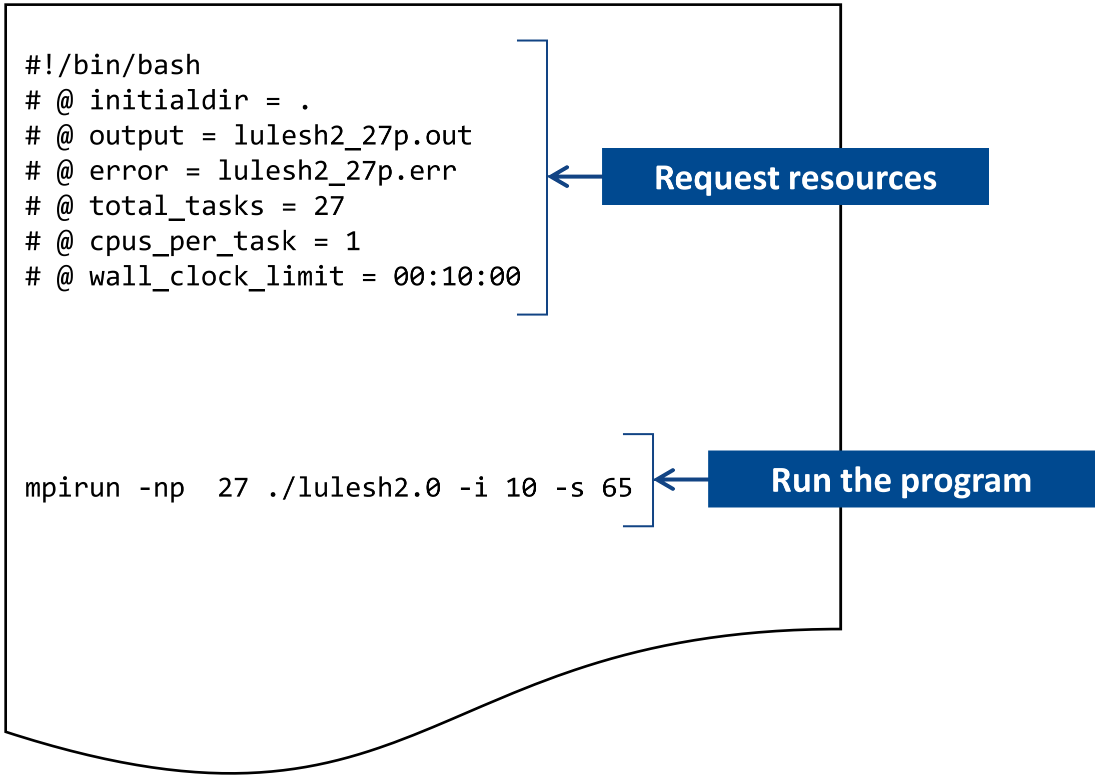
3. **Run** it!

- For further reference check the **Extrae User Guide:**
  - https://tools.bsc.es/sites/default/files/documentation/html/extrae/index.html
  - Also distributed with Extrae at $EXTRAE_HOME/share/doc

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

```
> vi tools-material/extrae/job_27p.sh
```

job_27p.sh



```
#!/bin/bash
# @ initialdir = .
# @ output = lulesh2_27p.out
# @ error = lulesh2_27p.err
# @ total_tasks = 27
# @ cpus_per_task = 1
# @ wall_clock_limit = 00:10:00



mpirun -np  27 ./lulesh2.0 -i 10 -s 65
```

Request resources
Run the program

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

```
> vi tools-material/extrae/job_27p.sh
```

job_27p.sh

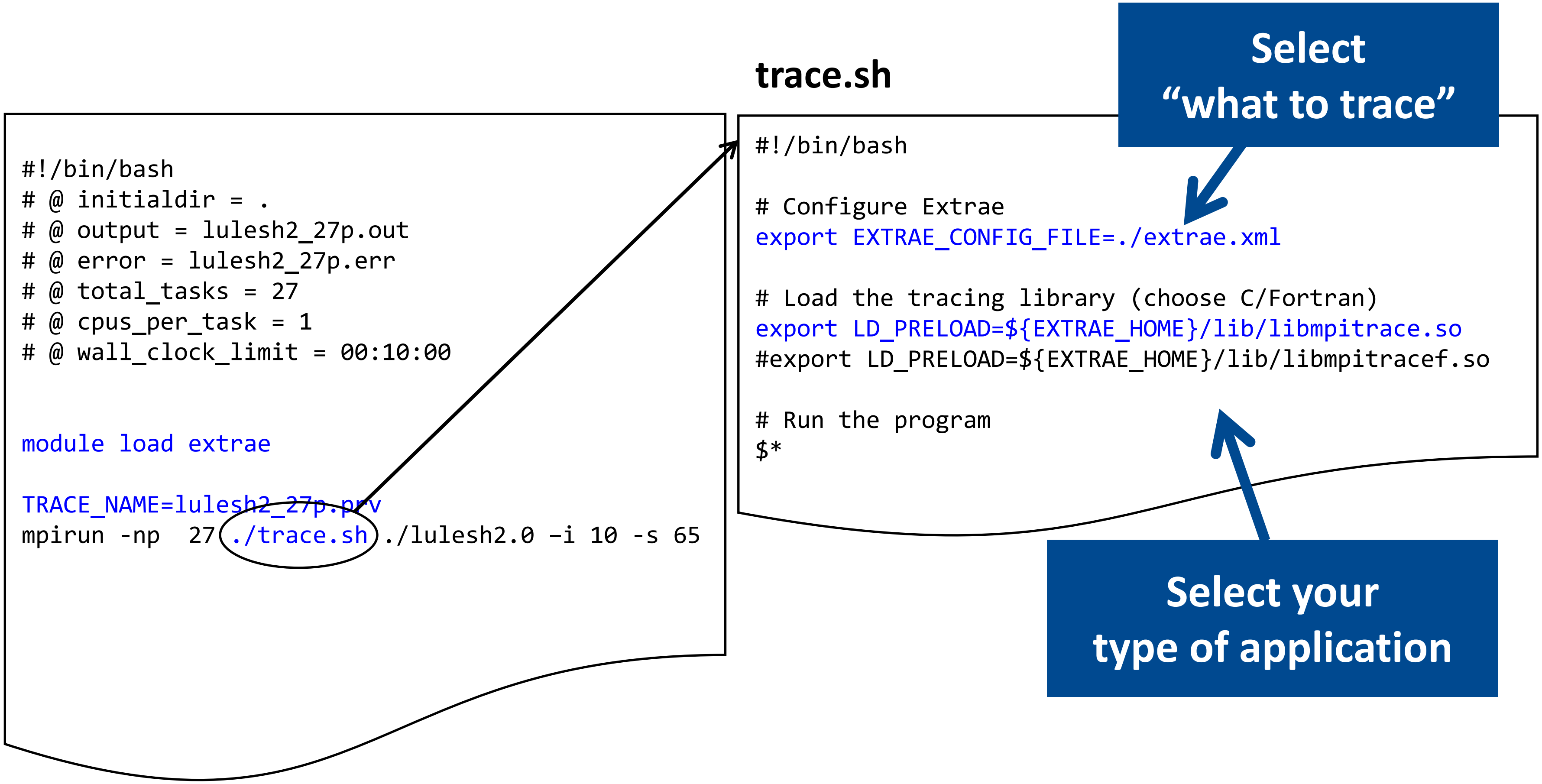
```
#!/bin/bash
# @ initialdir = .
# @ output = lulesh2_27p.out
# @ error = lulesh2_27p.err
# @ total_tasks = 27
# @ cpus_per_task = 1
# @ wall_clock_limit = 00:10:00


module load extrae

TRACE_NAME=lulesh2_27p.prv
mpirun -np  27 ./trace.sh ./lulesh2.0 -i 10 -s 65
```

Load Extrae

Activate Extrae in the execution

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

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



```
#!/bin/bash
# @ initialdir = .
# @ output = lulesh2_27p.out
# @ error = lulesh2_27p.err
# @ total_tasks = 27
# @ cpus_per_task = 1
# @ wall_clock_limit = 00:10:00


module load extrae

TRACE_NAME=lulesh2_27p.prv
mpirun -np  27 ./trace.sh ./lulesh2.0 -i 10 -s 65
```

trace.sh

```
#!/bin/bash

# Configure Extrae
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 “what to trace”

Select your type of application

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

Step 3: Run it!

- Submit your job

```
> cd tools-material/extrae

> qsub job_27p.sh
```

All done! Check your resulting trace

- Once finished you will have the trace (3 files):

```
> ls –l tools-material/extrae
    ...
    lulesh2_27p.pcf
    lulesh2_27p.prv
    lulesh2_27p.row
```

- To proceed with the example traces already generated here:

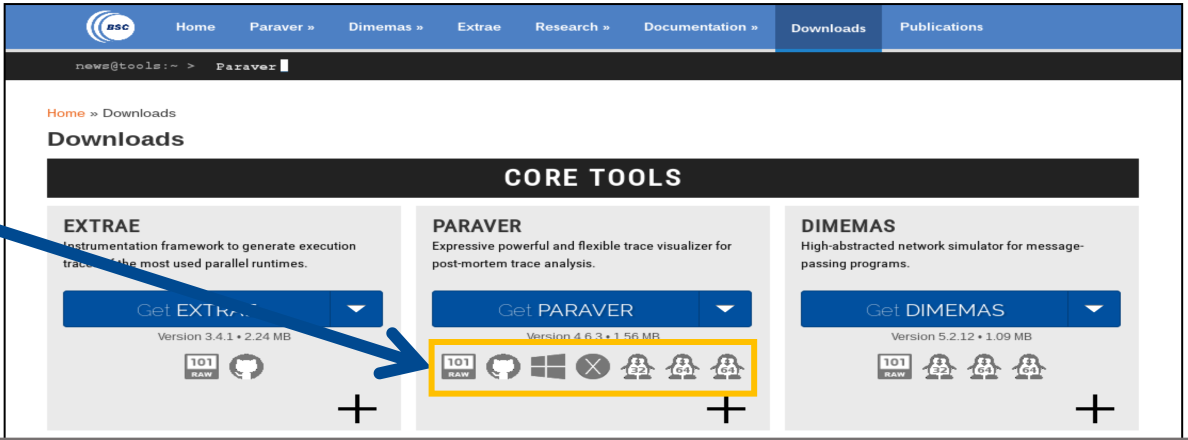
```
> ls tools-material/traces
```

- Now let’s look into it !

Install Paraver

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

Pick your version



```
news@tools:~ > Paraver
```

Home » Downloads

Downloads

CORE TOOLS

EXTRAE

Instrumentation framework to generate execution traces of the most used parallel runtimes.

Version 3.4.1 • 2.24 MB

PARAVER

Expressive powerful and flexible trace visualizer for post-mortem trace analysis.

Version 4.6.3 • 1.56 MB

DIMEMAS

High-abstracted network simulator for message-passing programs.

Version 5.2.12 • 1.09 MB

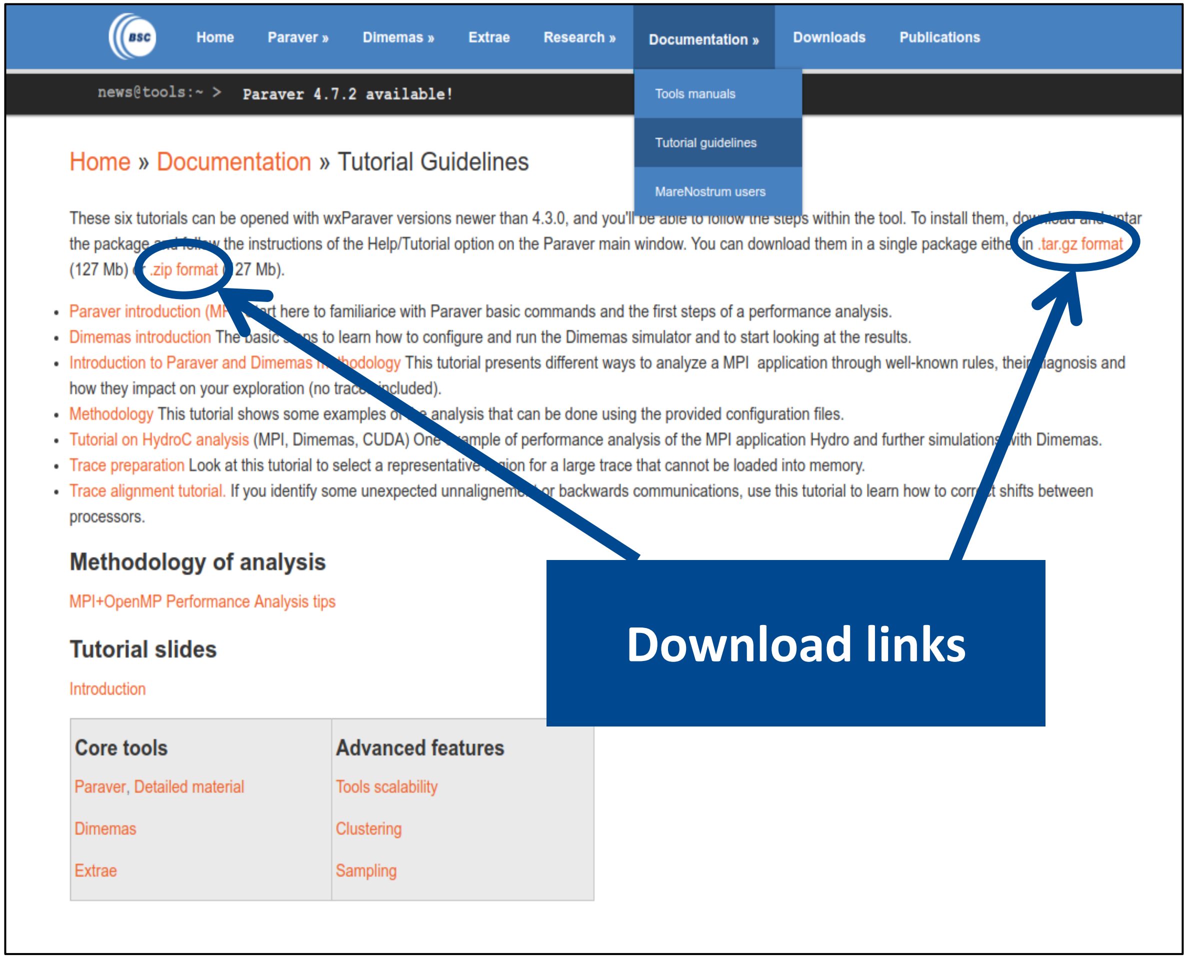
wxparaver-4.7.2-win.zip

wxparaver-4.7.2-mac.zip

wxparaver-4.7.2-Linux_i686.tar.gz (32-bits)
wxparaver-4.7.2-Linux_x86_64.tar.gz (64-bits)

Install Paraver (II)

- Download tutorials:
  - Documentation
    - Tutorial guidelines



Paraver 4.7.2 available!

Home » Documentation » Tutorial Guidelines

These six tutorials can be opened with wxParaver versions newer than 4.3.0, and you'll be able to follow the steps within the tool. To install them, download and untar the package and follow the instructions of the Help/Tutorial option on the Paraver main window. You can download them in a single package either in .tar.gz format (127 Mb) or .zip format (127 Mb).

- Paraver introduction (MPI) Start here to familiarice with Paraver basic commands and the first steps of a performance analysis.
- Dimemas introduction The basic steps to learn how to configure and run the Dimemas simulator and to start looking at the results.
- Introduction to Paraver and Dimemas methodology This tutorial presents different ways to analyze a MPI application through well-known rules, their diagnosis and how they impact on your exploration (no traces included).
- Methodology This tutorial shows some examples of the analysis that can be done using the provided configuration files.
- Tutorial on HydroC analysis (MPI, Dimemas, CUDA) One example of performance analysis of the MPI application Hydro and further simulations with Dimemas.
- Trace preparation Look at this tutorial to select a representative region for a large trace that cannot be loaded into memory.
- Trace alignment tutorial. If you identify some unexpected unnalignements or backwards communications, use this tutorial to learn how to correct shifts between processors.

Methodology of analysis

MPI+OpenMP Performance Analysis tips

Tutorial slides

Introduction

Core tools	Advanced features
Paraver, Detailed material	Tools scalability
Dimemas	Clustering
Extrae	Sampling

Download links

Uncompress, rename & move

- Command-line

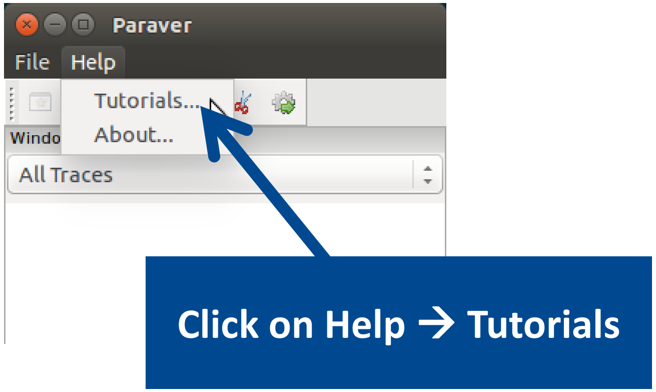
```
> tar xf wxparaver-4.7.2-linux-x86_64.tar.gz
> mv wxparaver-4.6.2-linux-x86_64 paraver
> tar xf paraver-tutorials-20150526.tar.gz
> mv paraver-tutorials-20150526 paraver/tutorials
```

Check that everything works

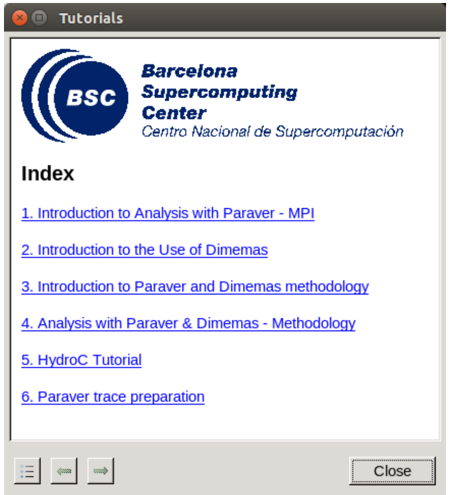
- Start Paraver

```
> paraver/bin/wxparaver
```

- Check that tutorials are available



Click on Help → Tutorials

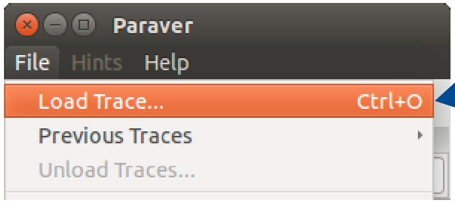

Tutorials
BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Index

1. Introduction to Analysis with Paraver - MPI
2. Introduction to the Use of Dimemas
3. Introduction to Paraver and Dimemas methodology
4. Analysis with Paraver & Dimemas - Methodology
5. HydroC Tutorial
6. Paraver trace preparation

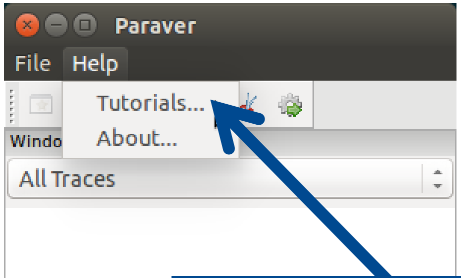
First steps of analysis

- Load the trace with Paraver

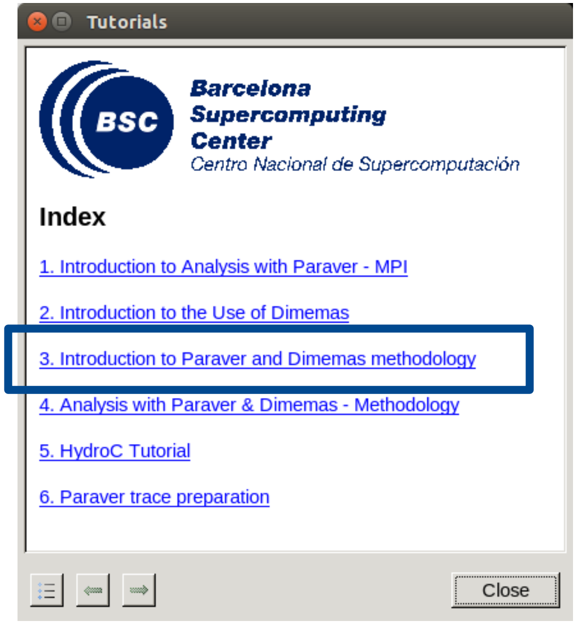


Click on File → Load Trace → Browse to “lulesh2_27p.prv”

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



Click on Help → Tutorials

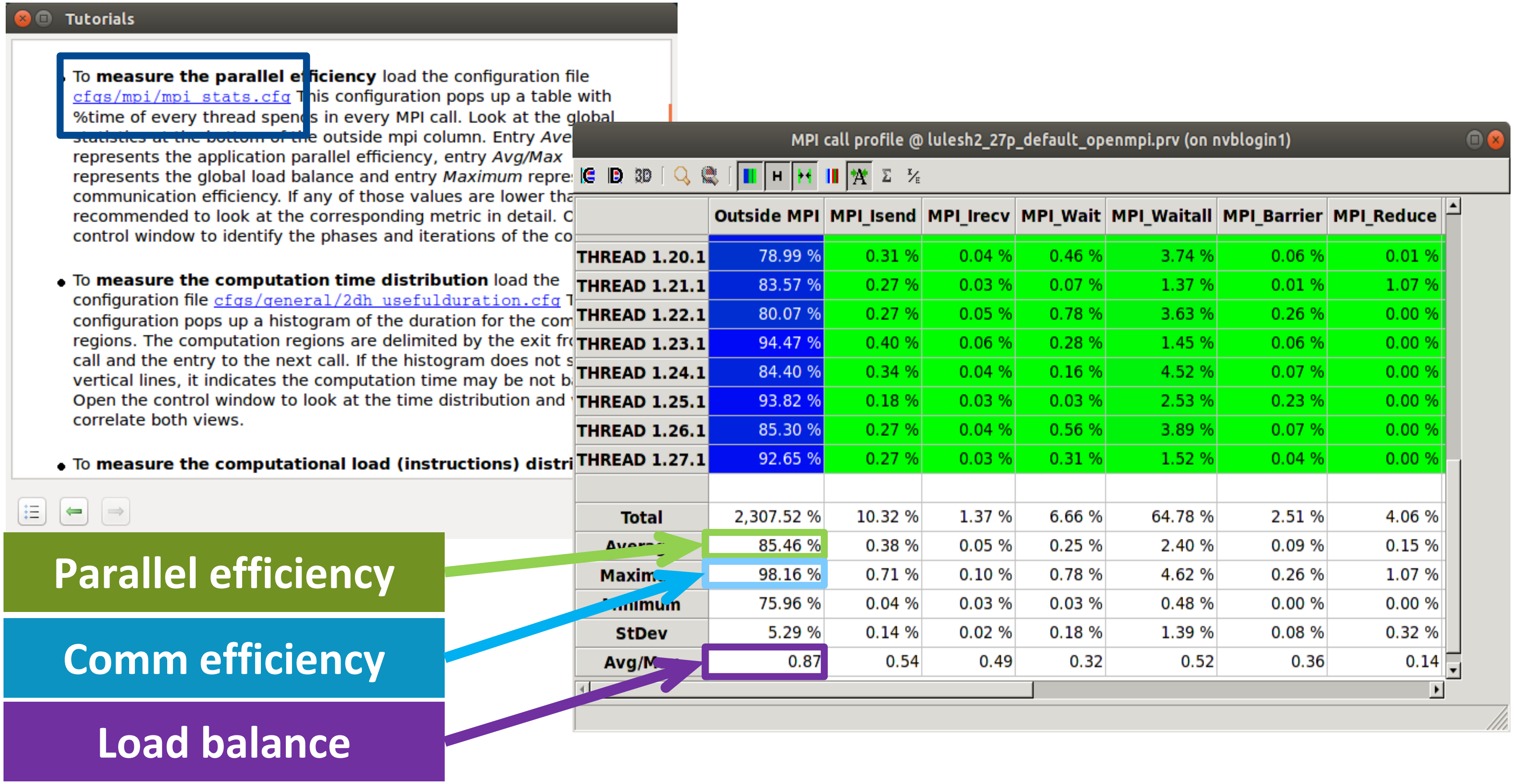

Tutorials
BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
Close

Index

1. Introduction to Analysis with Paraver - MPI
2. Introduction to the Use of Dimemas
3. Introduction to Paraver and Dimemas methodology
4. Analysis with Paraver & Dimemas - Methodology
5. HydroC Tutorial
6. Paraver trace preparation

Measure the parallel efficiency

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



- To **measure the parallel efficiency** load the configuration file cfgs/mpi/mpi_stats.cfg This configuration pops up a table with %time of every thread spends in every MPI call. Look at the global statistics at the bottom of the outside mpi column. Entry *Ave* represents the application parallel efficiency, entry *Avg/Max* represents the global load balance and entry *Maximum* repre communication efficiency. If any of those values are lower tha recommended to look at the corresponding metric in detail. C control window to identify the phases and iterations of the co

- To **measure the computation time distribution** load the configuration file cfgs/general/2dh_usefulduration.cfg T configuration pops up a histogram of the duration for the com regions. The computation regions are delimited by the exit fr call and the entry to the next call. If the histogram does not s vertical lines, it indicates the computation time may be not b Open the control window to look at the time distribution and correlate both views.

- To **measure the computational load (instructions) distri**

MPI call profile @ lulesh2_27p_default_openmpi.prv (on nvblogin1)

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.20.1	78.99 %	0.31 %	0.04 %	0.46 %	3.74 %	0.06 %	0.01 %
THREAD 1.21.1	83.57 %	0.27 %	0.03 %	0.07 %	1.37 %	0.01 %	1.07 %
THREAD 1.22.1	80.07 %	0.27 %	0.05 %	0.78 %	3.63 %	0.26 %	0.00 %
THREAD 1.23.1	94.47 %	0.40 %	0.06 %	0.28 %	1.45 %	0.06 %	0.00 %
THREAD 1.24.1	84.40 %	0.34 %	0.04 %	0.16 %	4.52 %	0.07 %	0.00 %
THREAD 1.25.1	93.82 %	0.18 %	0.03 %	0.03 %	2.53 %	0.23 %	0.00 %
THREAD 1.26.1	85.30 %	0.27 %	0.04 %	0.56 %	3.89 %	0.07 %	0.00 %
THREAD 1.27.1	92.65 %	0.27 %	0.03 %	0.31 %	1.52 %	0.04 %	0.00 %
Total	2,307.52 %	10.32 %	1.37 %	6.66 %	64.78 %	2.51 %	4.06 %
Avera	85.46 %	0.38 %	0.05 %	0.25 %	2.40 %	0.09 %	0.15 %
Maxim	98.16 %	0.71 %	0.10 %	0.78 %	4.62 %	0.26 %	1.07 %
nimum	75.96 %	0.04 %	0.03 %	0.03 %	0.48 %	0.00 %	0.00 %
StDev	5.29 %	0.14 %	0.02 %	0.18 %	1.39 %	0.08 %	0.32 %
Avg/M	0.87	0.54	0.49	0.32	0.52	0.36	0.14

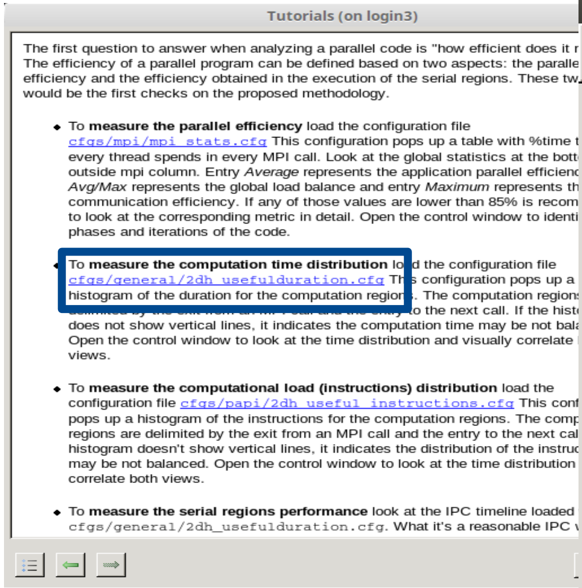
Parallel efficiency

Comm efficiency

Load balance

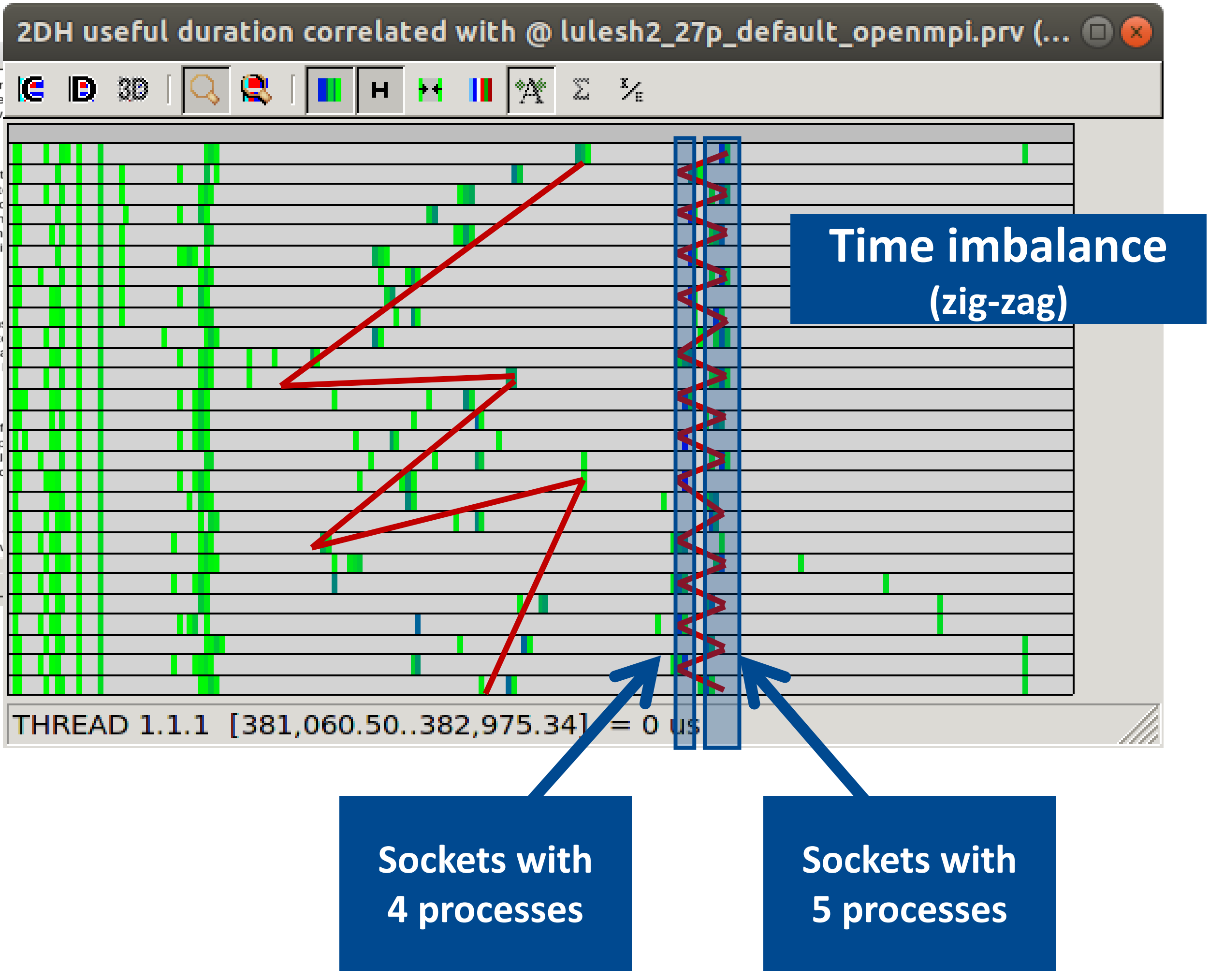
Computation load & time distribution

- Click on “2dh_usefulduration.cfg” (2nd link) ➔ Shows **time computing**



The first question to answer when analyzing a parallel code is "how efficient does it r
The efficiency of a parallel program can be defined based on two aspects: the paralle
efficiency and the efficiency obtained in the execution of the serial regions. These tw
would be the first checks on the proposed methodology.

- To **measure the parallel efficiency** load the configuration file cfgs/mpi/mpi_stats.cfg This configuration pops up a table with %time t every thread spends in every MPI call. Look at the global statistics at the bott outside mpi column. Entry *Average* represents the application parallel efficienc *Avg/Max* represents the global load balance and entry *Maximum* represents th communication efficiency. If any of those values are lower than 85% is recom to look at the corresponding metric in detail. Open the control window to identi phases and iterations of the code.
- To **measure the computation time distribution** load the configuration file cfgs/general/2dh_usefulduration.cfg This configuration pops up a histogram of the duration for the computation regions. The computation regions delimited by the exit from an MPI call and the entry to the next call. If the histo does not show vertical lines, it indicates the computation time may be not bala Open the control window to look at the time distribution and visually correlate views.
- To **measure the computational load (instructions) distribution** load the configuration file cfgs/papi/2dh_useful_instructions.cfg This conf pops up a histogram of the instructions for the computation regions. The comp regions are delimited by the exit from an MPI call and the entry to the next cal histogram doesn't show vertical lines, it indicates the distribution of the instruc may be not balanced. Open the control window to look at the time distribution correlate both views.
- To **measure the serial regions performance** look at the IPC timeline loaded cfgs/general/2dh_usefulduration.cfg. What it's a reasonable IPC v



2DH useful duration correlated with @ lulesh2_27p_default_openmpi.prv (...

THREAD 1.1.1 [381,060.50..382,975.34] = 0 us

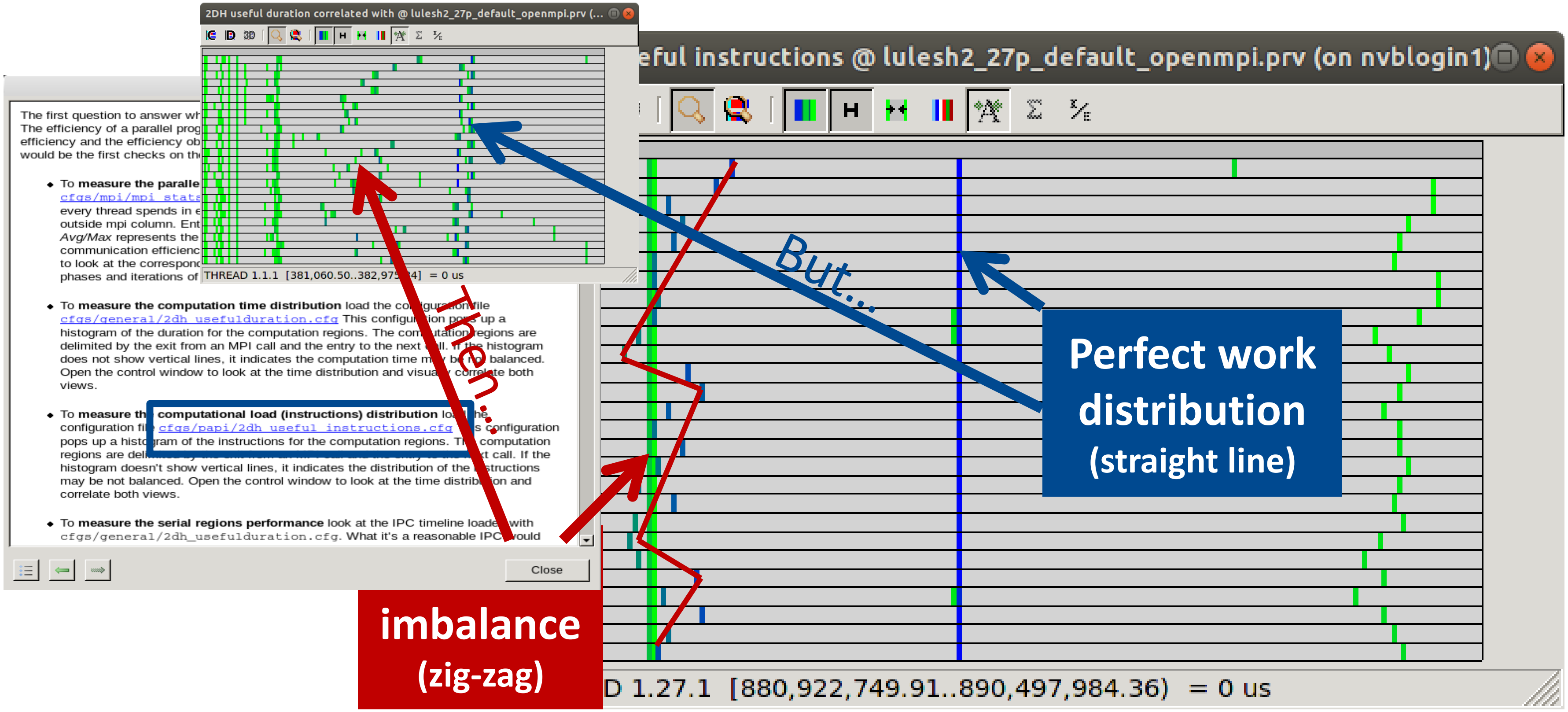
Time imbalance (zig-zag)

Sockets with 4 processes

Sockets with 5 processes

Computation load & time distribution

- Click on “2dh_useful_instructions.cfg” (3rd link) ➔ Shows **amount of work**



2DH useful duration correlated with @ lulesh2_27p_default_openmpi.prv (...
eful instructions @ lulesh2_27p_default_openmpi.prv (on nvblogin1)
Close

The first question to answer wh
The efficiency of a parallel prog
efficiency and the efficiency ob
would be the first checks on the

- To **measure the paralle**
  cfgs/mpi/mpi_stats
  every thread spends in e
  outside mpi column. Ent
  Avg/Max represents the
  communication efficienc
  to look at the correspond
  phases and iterations of

- To **measure the computation time distribution** load the configuration file cfgs/general/2dh_usefulduration.cfg This configuration pops up a histogram of the duration for the computation regions. The computation regions are delimited by the exit from an MPI call and the entry to the next call. If the histogram does not show vertical lines, it indicates the computation time may be not balanced. Open the control window to look at the time distribution and visually correlate both views.

- To **measure the computational load (instructions) distribution** load the configuration file cfgs/papi/2dh_useful_instructions.cfg This configuration pops up a histogram of the instructions for the computation regions. The computation regions are delimited by the exit from an MPI call and the entry to the next call. If the histogram doesn't show vertical lines, it indicates the distribution of the instructions may be not balanced. Open the control window to look at the time distribution and correlate both views.

- To **measure the serial regions performance** look at the IPC timeline loaded with cfgs/general/2dh_usefulduration.cfg. What it's a reasonable IPC would

THREAD 1.1.1 [381,060.50..382,9754] = 0 us

D 1.27.1 [880,922,749.91..890,497,984.36) = 0 us

Then..

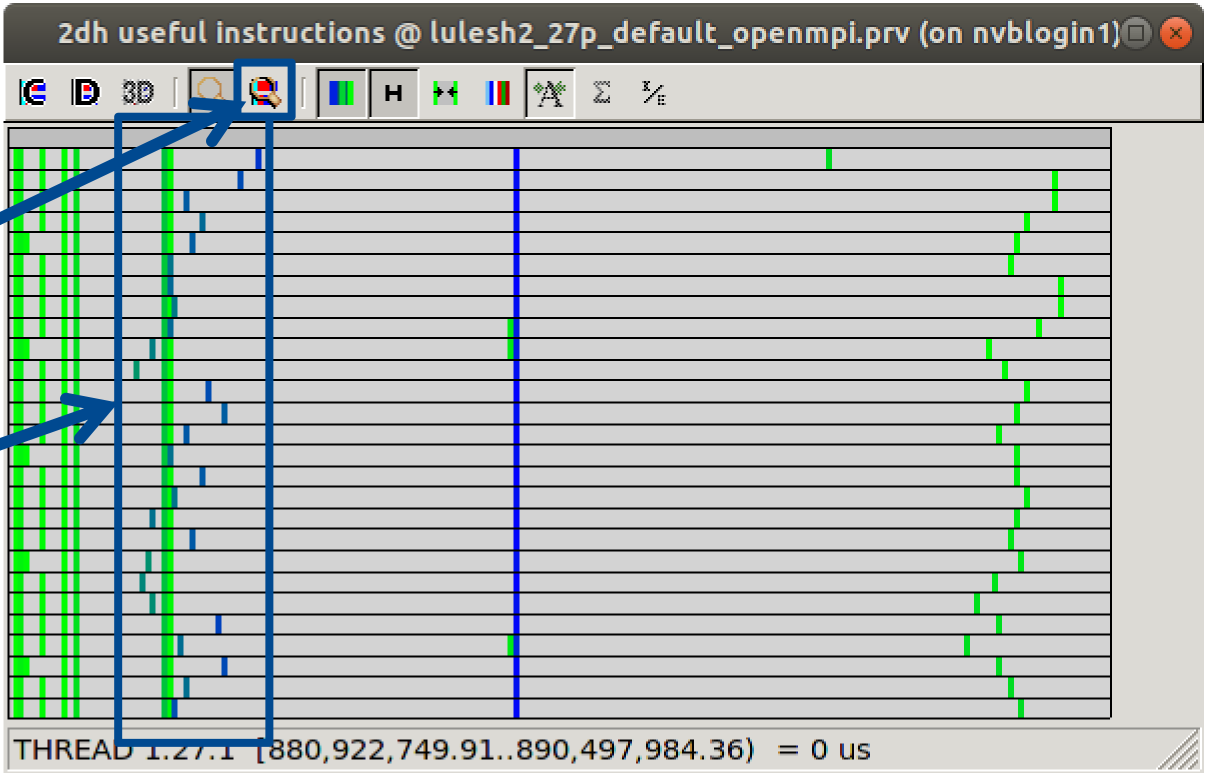
But...

Perfect work distribution (straight line)

imbalance (zig-zag)

Where does this happen?

- Go from the table to the timeline



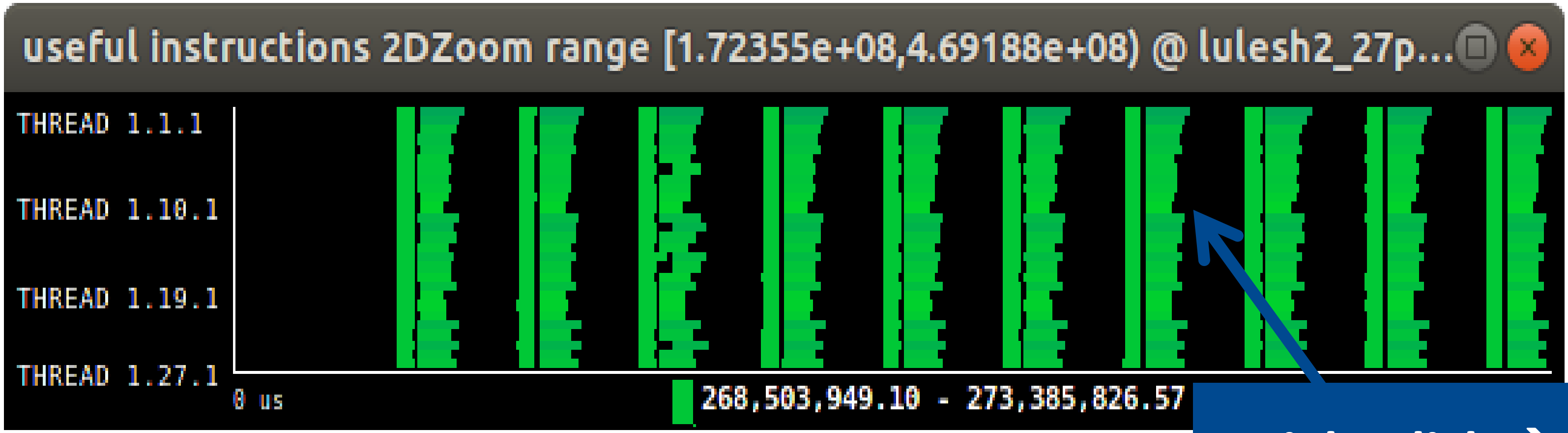
2dh useful instructions @ lulesh2_27p_default_openmpi.prv (on nvblogin1)

THREAD 1.27.1 [880,922,749.91..890,497,984.36) = 0 us

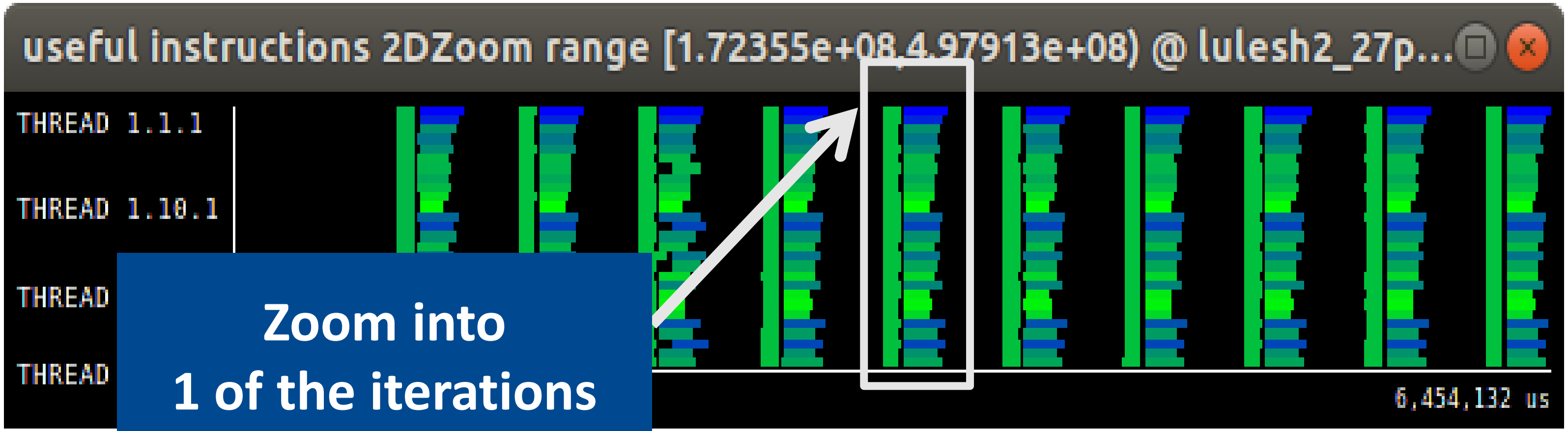
1. Click on "Open Filtered Control Window"

Select this area (by drag-and-dropping)

Where does this happen?



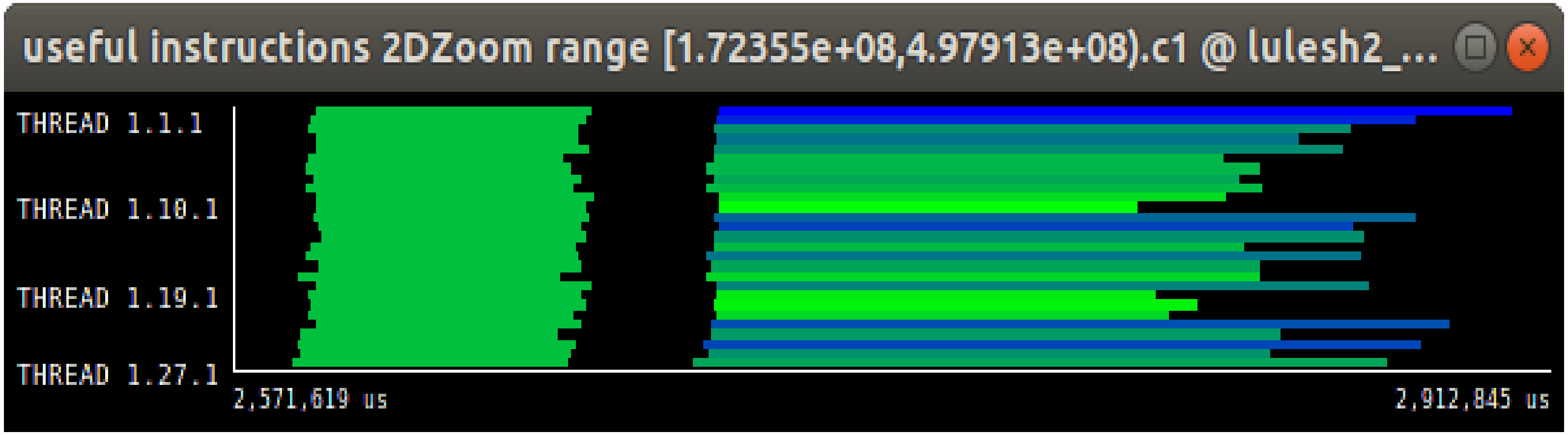
Right click → Fit Semantic Scale → Fit both


useful instructions 2DZoom range [1.72355e+08,4.97913e+08) @ lulesh2_27p...
THREAD 1.1.1
THREAD 1.10.1
THREAD
THREAD
6,454,132 us

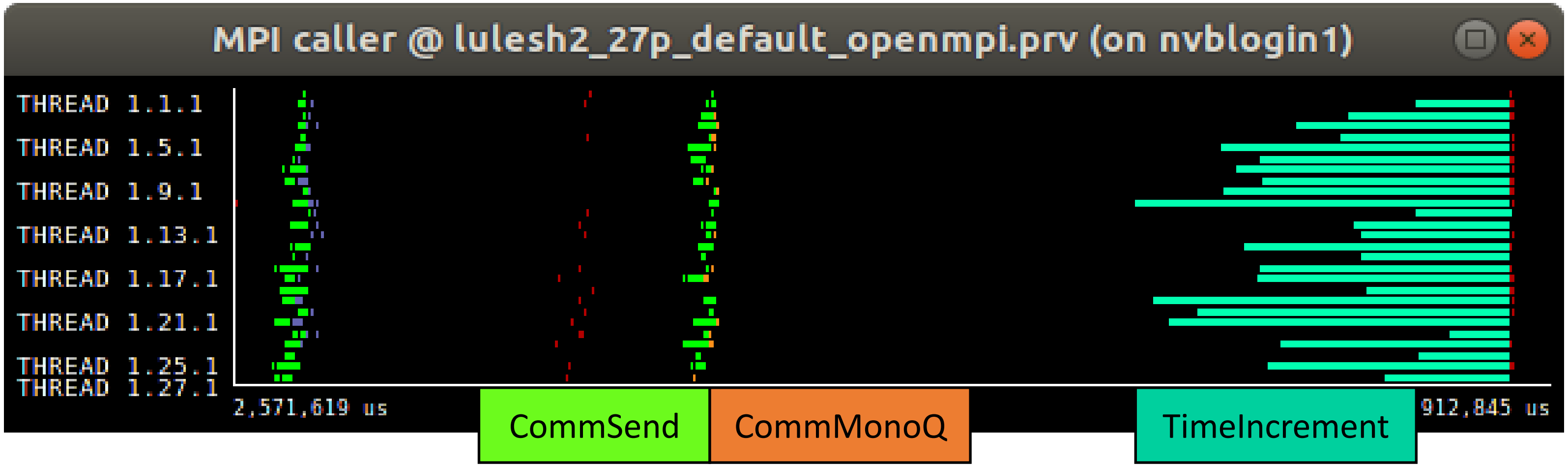
Zoom into 1 of the iterations (by drag-and-dropping)

Where does this happen?

- **Slow** & **Fast** at the same time? ➔ Imbalance


useful instructions 2DZoom range [1.72355e+08,4.97913e+08).c1 @ lulesh2_...
THREAD 1.1.1
THREAD 1.10.1
THREAD 1.19.1
THREAD 1.27.1
2,571,619 us
2,912,845 us

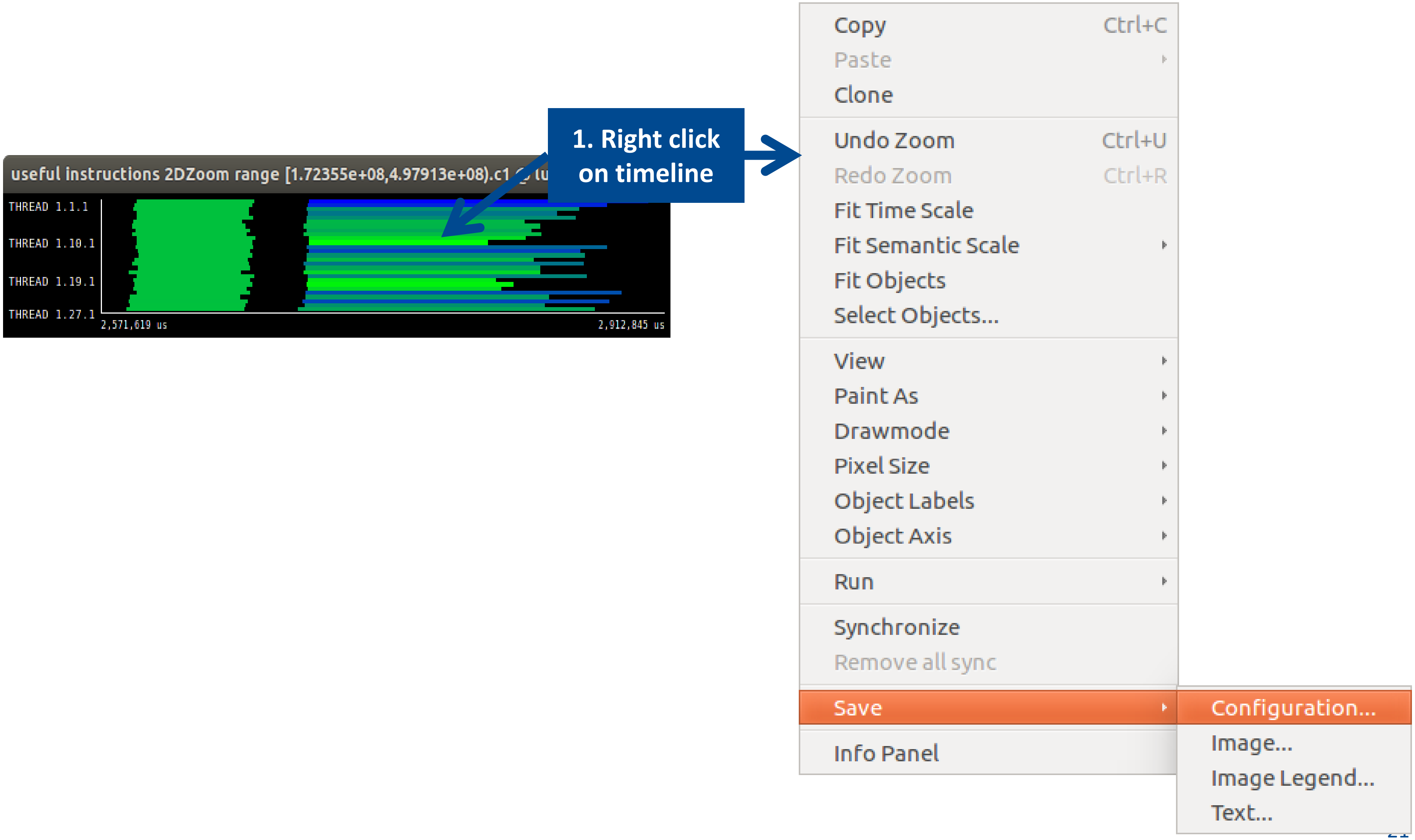
- Reference to the source code: Hints → Callers → Caller function



CommSend CommMonoQ TimeIncrement

Save CFG's (2 methods)

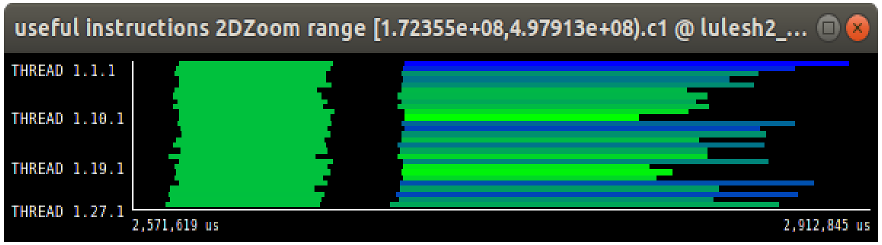
- From the contextual menu



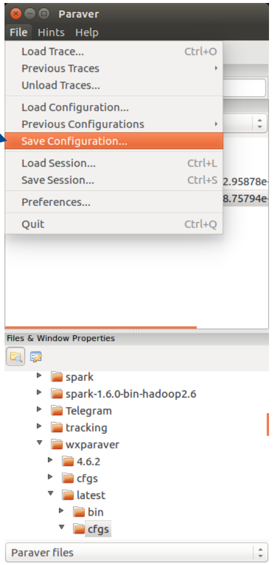
1. Right click on timeline

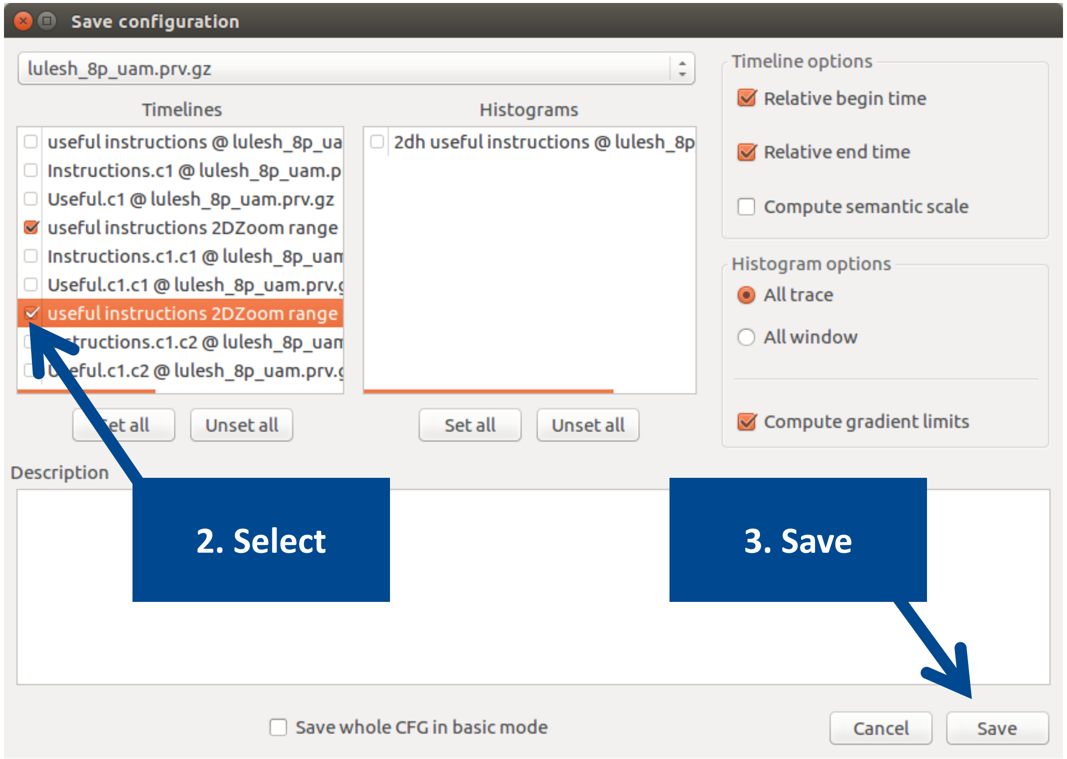
Save CFG's (2 methods)

- From Paraver main window


useful instructions 2DZoom range [1.72355e+08,4.97913e+08).c1 @ lulesh2_...
THREAD 1.1.1
THREAD 1.10.1
THREAD 1.19.1
THREAD 1.27.1
2,571,619 us
2,912,845 us

2. Main Paraver window




Save configuration

lulesh_8p_uam.prv.gz

Timelines
- ☐ useful instructions @ lulesh_8p_ua
- ☐ Instructions.c1 @ lulesh_8p_uam.p
- ☐ Useful.c1 @ lulesh_8p_uam.prv.gz
- ☑ useful instructions 2DZoom range
- ☐ Instructions.c1.c1 @ lulesh_8p_uam
- ☐ Useful.c1.c1 @ lulesh_8p_uam.prv.g
- ☑ useful instructions 2DZoom range
- ☐ Instructions.c1.c2 @ lulesh_8p_uam
- ☐ Useful.c1.c2 @ lulesh_8p_uam.prv.g

Histograms
- ☐ 2dh useful instructions @ lulesh_8p

Timeline options
- ☑ Relative begin time
- ☑ Relative end time
- ☐ Compute semantic scale

Histogram options
- ● All trace
- ○ All window
- ☑ Compute gradient limits

Description

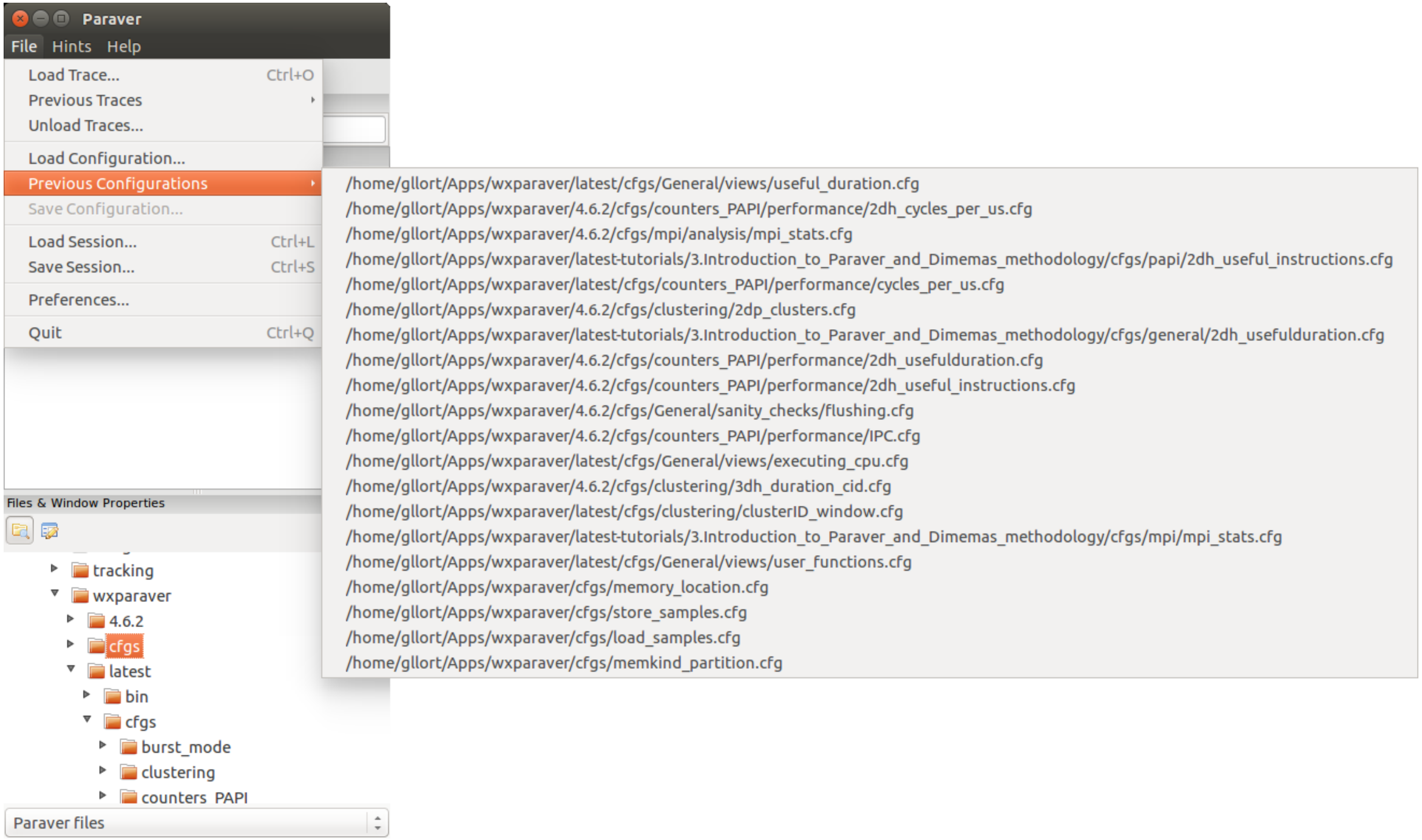
☐ Save whole CFG in basic mode

2. Select

3. Save

CFG's distribution

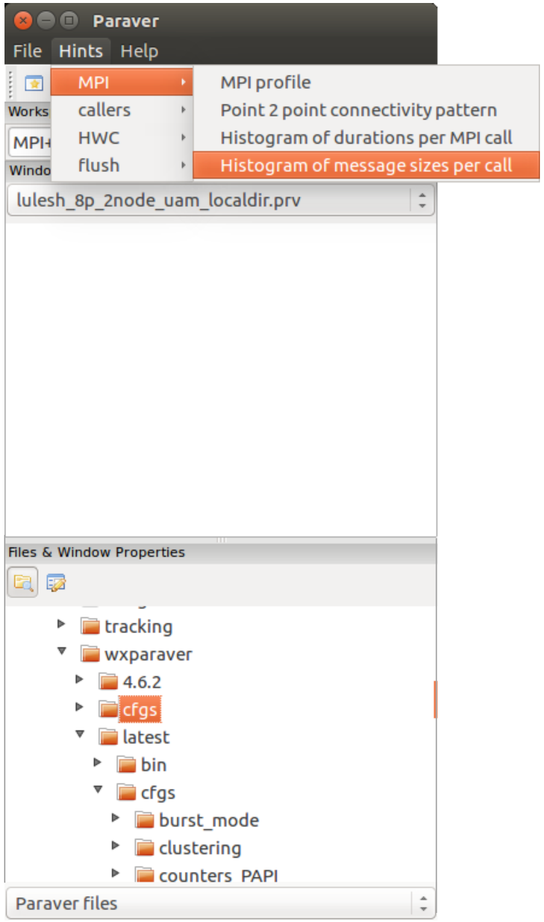
- Paraver comes with many more included CFG's



```
/home/gllort/Apps/wxparaver/latest/cfgs/General/views/useful_duration.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/counters_PAPI/performance/2dh_cycles_per_us.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/mpi/analysis/mpi_stats.cfg
/home/gllort/Apps/wxparaver/latest-tutorials/3.Introduction_to_Paraver_and_Dimemas_methodology/cfgs/papi/2dh_useful_instructions.cfg
/home/gllort/Apps/wxparaver/latest/cfgs/counters_PAPI/performance/cycles_per_us.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/clustering/2dp_clusters.cfg
/home/gllort/Apps/wxparaver/latest-tutorials/3.Introduction_to_Paraver_and_Dimemas_methodology/cfgs/general/2dh_usefulduration.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/counters_PAPI/performance/2dh_usefulduration.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/counters_PAPI/performance/2dh_useful_instructions.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/General/sanity_checks/flushing.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/counters_PAPI/performance/IPC.cfg
/home/gllort/Apps/wxparaver/latest/cfgs/General/views/executing_cpu.cfg
/home/gllort/Apps/wxparaver/4.6.2/cfgs/clustering/3dh_duration_cid.cfg
/home/gllort/Apps/wxparaver/latest/cfgs/clustering/clusterID_window.cfg
/home/gllort/Apps/wxparaver/latest-tutorials/3.Introduction_to_Paraver_and_Dimemas_methodology/cfgs/mpi/mpi_stats.cfg
/home/gllort/Apps/wxparaver/latest/cfgs/General/views/user_functions.cfg
/home/gllort/Apps/wxparaver/cfgs/memory_location.cfg
/home/gllort/Apps/wxparaver/cfgs/store_samples.cfg
/home/gllort/Apps/wxparaver/cfgs/load_samples.cfg
/home/gllort/Apps/wxparaver/cfgs/memkind_partition.cfg
```

Hints: a good place to start!

- Paraver suggests CFG's based on the information present in the trace


Paraver
File
Hints
Help
Works
MPI
Windo
MPI
callers
HWC
flush
MPI profile
Point 2 point connectivity pattern
Histogram of durations per MPI call
Histogram of message sizes per call
lulesh_8p_2node_uam_localdir.prv
Files & Window Properties
tracking
wxparaver
4.6.2
cfgs
latest
bin
cfgs
burst_mode
clustering
counters_PAPI
Paraver files