Barcelona
Supercomputing
Center
Centro Nacional de Supercomputación

EXCELENCIA
SEVERO
OCHOA

BSC Tools Hands-On

Tutorial: Determining Parallel Application Execution Efficiency and Scaling using the POP Methodology

12/05/2024

Sandra Mendez

✉ tools@bsc.es

Using BSC Tools

- Generating the Traces with **Extrae** Tracing Tools ()
- Viewing and Analyzing traces with **Paraver**
- Generating POP Metrics from **Paraver** traces with **BasicAnalysis** Tool
- Analyzing different environments for MPI Applications with **Dimemas** simulator
- Cluster analysis to detect different trends in the application computation regions with **Clustering** Tool

All available in https://tools.bsc.es/downloads

Extrae features

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

No need to recompile nor relink!

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

How does Extrae work?

- Symbol substitution through LD_PRELOAD

```
export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitrace.so
```

  - Specific libraries for each runtime and combinations
    - MPI
    - OpenMP
    - OpenMP+MPI
    - …

Recommended

- 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** your job submission scripts

2. **Configure** what to trace
   - 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/doc/html/extrae
  - Also distributed with Extrae at `$EXTRAE_HOME/share/doc`

Traces from Extrae

- You will have the trace (3 files):

```
mn$ ls -l $HOME/My_Folder/extrae
   ...
   lulesh2.0_i_27p.pcf
   lulesh2.0_i_27p.prv
   lulesh2.0_i_27p.row
```

- Analyzing the trace with Paraver! First we will install it!

Install Paraver

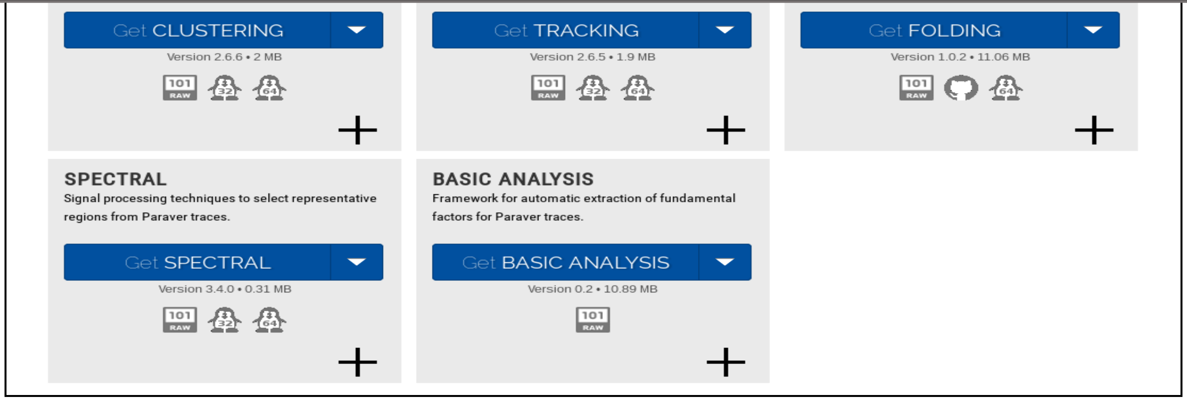
- Download from https://tools.bsc.es/downloads

Pick your version

wxparaver-4.11.4-win.zip

wxparaver-4.11.4-mac.zip

wxparaver-4.11.4-Linux_x86_64.tar.gz (64-bits)



SPECTRAL

Signal processing techniques to select representative regions from Paraver traces.

BASIC ANALYSIS

Framework for automatic extraction of fundamental factors for Paraver traces.

Install Paraver (II)

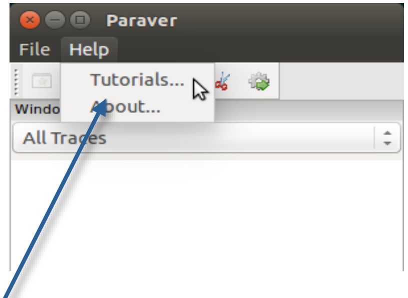
```
laptop$ tar xf wxparaver-4.11.4-linux-x86_64.tar.gz

laptop$ mv wxparaver-4.11.4-linux-x86_64 paraver
```

- Start Paraver

```
laptop$ paraver/bin/wxparaver
```

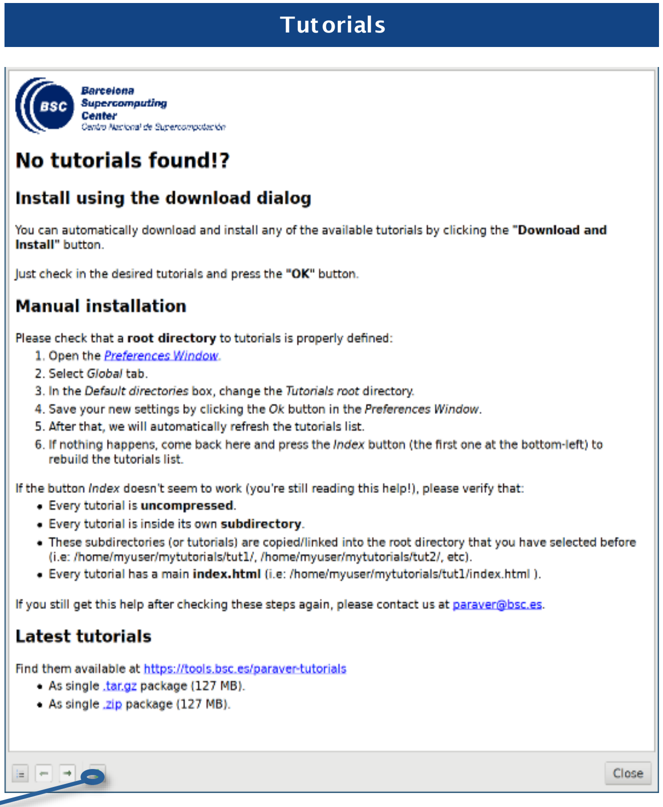
Install Paraver tutorials (I)

- Download tutorials:



Click on Help → Tutorials

Tutorials window will pop-up


Tutorials
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
Close

No tutorials found!?

Install using the download dialog

You can automatically download and install any of the available tutorials by clicking the **"Download and Install"** button.

Just check in the desired tutorials and press the **"OK"** button.

Manual installation

Please check that a **root directory** to tutorials is properly defined:

1. Open the *Preferences Window*.
2. Select *Global* tab.
3. In the *Default directories* box, change the *Tutorials root* directory.
4. Save your new settings by clicking the *Ok* button in the *Preferences Window*.
5. After that, we will automatically refresh the tutorials list.
6. If nothing happens, come back here and press the *Index* button (the first one at the bottom-left) to rebuild the tutorials list.

If the button *Index* doesn't seem to work (you're still reading this help!), please verify that:

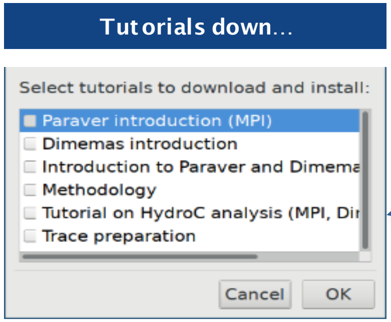
- Every tutorial is **uncompressed**.
- Every tutorial is inside its own **subdirectory**.
- These subdirectories (or tutorials) are copied/linked into the root directory that you have selected before (i.e: /home/myuser/mytutorials/tut1/, /home/myuser/mytutorials/tut2/, etc).
- Every tutorial has a main **index.html** (i.e: /home/myuser/mytutorials/tut1/index.html).

If you still get this help after checking these steps again, please contact us at paraver@bsc.es.

Latest tutorials

Find them available at https://tools.bsc.es/paraver-tutorials

- As single .tar.gz package (127 MB).
- As single .zip package (127 MB).



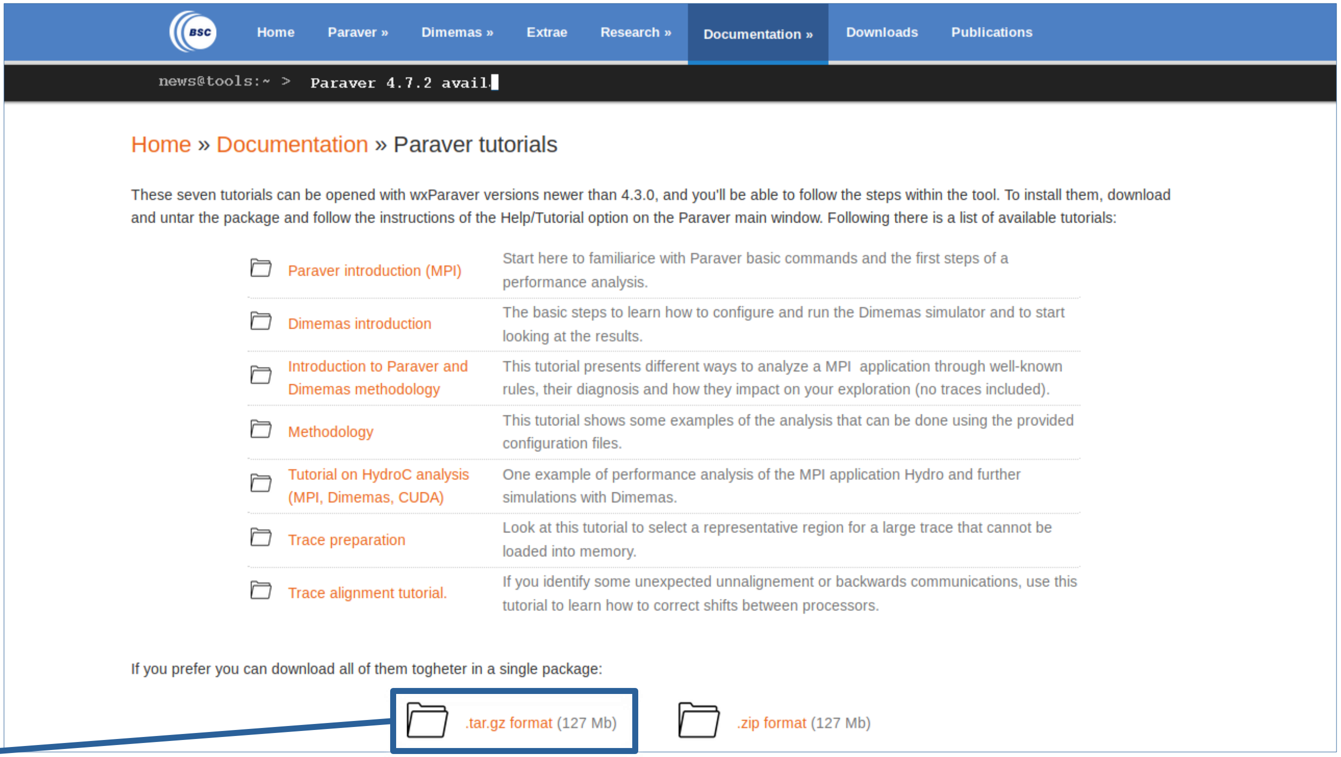
Select tutorials to download and install:

- □ Paraver introduction (MPI)
- □ Dimemas introduction
- □ Introduction to Paraver and Dimema
- □ Methodology
- □ Tutorial on HydroC analysis (MPI, Di
- □ Trace preparation

Follow these tutorials by clicking on the hyperlinks and reading the explanations. When you click on a link, multiple views will open.

Install Paraver tutorials – alternative methods(II)

- Download tutorials archive
  - https://tools.bsc.es/paraver-tutorials



Home » Documentation » Paraver tutorials

These seven 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. Following there is a list of available tutorials:

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 unnalignement or backwards communications, use this tutorial to learn how to correct shifts between processors.

If you prefer you can download all of them togheter in a single package:

.tar.gz format (127 Mb)   .zip format (127 Mb)

All tutorials

```
paraver-tutorials-20150526.tar.gz
```

Install Paraver tutorials – alternative methods(III)

- Start Paraver:
  - Linux: Run the command:

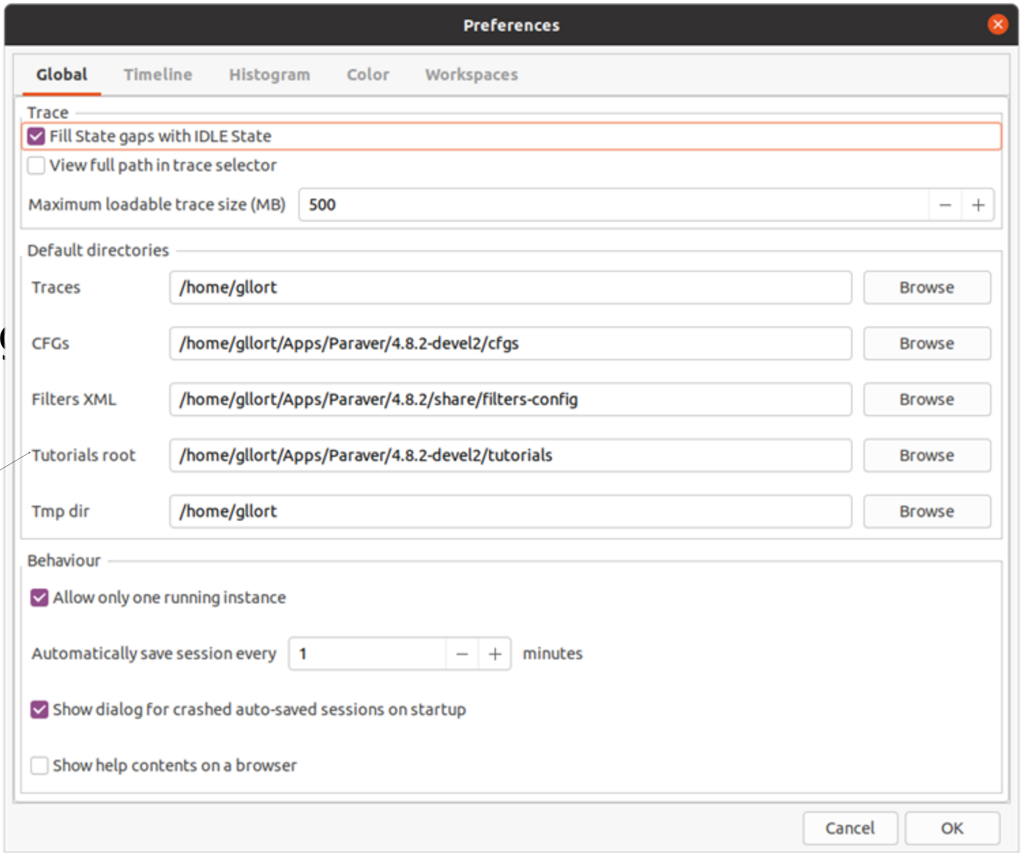
```
laptop$ paraver/bin/wxparaver
```

  - Windows: Double-click on paraver/wxparaver.exe
  - MAC: Double click on paraver/wxparaver.app

- Open File → Preferences

Setup the "Tutorials root" pointin to your folder "tutorials"

Click Browse and select your folder "tutorials"



Preferences

Trace
☑ Fill State gaps with IDLE State
☐ View full path in trace selector
Maximum loadable trace size (MB) 500

Default directories

Traces	/home/gllort	Browse
CFGs	/home/gllort/Apps/Paraver/4.8.2-devel2/cfgs	Browse
Filters XML	/home/gllort/Apps/Paraver/4.8.2/share/filters-config	Browse
Tutorials root	/home/gllort/Apps/Paraver/4.8.2-devel2/tutorials	Browse
Tmp dir	/home/gllort	Browse

Behaviour
☑ Allow only one running instance
Automatically save session every 1 minutes
☑ Show dialog for crashed auto-saved sessions on startup
☐ Show help contents on a browser

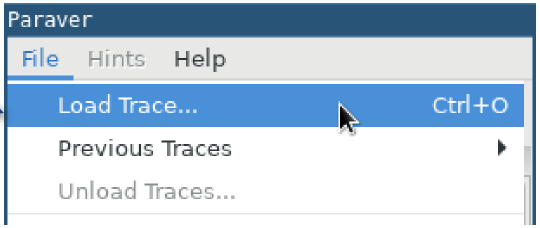
First steps of analysis

- Copy the trace to your laptop

```
laptop$ cp tools-material/extrae/lulesh2.0_i_27p.* $HOME
```

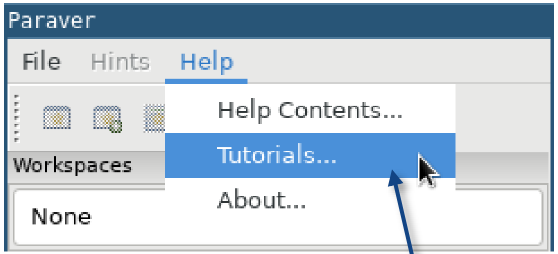
- Load the trace with Paraver

Click on File → Load Trace
⑦ Browse to "lulesh2.0_i_27p.prv"

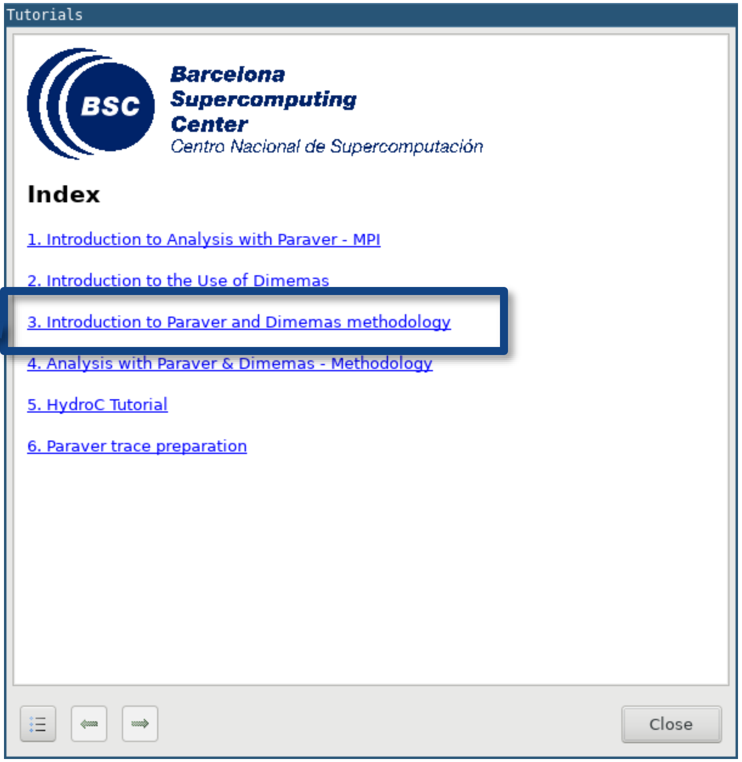


First steps of analysis

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



Click on Help → Tutorials

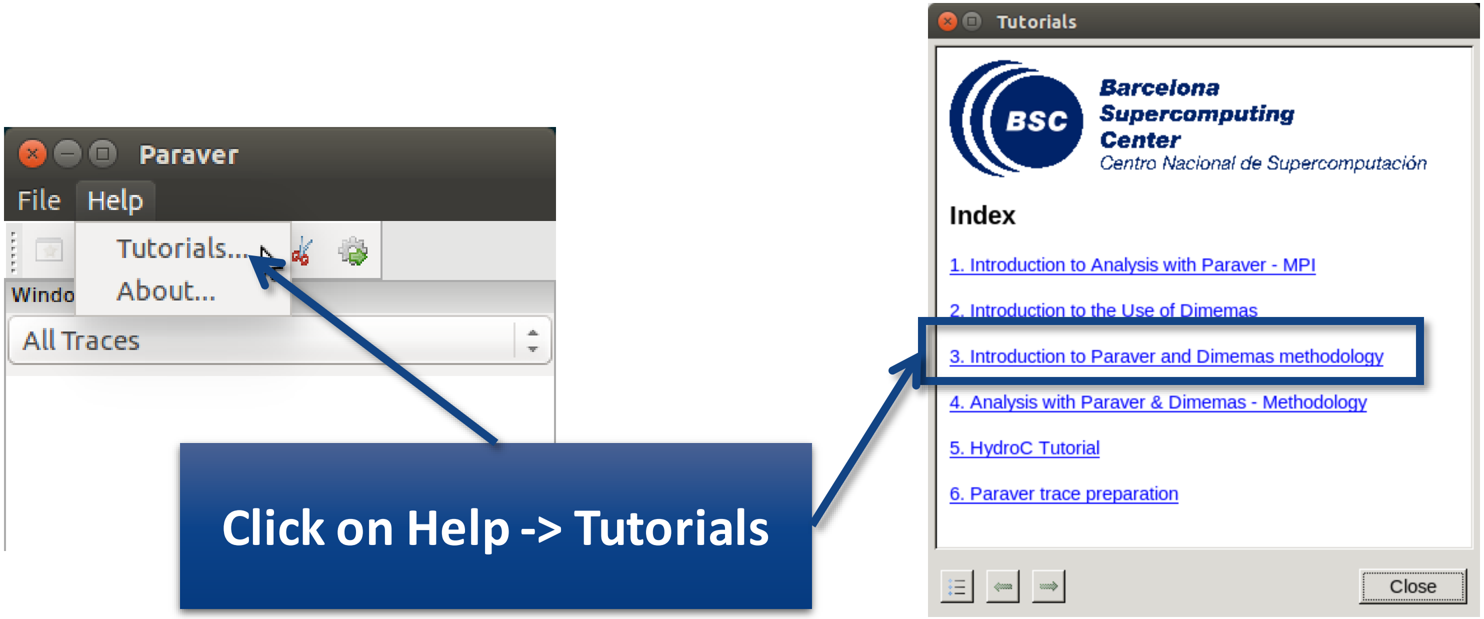

Tutorials
BSC
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
Close

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

First steps of analysis

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


Paraver
File
Help
Tutorials...
About...
Windo
All Traces
Tutorials
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
Close

Click on Help -> Tutorials

Index

1. Introduction to Analysis with Paraver - MPI

2. Introduction to the Use of Dimemas

3. Introduction to Paraver and Dimemas methodology

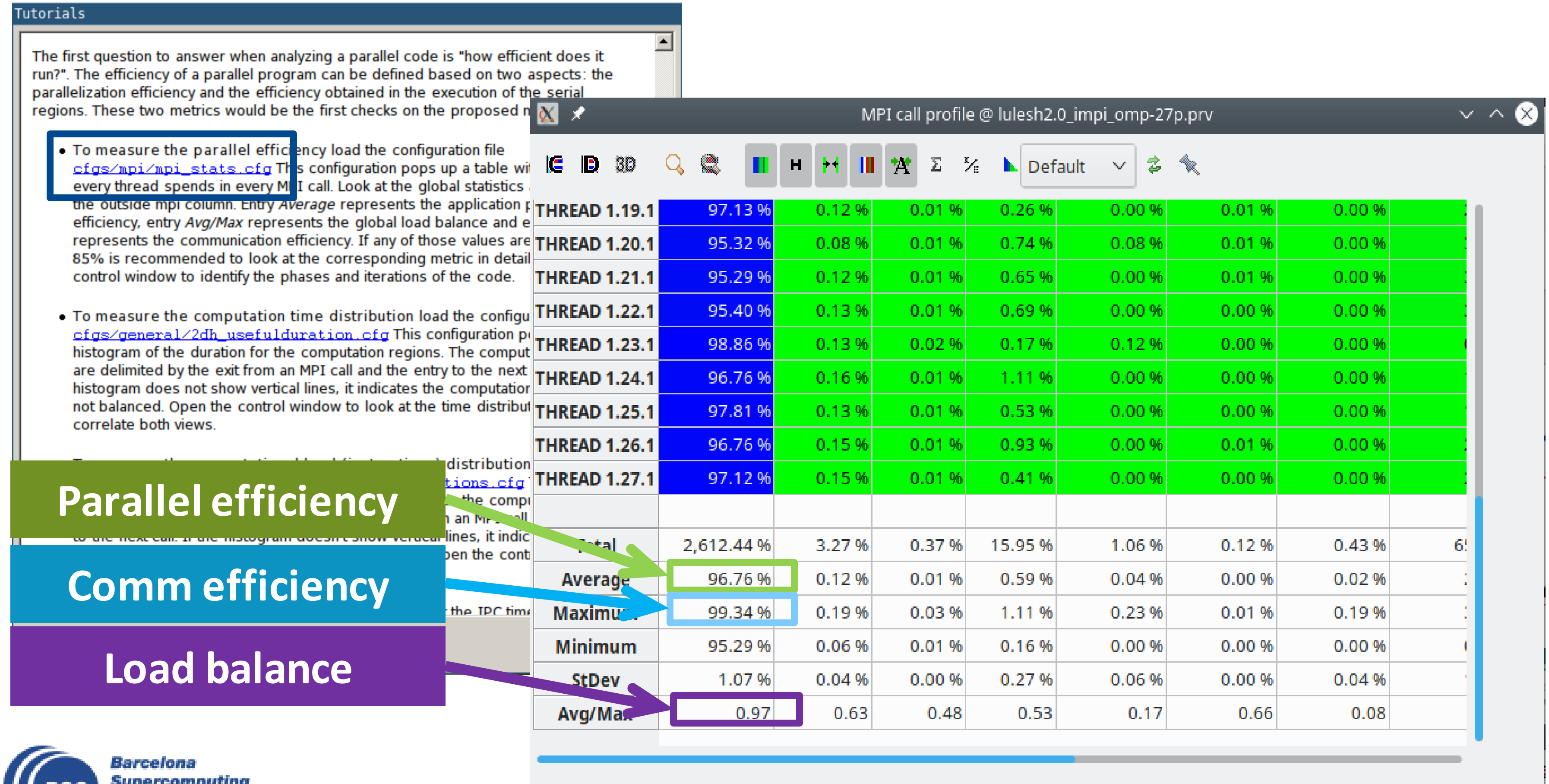
4. Analysis with Paraver & Dimemas - Methodology

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**”



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

- To measure the parallel efficiency load the configuration file cfgs/mpi/mpi_stats.cfg This configuration pops up a table wi every thread spends in every MPI call. Look at the global statistics the outside mpi column. Entry *Average* represents the application p efficiency, entry *Avg/Max* represents the global load balance and e represents the communication efficiency. If any of those values are 85% is recommended to look at the corresponding metric in detail control window to identify the phases and iterations of the code.
- To measure the computation time distribution load the configu cfgs/general/2dh_usefulduration.cfg This configuration p histogram of the duration for the computation regions. The comput are delimited by the exit from an MPI call and the entry to the next histogram does not show vertical lines, it indicates the computatior not balanced. Open the control window to look at the time distribut correlate both views.

distribution tions.cfg the compu n an MPI call lines, it indic pen the cont the IPC time

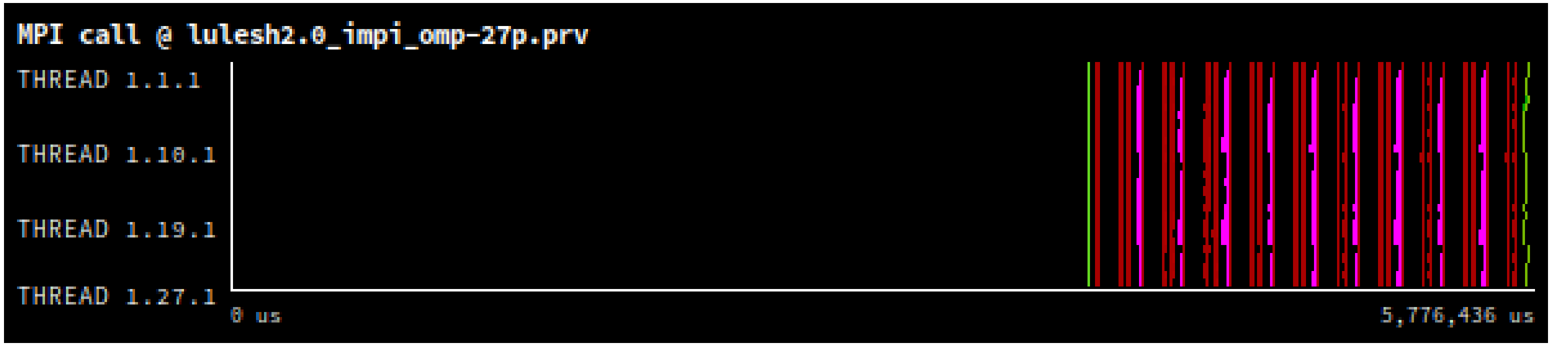
THREAD 1.19.1	97.13 %	0.12 %	0.01 %	0.26 %	0.00 %	0.01 %	0.00 %
THREAD 1.20.1	95.32 %	0.08 %	0.01 %	0.74 %	0.08 %	0.01 %	0.00 %
THREAD 1.21.1	95.29 %	0.12 %	0.01 %	0.65 %	0.00 %	0.01 %	0.00 %
THREAD 1.22.1	95.40 %	0.13 %	0.01 %	0.69 %	0.00 %	0.00 %	0.00 %
THREAD 1.23.1	98.86 %	0.13 %	0.02 %	0.17 %	0.12 %	0.00 %	0.00 %
THREAD 1.24.1	96.76 %	0.16 %	0.01 %	1.11 %	0.00 %	0.00 %	0.00 %
THREAD 1.25.1	97.81 %	0.13 %	0.01 %	0.53 %	0.00 %	0.00 %	0.00 %
THREAD 1.26.1	96.76 %	0.15 %	0.01 %	0.93 %	0.00 %	0.01 %	0.00 %
THREAD 1.27.1	97.12 %	0.15 %	0.01 %	0.41 %	0.00 %	0.00 %	0.00 %
Total	2,612.44 %	3.27 %	0.37 %	15.95 %	1.06 %	0.12 %	0.43 %
Average	96.76 %	0.12 %	0.01 %	0.59 %	0.04 %	0.00 %	0.02 %
Maximum	99.34 %	0.19 %	0.03 %	1.11 %	0.23 %	0.01 %	0.19 %
Minimum	95.29 %	0.06 %	0.01 %	0.16 %	0.00 %	0.00 %	0.00 %
StDev	1.07 %	0.04 %	0.00 %	0.27 %	0.06 %	0.00 %	0.04 %
Avg/Max	0.97	0.63	0.48	0.53	0.17	0.66	0.08

Parallel efficiency

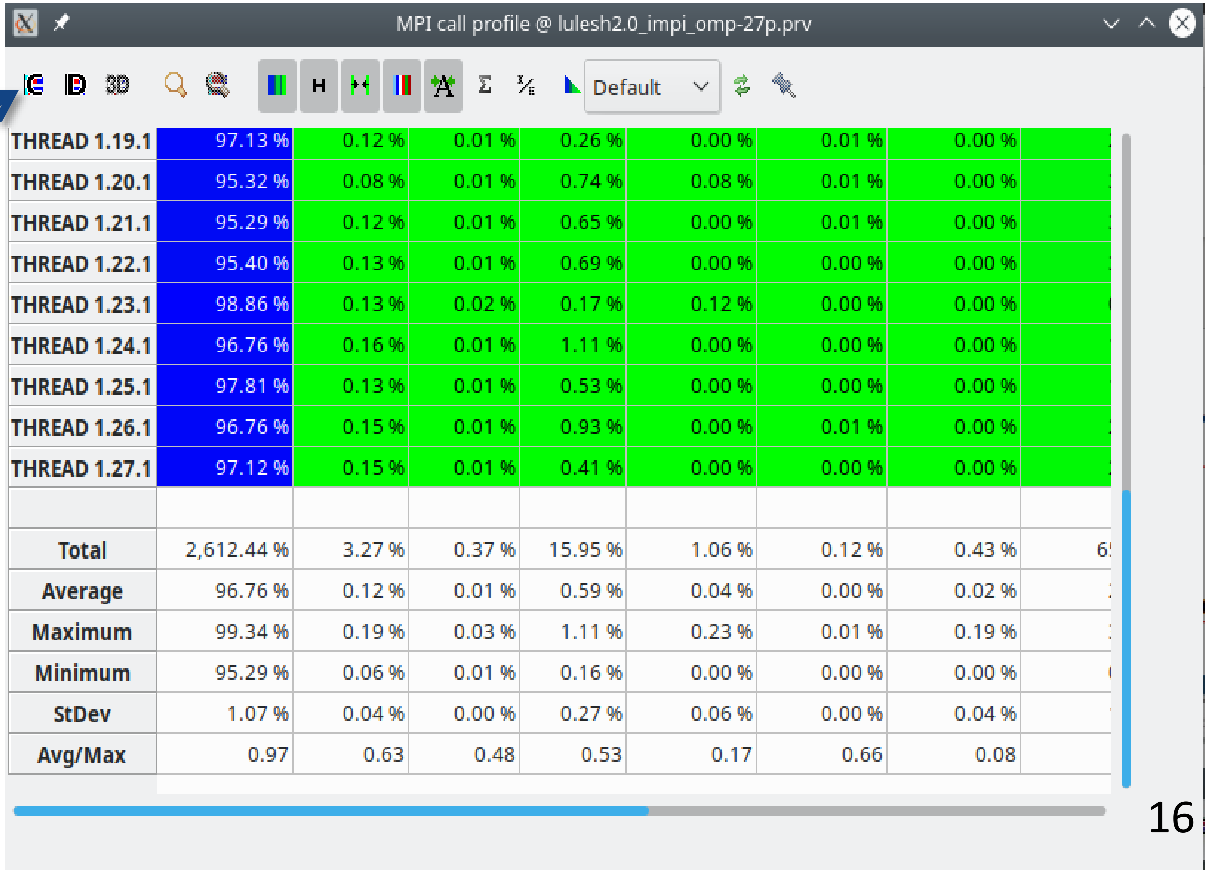
Comm efficiency

Load balance

Focus on the iterative part



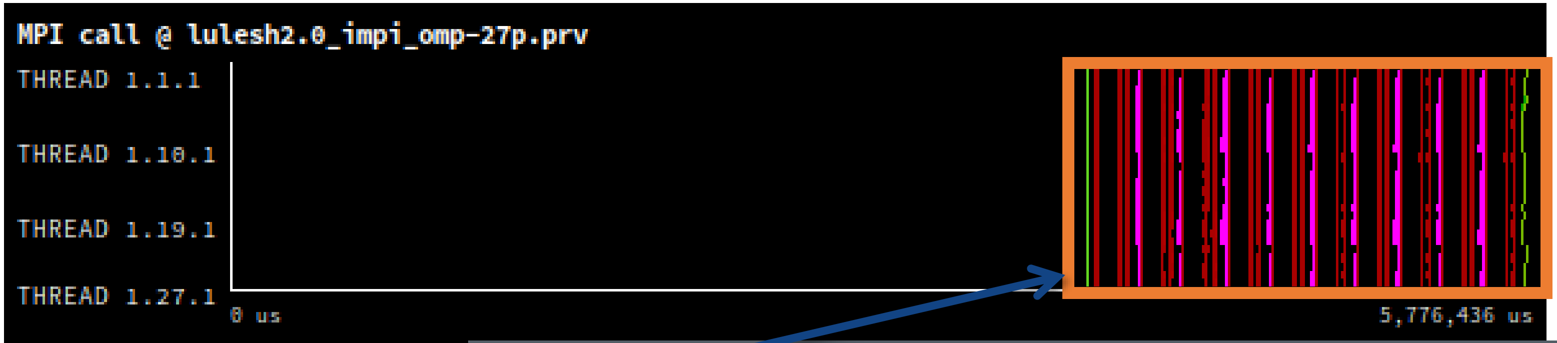
Click on Open Control Window

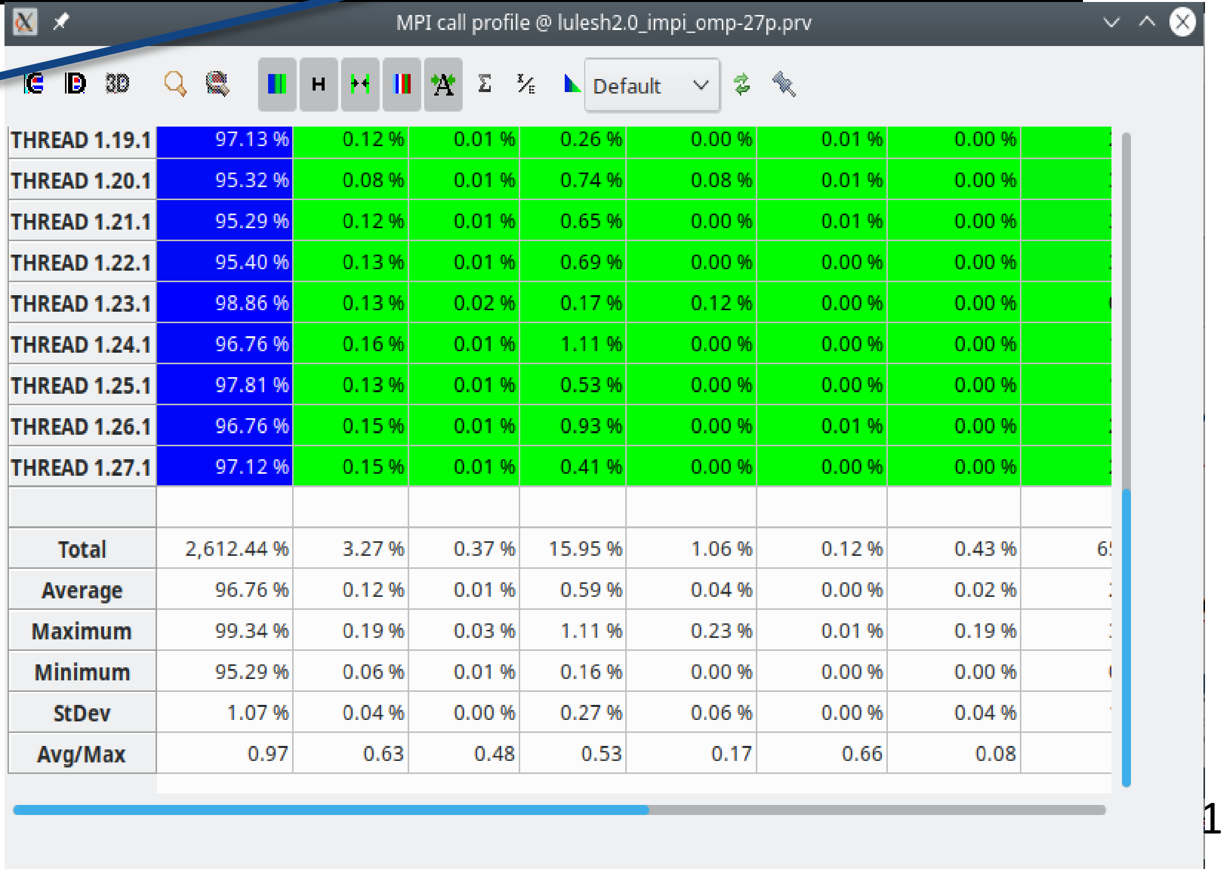


THREAD 1.19.1	97.13 %	0.12 %	0.01 %	0.26 %	0.00 %	0.01 %	0.00 %
THREAD 1.20.1	95.32 %	0.08 %	0.01 %	0.74 %	0.08 %	0.01 %	0.00 %
THREAD 1.21.1	95.29 %	0.12 %	0.01 %	0.65 %	0.00 %	0.01 %	0.00 %
THREAD 1.22.1	95.40 %	0.13 %	0.01 %	0.69 %	0.00 %	0.00 %	0.00 %
THREAD 1.23.1	98.86 %	0.13 %	0.02 %	0.17 %	0.12 %	0.00 %	0.00 %
THREAD 1.24.1	96.76 %	0.16 %	0.01 %	1.11 %	0.00 %	0.00 %	0.00 %
THREAD 1.25.1	97.81 %	0.13 %	0.01 %	0.53 %	0.00 %	0.00 %	0.00 %
THREAD 1.26.1	96.76 %	0.15 %	0.01 %	0.93 %	0.00 %	0.01 %	0.00 %
THREAD 1.27.1	97.12 %	0.15 %	0.01 %	0.41 %	0.00 %	0.00 %	0.00 %
Total	2,612.44 %	3.27 %	0.37 %	15.95 %	1.06 %	0.12 %	0.43 %
Average	96.76 %	0.12 %	0.01 %	0.59 %	0.04 %	0.00 %	0.02 %
Maximum	99.34 %	0.19 %	0.03 %	1.11 %	0.23 %	0.01 %	0.19 %
Minimum	95.29 %	0.06 %	0.01 %	0.16 %	0.00 %	0.00 %	0.00 %
StDev	1.07 %	0.04 %	0.00 %	0.27 %	0.06 %	0.00 %	0.04 %
Avg/Max	0.97	0.63	0.48	0.53	0.17	0.66	0.08

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Focus on the iterative part

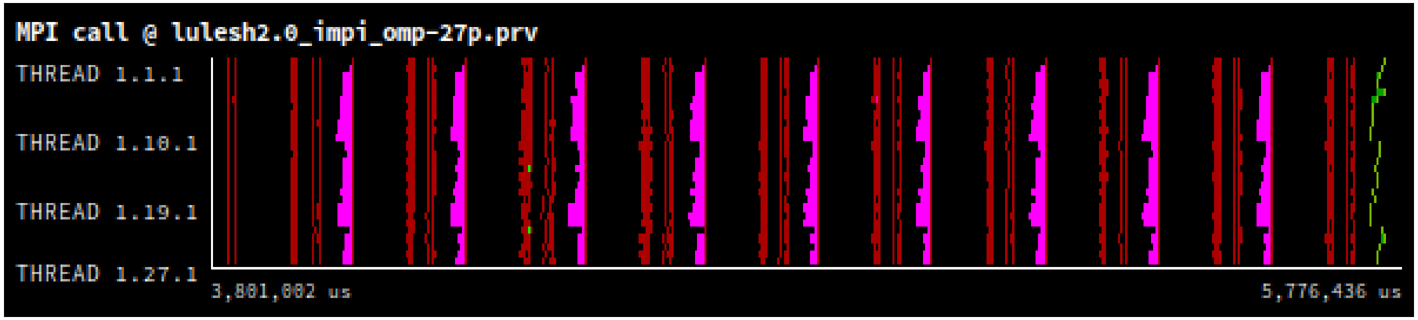




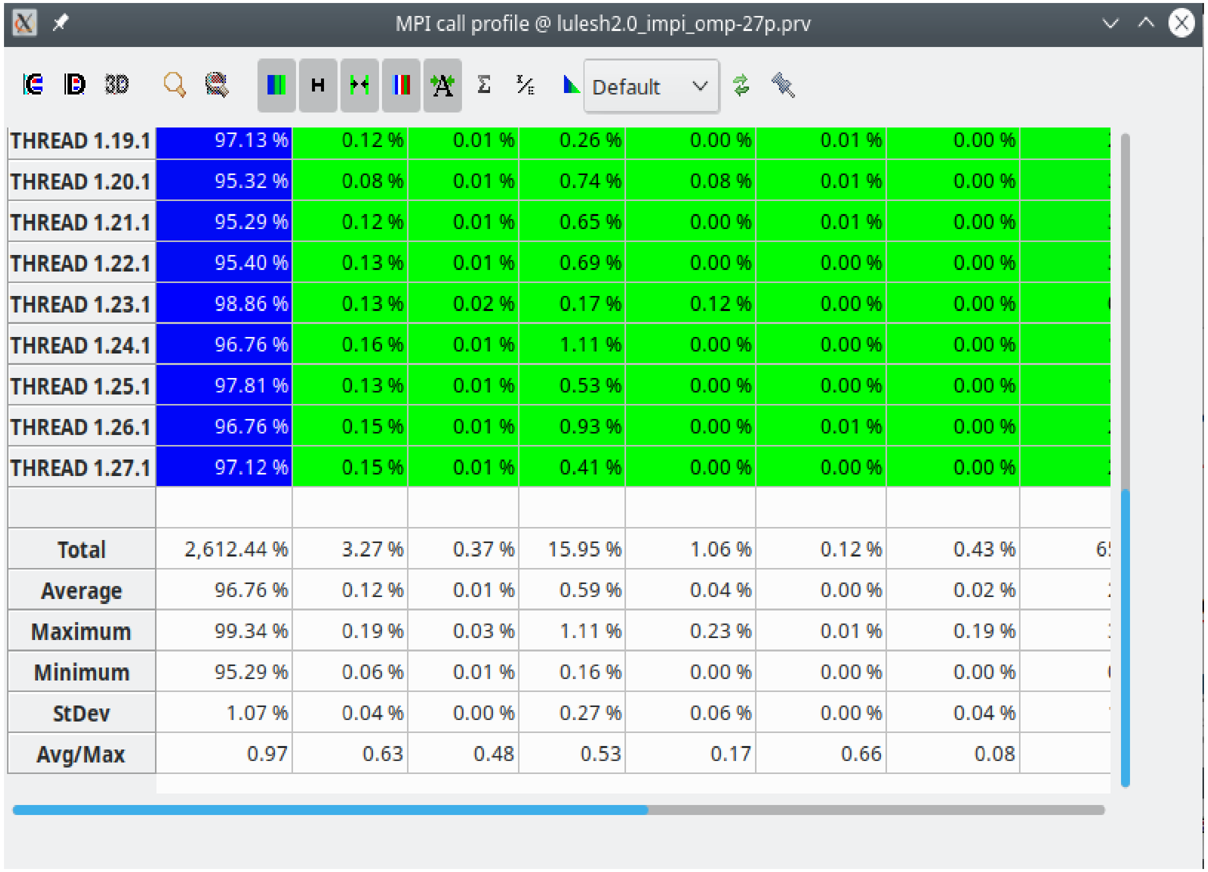
THREAD 1.19.1	97.13 %	0.12 %	0.01 %	0.26 %	0.00 %	0.01 %	0.00 %
THREAD 1.20.1	95.32 %	0.08 %	0.01 %	0.74 %	0.08 %	0.01 %	0.00 %
THREAD 1.21.1	95.29 %	0.12 %	0.01 %	0.65 %	0.00 %	0.01 %	0.00 %
THREAD 1.22.1	95.40 %	0.13 %	0.01 %	0.69 %	0.00 %	0.00 %	0.00 %
THREAD 1.23.1	98.86 %	0.13 %	0.02 %	0.17 %	0.12 %	0.00 %	0.00 %
THREAD 1.24.1	96.76 %	0.16 %	0.01 %	1.11 %	0.00 %	0.00 %	0.00 %
THREAD 1.25.1	97.81 %	0.13 %	0.01 %	0.53 %	0.00 %	0.00 %	0.00 %
THREAD 1.26.1	96.76 %	0.15 %	0.01 %	0.93 %	0.00 %	0.01 %	0.00 %
THREAD 1.27.1	97.12 %	0.15 %	0.01 %	0.41 %	0.00 %	0.00 %	0.00 %
Total	2,612.44 %	3.27 %	0.37 %	15.95 %	1.06 %	0.12 %	0.43 %
Average	96.76 %	0.12 %	0.01 %	0.59 %	0.04 %	0.00 %	0.02 %
Maximum	99.34 %	0.19 %	0.03 %	1.11 %	0.23 %	0.01 %	0.19 %
Minimum	95.29 %	0.06 %	0.01 %	0.16 %	0.00 %	0.00 %	0.00 %
StDev	1.07 %	0.04 %	0.00 %	0.27 %	0.06 %	0.00 %	0.04 %
Avg/Max	0.97	0.63	0.48	0.53	0.17	0.66	0.08

Drag & drop on this area to zoom on the iterative region

Recalculate efficiency of iterative region

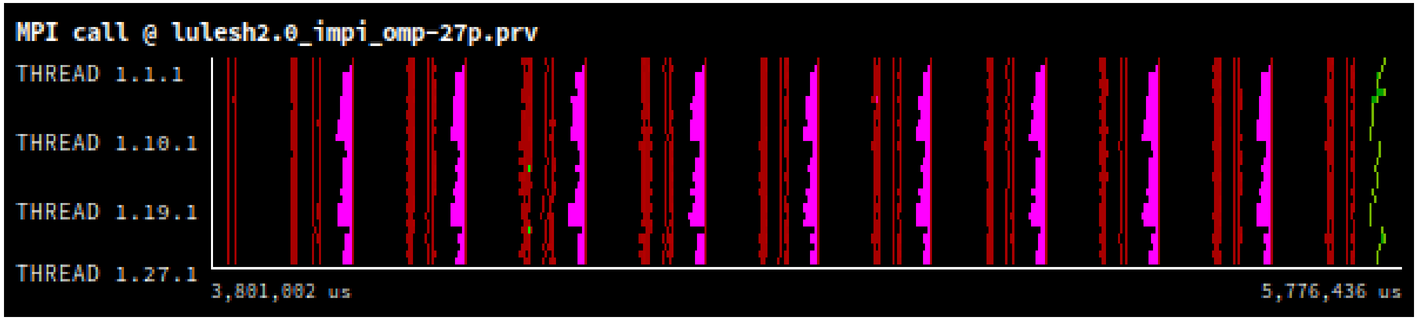


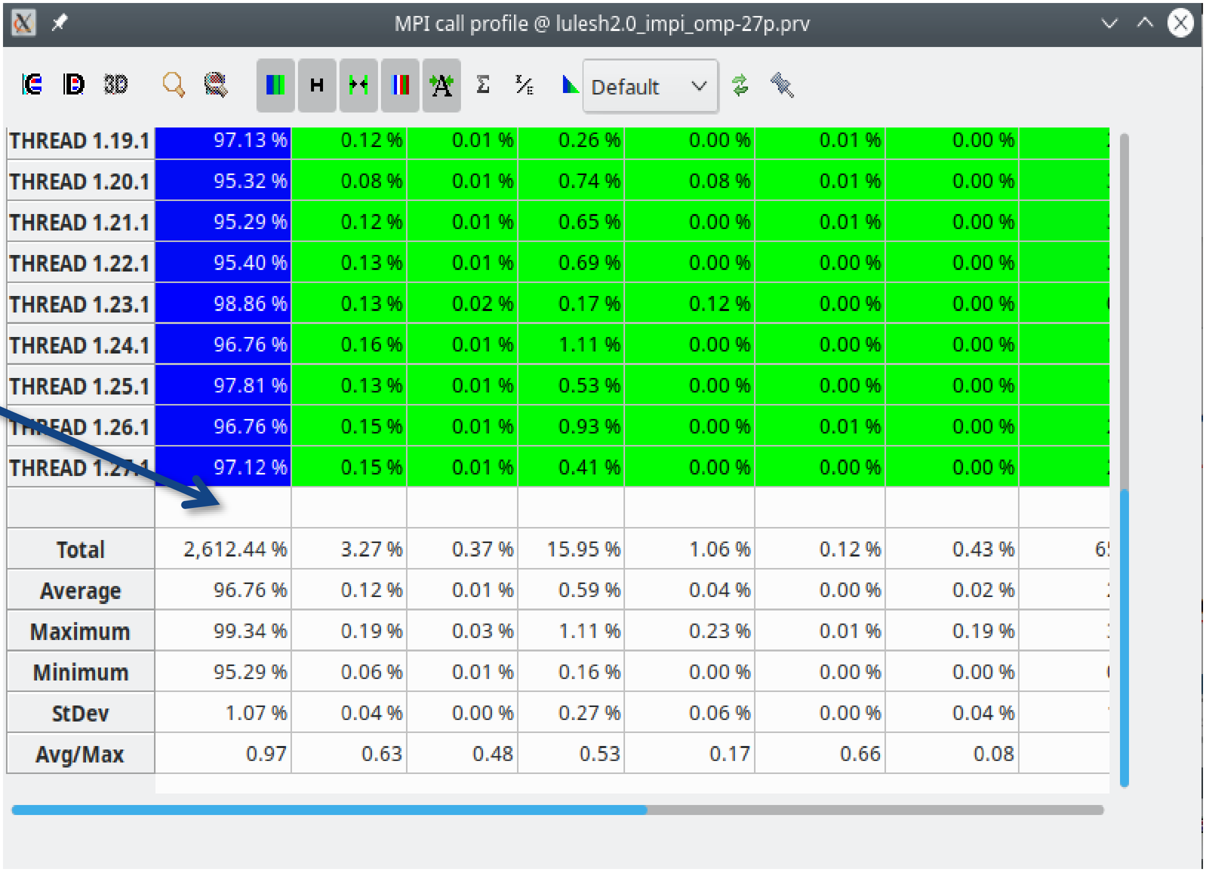
Right click -> Copy



THREAD 1.19.1	97.13 %	0.12 %	0.01 %	0.26 %	0.00 %	0.01 %	0.00 %
THREAD 1.20.1	95.32 %	0.08 %	0.01 %	0.74 %	0.08 %	0.01 %	0.00 %
THREAD 1.21.1	95.29 %	0.12 %	0.01 %	0.65 %	0.00 %	0.01 %	0.00 %
THREAD 1.22.1	95.40 %	0.13 %	0.01 %	0.69 %	0.00 %	0.00 %	0.00 %
THREAD 1.23.1	98.86 %	0.13 %	0.02 %	0.17 %	0.12 %	0.00 %	0.00 %
THREAD 1.24.1	96.76 %	0.16 %	0.01 %	1.11 %	0.00 %	0.00 %	0.00 %
THREAD 1.25.1	97.81 %	0.13 %	0.01 %	0.53 %	0.00 %	0.00 %	0.00 %
THREAD 1.26.1	96.76 %	0.15 %	0.01 %	0.93 %	0.00 %	0.01 %	0.00 %
THREAD 1.27.1	97.12 %	0.15 %	0.01 %	0.41 %	0.00 %	0.00 %	0.00 %
Total	2,612.44 %	3.27 %	0.37 %	15.95 %	1.06 %	0.12 %	0.43 %
Average	96.76 %	0.12 %	0.01 %	0.59 %	0.04 %	0.00 %	0.02 %
Maximum	99.34 %	0.19 %	0.03 %	1.11 %	0.23 %	0.01 %	0.19 %
Minimum	95.29 %	0.06 %	0.01 %	0.16 %	0.00 %	0.00 %	0.00 %
StDev	1.07 %	0.04 %	0.00 %	0.27 %	0.06 %	0.00 %	0.04 %
Avg/Max	0.97	0.63	0.48	0.53	0.17	0.66	0.08

Recalculate efficiency of iterative region

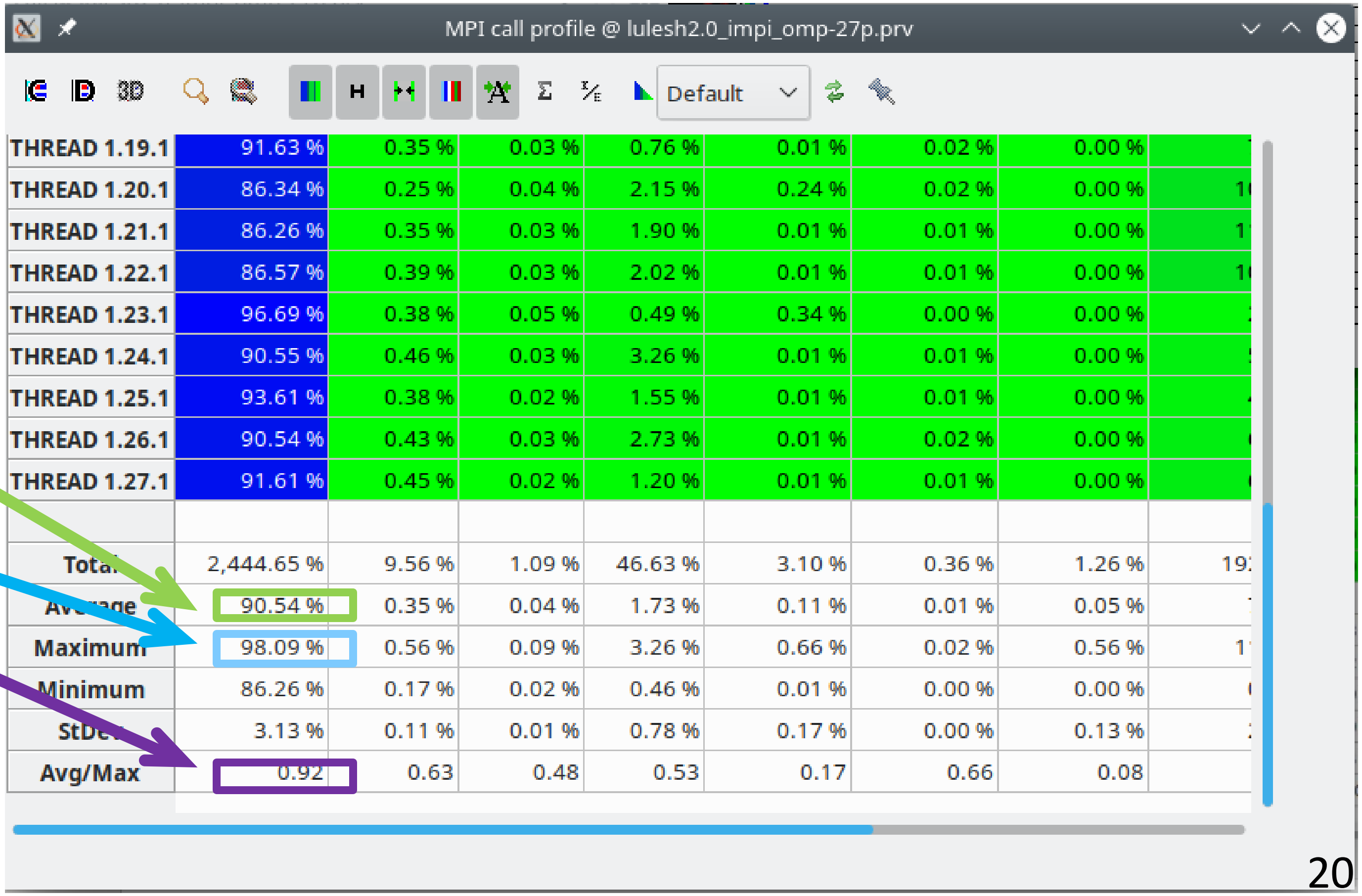




THREAD 1.19.1	97.13 %	0.12 %	0.01 %	0.26 %	0.00 %	0.01 %	0.00 %
THREAD 1.20.1	95.32 %	0.08 %	0.01 %	0.74 %	0.08 %	0.01 %	0.00 %
THREAD 1.21.1	95.29 %	0.12 %	0.01 %	0.65 %	0.00 %	0.01 %	0.00 %
THREAD 1.22.1	95.40 %	0.13 %	0.01 %	0.69 %	0.00 %	0.00 %	0.00 %
THREAD 1.23.1	98.86 %	0.13 %	0.02 %	0.17 %	0.12 %	0.00 %	0.00 %
THREAD 1.24.1	96.76 %	0.16 %	0.01 %	1.11 %	0.00 %	0.00 %	0.00 %
THREAD 1.25.1	97.81 %	0.13 %	0.01 %	0.53 %	0.00 %	0.00 %	0.00 %
THREAD 1.26.1	96.76 %	0.15 %	0.01 %	0.93 %	0.00 %	0.01 %	0.00 %
THREAD 1.27.1	97.12 %	0.15 %	0.01 %	0.41 %	0.00 %	0.00 %	0.00 %
Total	2,612.44 %	3.27 %	0.37 %	15.95 %	1.06 %	0.12 %	0.43 %
Average	96.76 %	0.12 %	0.01 %	0.59 %	0.04 %	0.00 %	0.02 %
Maximum	99.34 %	0.19 %	0.03 %	1.11 %	0.23 %	0.01 %	0.19 %
Minimum	95.29 %	0.06 %	0.01 %	0.16 %	0.00 %	0.00 %	0.00 %
StDev	1.07 %	0.04 %	0.00 %	0.27 %	0.06 %	0.00 %	0.04 %
Avg/Max	0.97	0.63	0.48	0.53	0.17	0.66	0.08

Right click -> Paste -> Time

Efficiency of iterative region



THREAD 1.19.1	91.63 %	0.35 %	0.03 %	0.76 %	0.01 %	0.02 %	0.00 %	
THREAD 1.20.1	86.34 %	0.25 %	0.04 %	2.15 %	0.24 %	0.02 %	0.00 %	1
THREAD 1.21.1	86.26 %	0.35 %	0.03 %	1.90 %	0.01 %	0.01 %	0.00 %	1
THREAD 1.22.1	86.57 %	0.39 %	0.03 %	2.02 %	0.01 %	0.01 %	0.00 %	1
THREAD 1.23.1	96.69 %	0.38 %	0.05 %	0.49 %	0.34 %	0.00 %	0.00 %	
THREAD 1.24.1	90.55 %	0.46 %	0.03 %	3.26 %	0.01 %	0.01 %	0.00 %	
THREAD 1.25.1	93.61 %	0.38 %	0.02 %	1.55 %	0.01 %	0.01 %	0.00 %	
THREAD 1.26.1	90.54 %	0.43 %	0.03 %	2.73 %	0.01 %	0.02 %	0.00 %	
THREAD 1.27.1	91.61 %	0.45 %	0.02 %	1.20 %	0.01 %	0.01 %	0.00 %	
Total	2,444.65 %	9.56 %	1.09 %	46.63 %	3.10 %	0.36 %	1.26 %	19
Average	90.54 %	0.35 %	0.04 %	1.73 %	0.11 %	0.01 %	0.05 %	
Maximum	98.09 %	0.56 %	0.09 %	3.26 %	0.66 %	0.02 %	0.56 %	1
Minimum	86.26 %	0.17 %	0.02 %	0.46 %	0.01 %	0.00 %	0.00 %	
StDev	3.13 %	0.11 %	0.01 %	0.78 %	0.17 %	0.00 %	0.13 %	
Avg/Max	0.92	0.63	0.48	0.53	0.17	0.66	0.08	

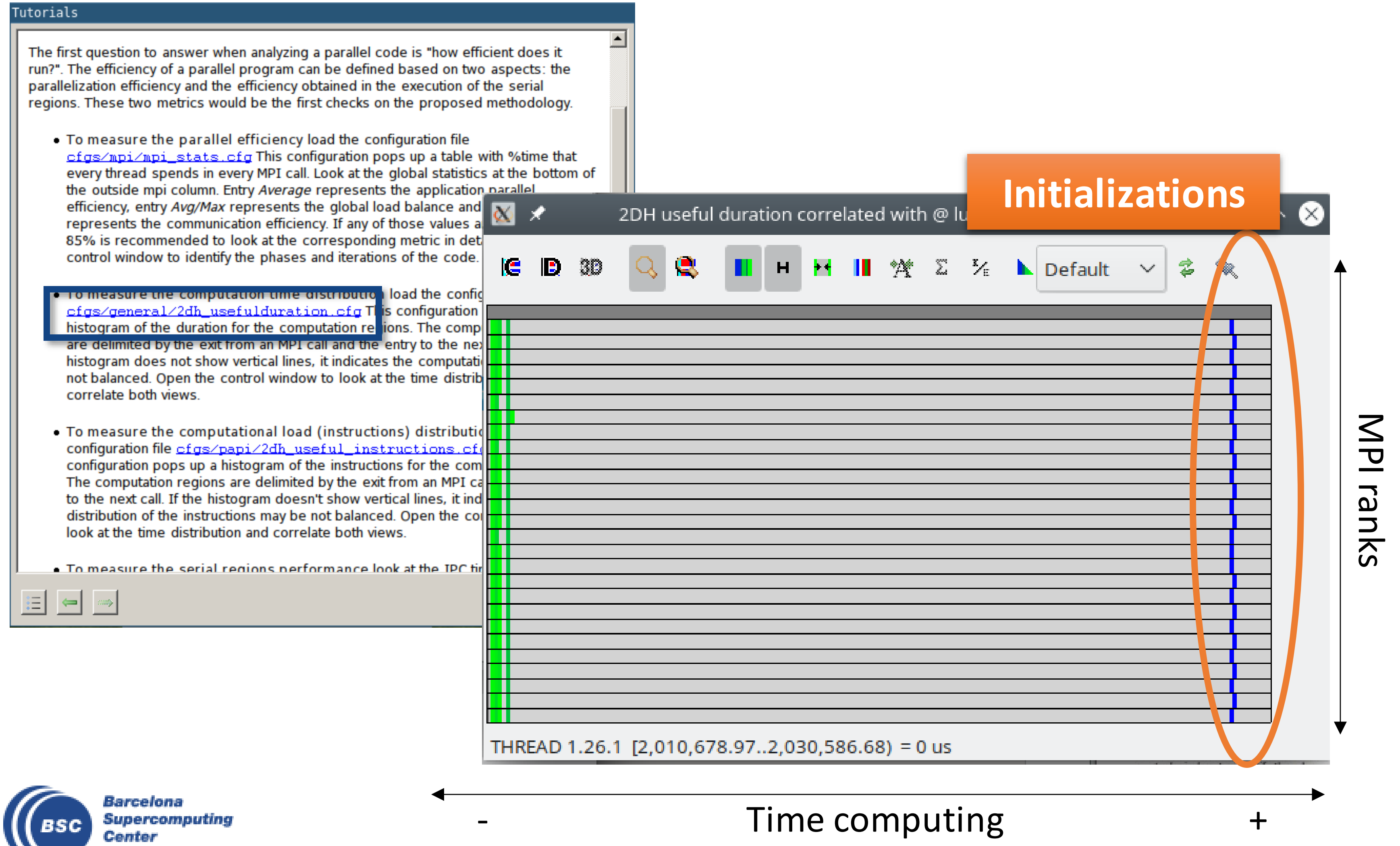
Parallel efficiency

Comm efficiency

Load balance

Computation time distribution

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



Tutorials

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

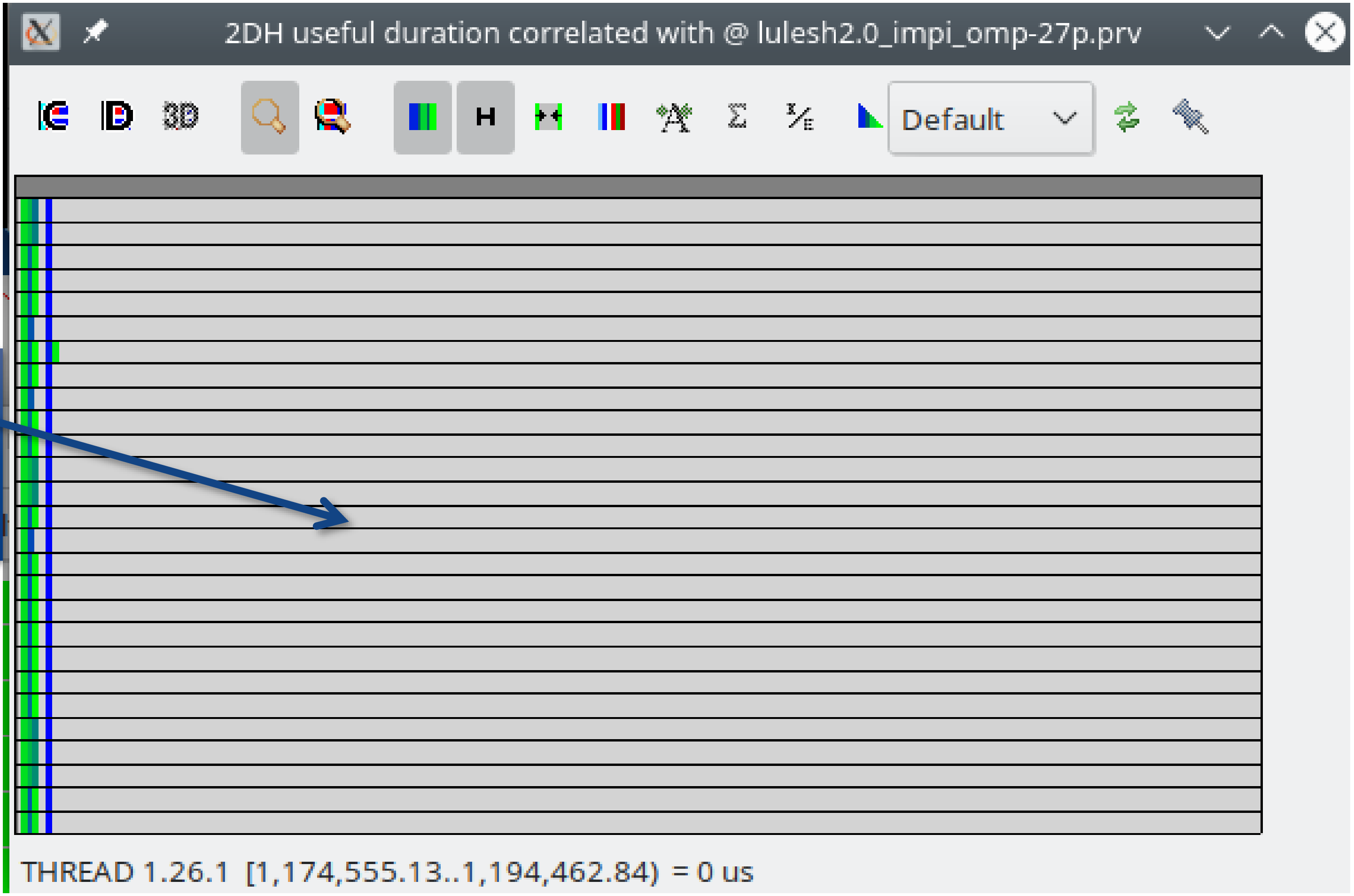
- To measure the parallel efficiency load the configuration file cfgs/mpi/mpi_stats.cfg This configuration pops up a table with %time that every thread spends in every MPI call. Look at the global statistics at the bottom of the outside mpi column. Entry *Average* represents the application parallel efficiency, entry *Avg/Max* represents the global load balance and represents the communication efficiency. If any of those values a 85% is recommended to look at the corresponding metric in det control window to identify the phases and iterations of the code.
- To measure the computation time distributio load the config cfgs/general/2dh_usefulduration.cfg This configuration histogram of the duration for the computation re ions. The comp are delimited by the exit from an MPI call and the entry to the ne histogram does not show vertical lines, it indicates the computati not balanced. Open the control window to look at the time distrib correlate both views.
- To measure the computational load (instructions) distributi configuration file cfgs/papi/2dh_useful_instructions.cf configuration pops up a histogram of the instructions for the com The computation regions are delimited by the exit from an MPI ca to the next call. If the histogram doesn't show vertical lines, it ind distribution of the instructions may be not balanced. Open the co look at the time distribution and correlate both views.
- To measure the serial regions performance look at the IPC ti

2DH useful duration correlated with @ lu

THREAD 1.26.1 [2,010,678.97..2,030,586.68) = 0 us

Focus on the iterative part

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

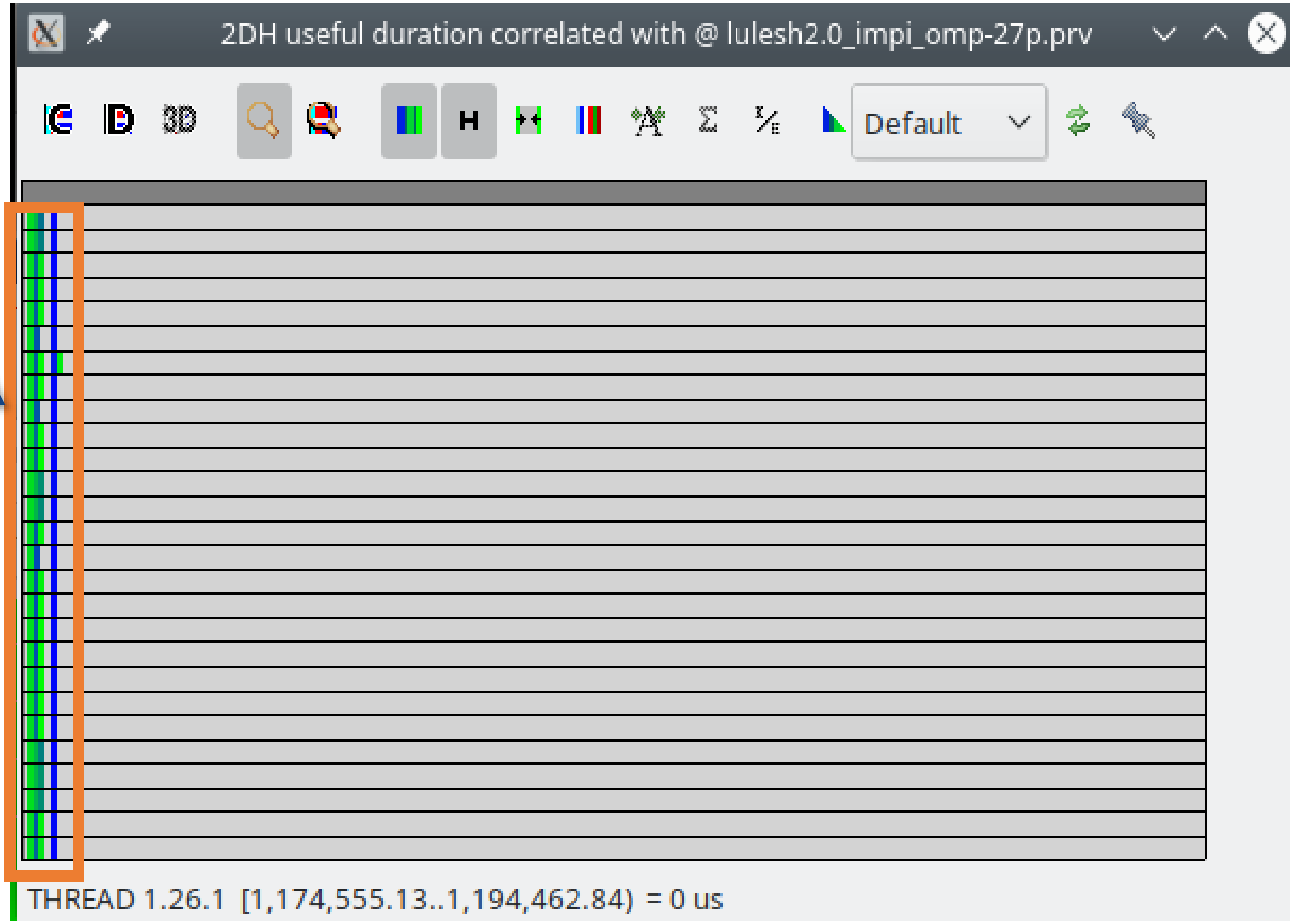

2DH useful duration correlated with @ lulesh2.0_impi_omp-27p.prv
3D
H
Default

THREAD 1.26.1 [1,174,555.13..1,194,462.84) = 0 us

Right click -> Paste -> Time

Focus on the iterative part

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


2DH useful duration correlated with @ lulesh2.0_impi_omp-27p.prv
3D
H
Default

THREAD 1.26.1 [1,174,555.13..1,194,462.84) = 0 us

Drag & drop on this area to zoom

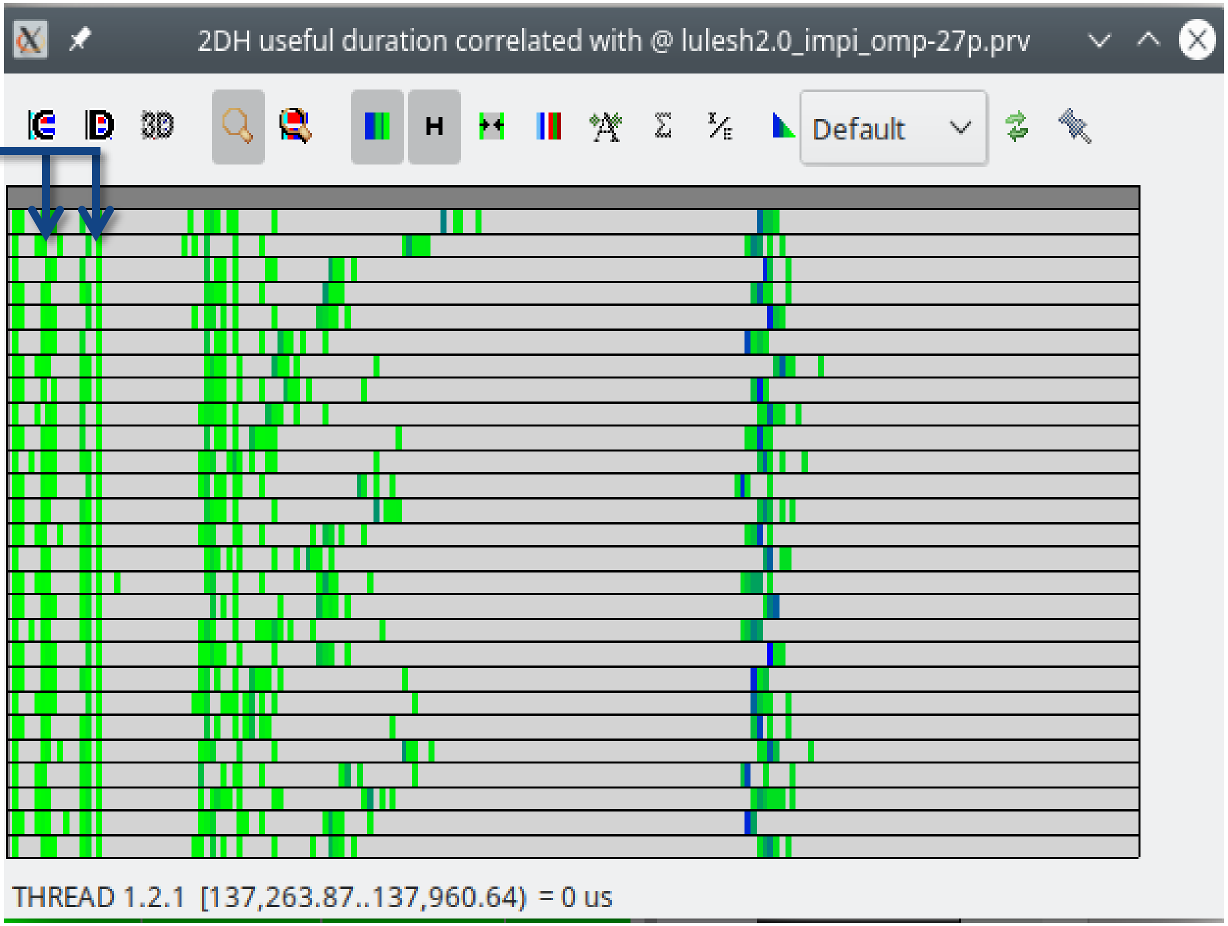
Computation time distribution

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

Mostly straight vertical lines

=

All processes similar computing time

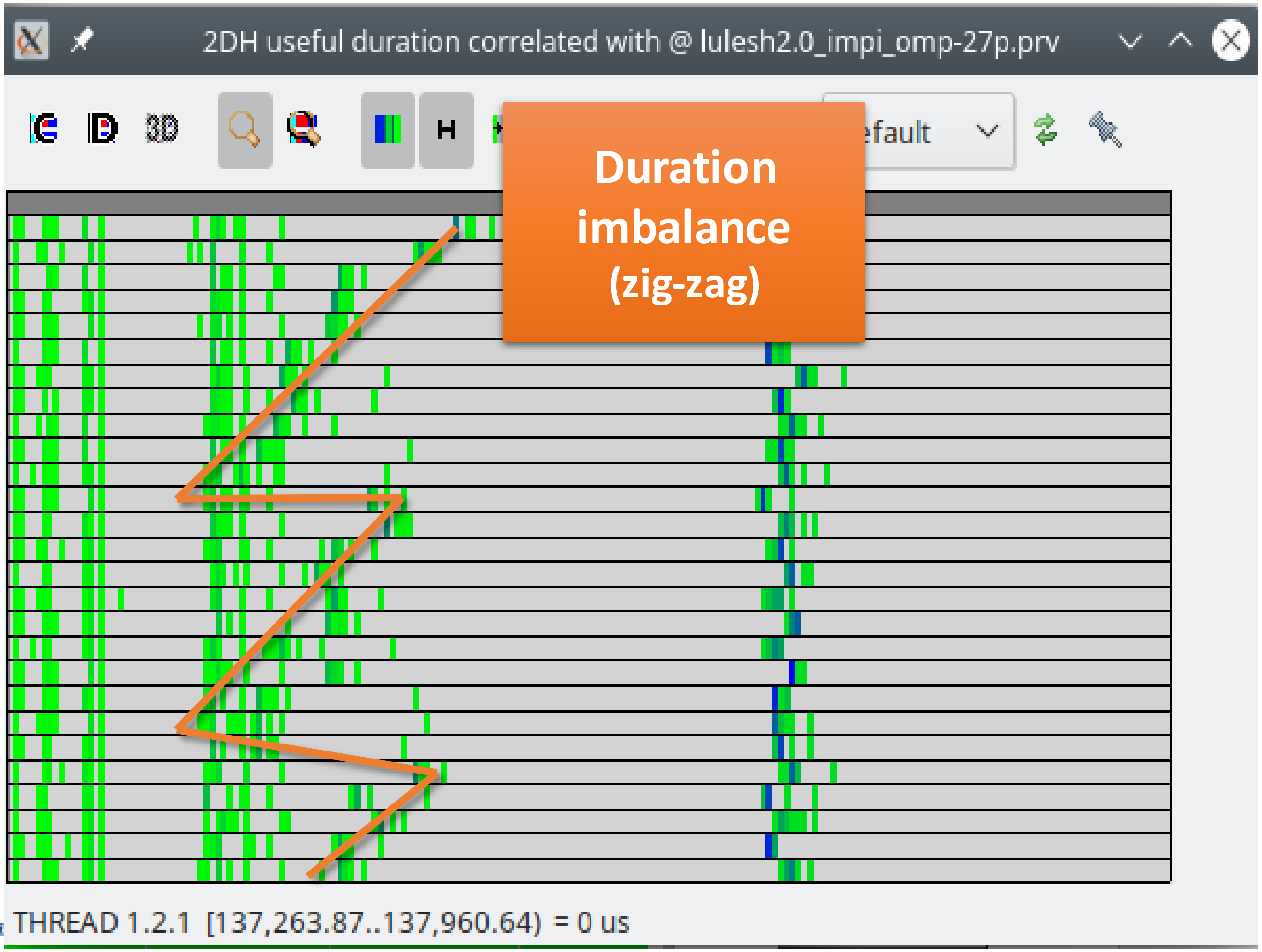


2DH useful duration correlated with @ lulesh2.0_impi_omp-27p.prv
Default

THREAD 1.2.1 [137,263.87..137,960.64) = 0 us

Computation time distribution

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



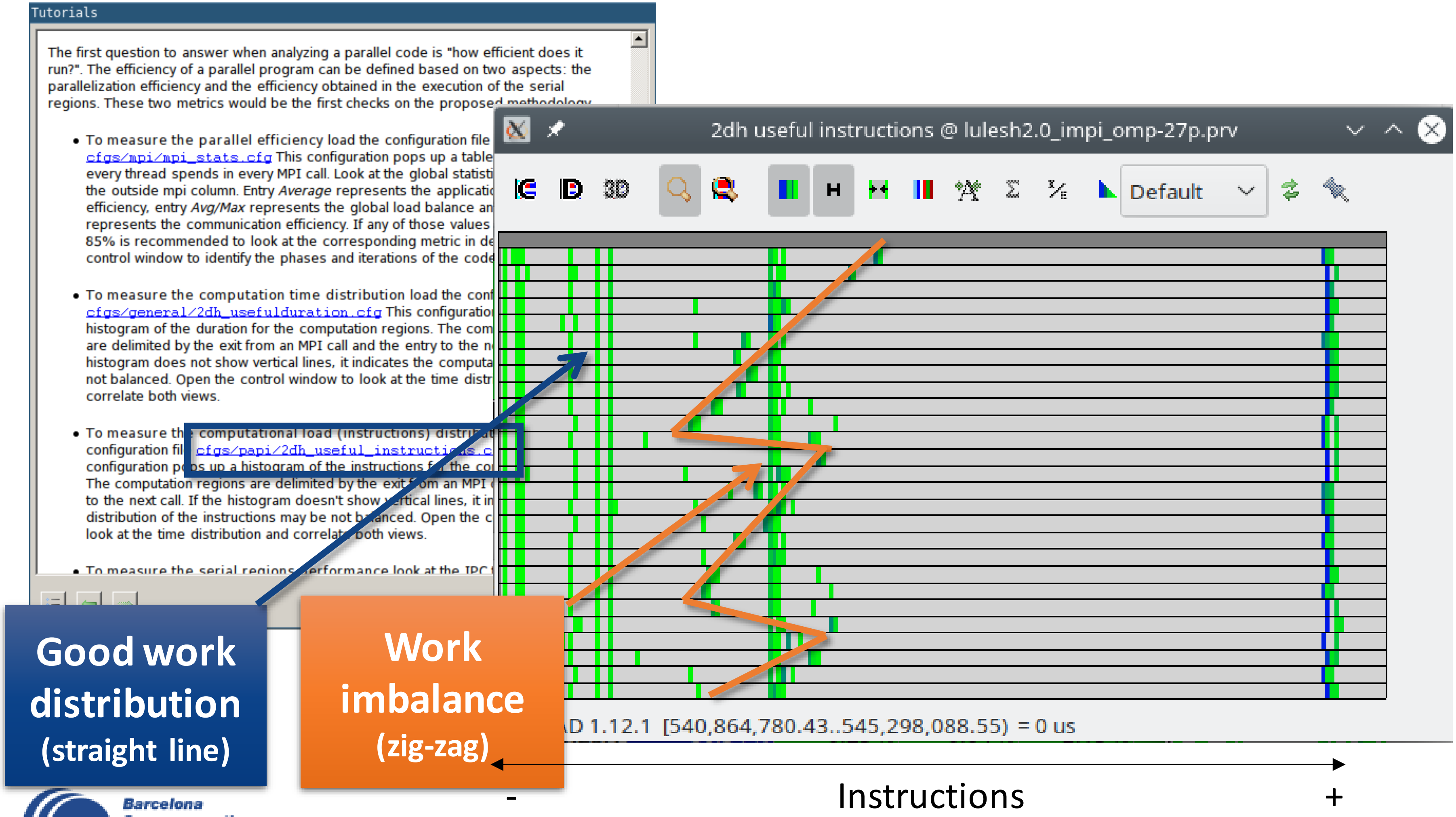
2DH useful duration correlated with @ lulesh2.0_impi_omp-27p.prv
H
efault

Duration imbalance (zig-zag)

THREAD 1.2.1 [137,263.87..137,960.64) = 0 us

Computation load distribution

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



The first question to answer when analyzing a parallel code is "how efficient does it run?". The efficiency of a parallel program can be defined based on two aspects: the parallelization efficiency and the efficiency obtained in the execution of the serial regions. These two metrics 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 every thread spends in every MPI call. Look at the global statisti the outside mpi column. Entry *Average* represents the applicatio efficiency, entry *Avg/Max* represents the global load balance an represents the communication efficiency. If any of those values 85% is recommended to look at the corresponding metric in de control window to identify the phases and iterations of the code
- To measure the computation time distribution load the conf cfgs/general/2dh_usefulduration.cfg This configuratio histogram of the duration for the computation regions. The com are delimited by the exit from an MPI call and the entry to the n histogram does not show vertical lines, it indicates the computa not balanced. Open the control window to look at the time distr correlate both views.
- To measure the computational load (instructions) distributi configuration file cfgs/papi/2dh_useful_instructions.c configuration pops up a histogram of the instructions for the co The computation regions are delimited by the exit from an MPI to the next call. If the histogram doesn't show vertical lines, it in distribution of the instructions may be not balanced. Open the c look at the time distribution and correlate both views.
- To measure the serial regions performance look at the IPC t

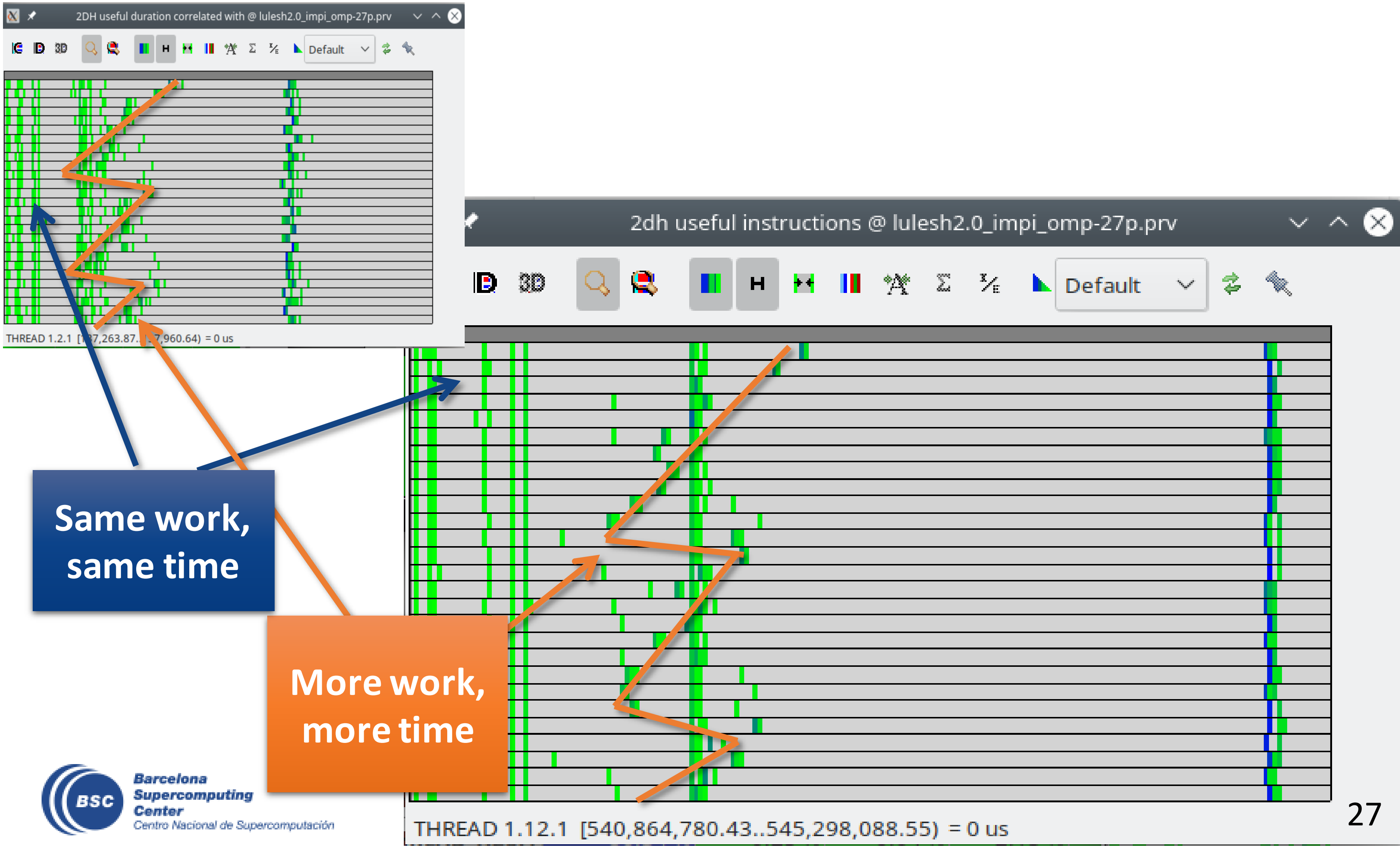
2dh useful instructions @ lulesh2.0_impi_omp-27p.prv
3D
H
Σ
Default
D 1.12.1 [540,864,780.43..545,298,088.55) = 0 us
-
Instructions
+

Good work distribution (straight line)

Work imbalance (zig-zag)

Correlate two histograms

- Clear correlation between the **amount of work** and the **time computing**



2DH useful duration correlated with @ lulesh2.0_impi_omp-27p.prv

THREAD 1.2.1 [1[illegible]7,263.87..[illegible]7,960.64) = 0 us

2dh useful instructions @ lulesh2.0_impi_omp-27p.prv

THREAD 1.12.1 [540,864,780.43..545,298,088.55) = 0 us

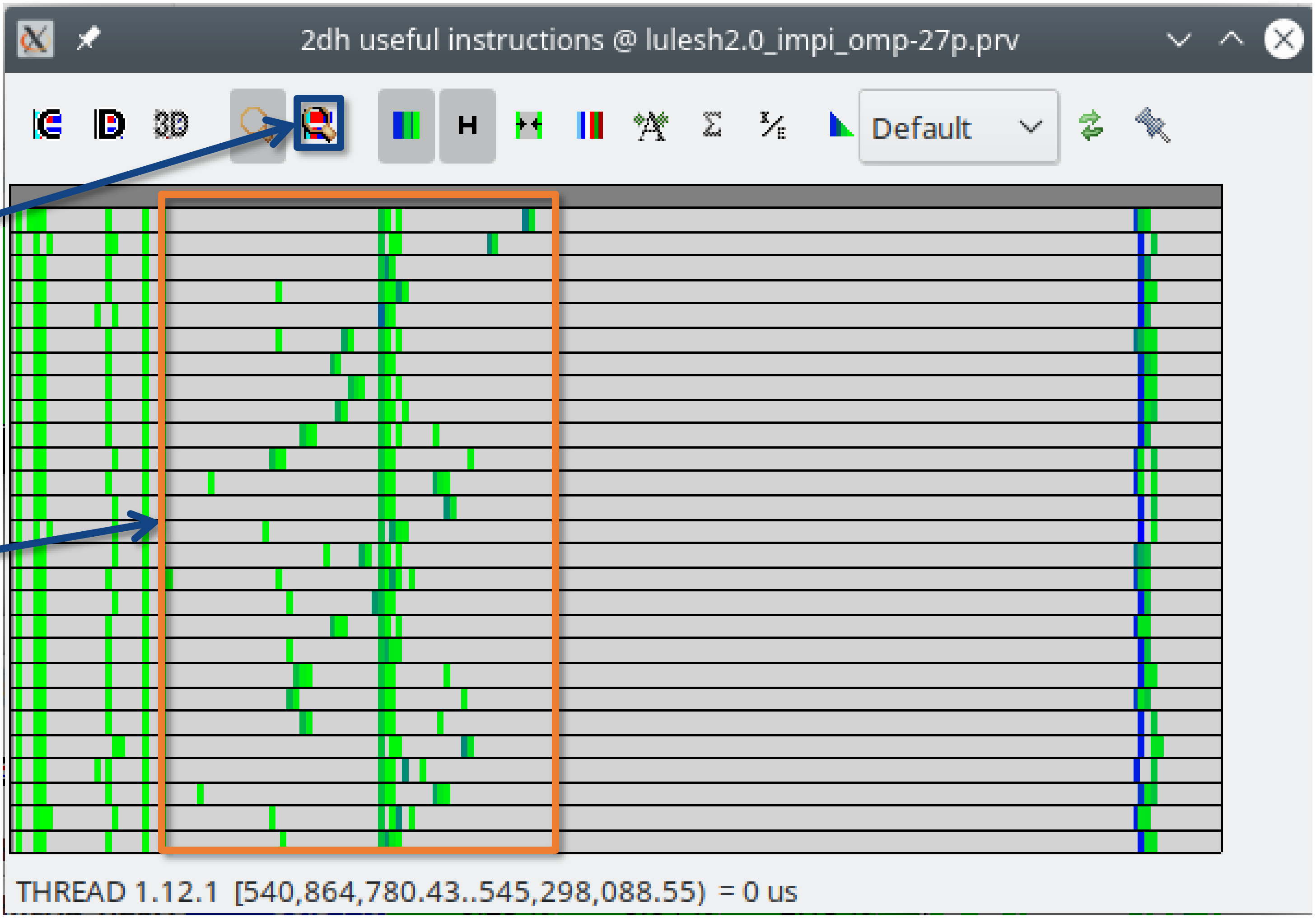
Same work, same time

More work, more time

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Where does this happen?

- Go from the table to the timeline



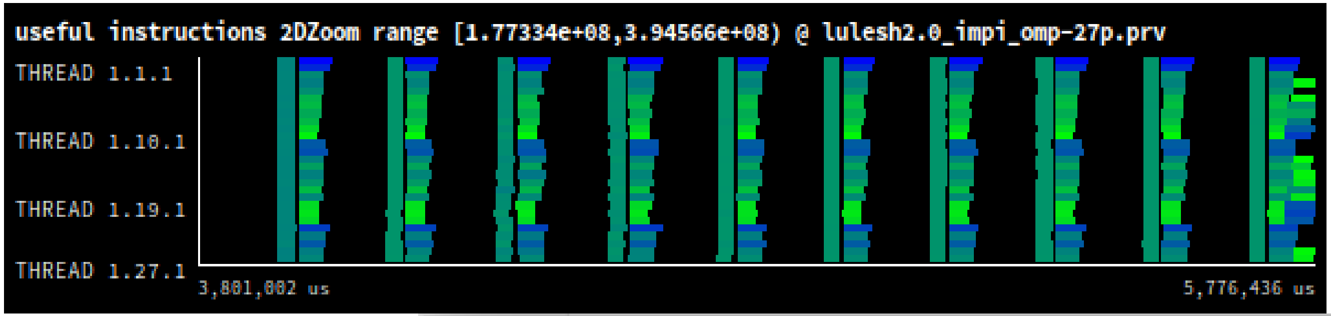
THREAD 1.12.1 [540,864,780.43..545,298,088.55) = 0 us

1. Click on "Open Filtered Control Window"

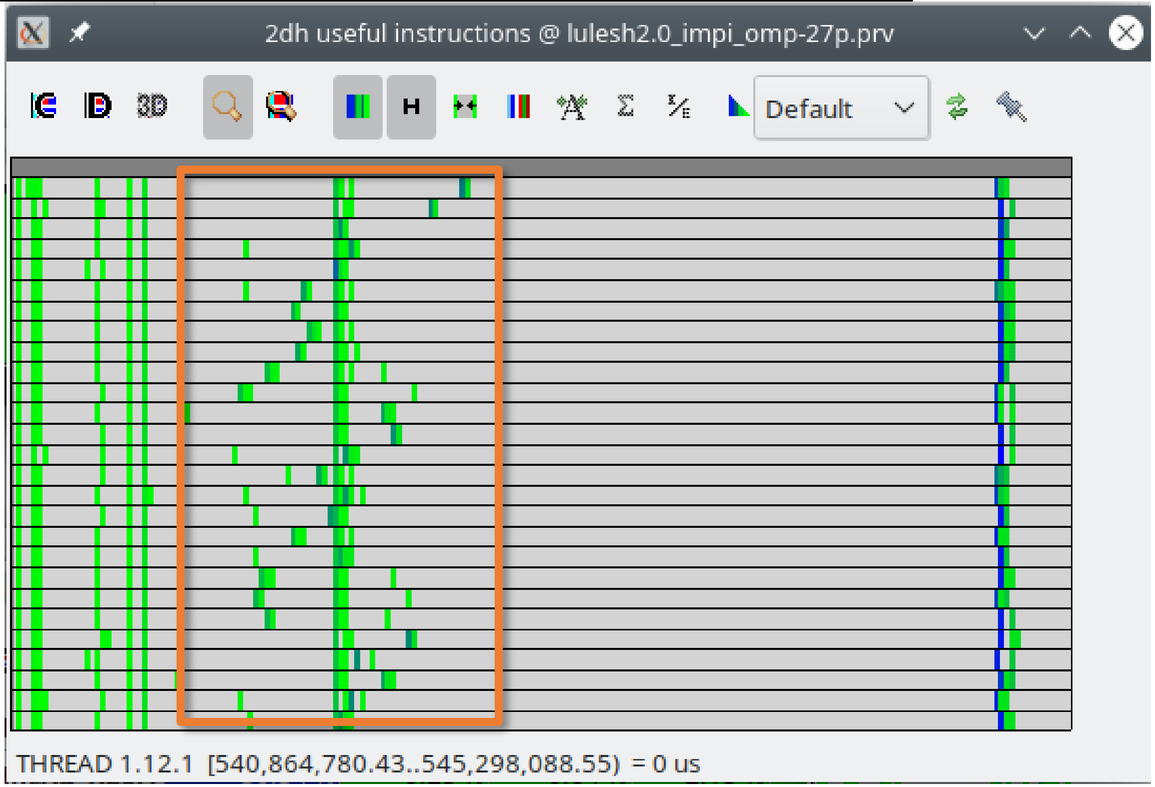
2. Select this area (drag-and-drop)

Where does this happen?

- Go from the table to the timeline



```
useful instructions 2DZoom range [1.77334e+08,3.94566e+08) @ lulesh2.0_impi_omp-27p.prv
THREAD 1.1.1
THREAD 1.10.1
THREAD 1.19.1
THREAD 1.27.1
3,801,002 us                                                     5,776,436 us
```



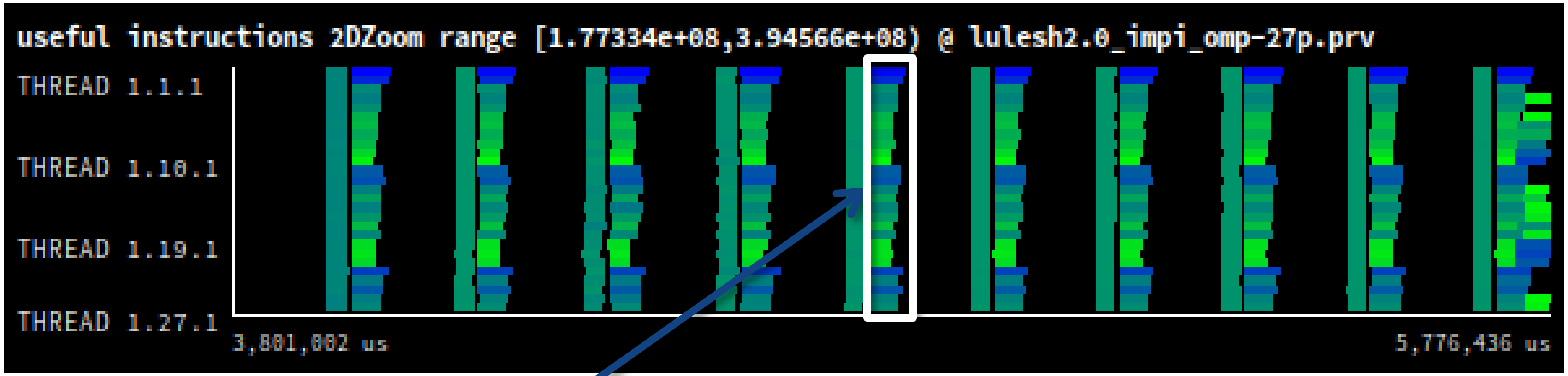
2dh useful instructions @ lulesh2.0_impi_omp-27p.prv

THREAD 1.12.1 [540,864,780.43..545,298,088.55) = 0 us

Clicking here always rescales. Same as… Right click -> Fit Semantic Scale -> Fit Both

Where does this happen?

- Slow & Fast at the same time? -> Imbalance

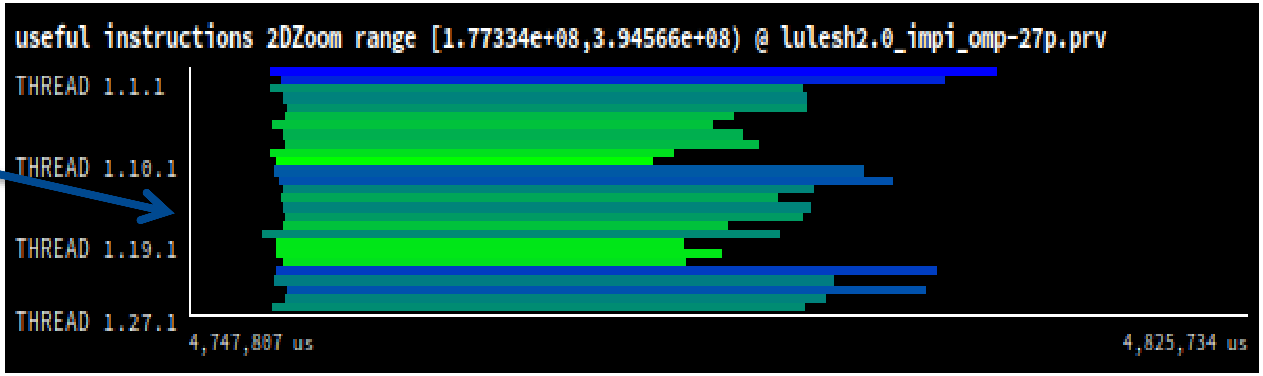


useful instructions 2DZoom range [1.77334e+08,3.94566e+08) @ lulesh2.0_impi_omp-27p.prv

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

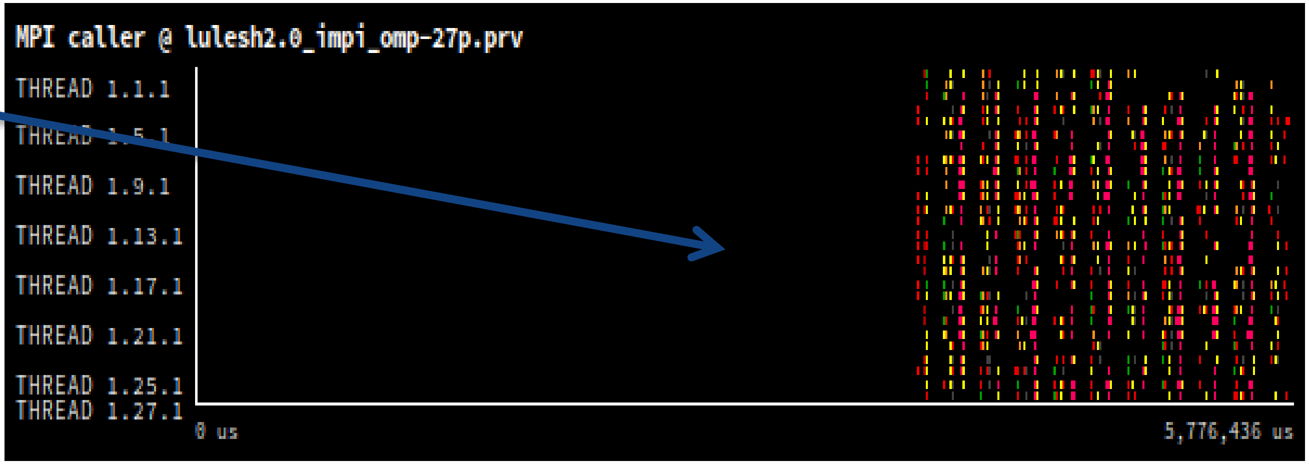
Where does this happen?

- Slow & Fast at the same time? -> Imbalance


useful instructions 2DZoom range [1.77334e+08,3.94566e+08) @ lulesh2.0_impi_omp-27p.prv

1. Right click -> Copy

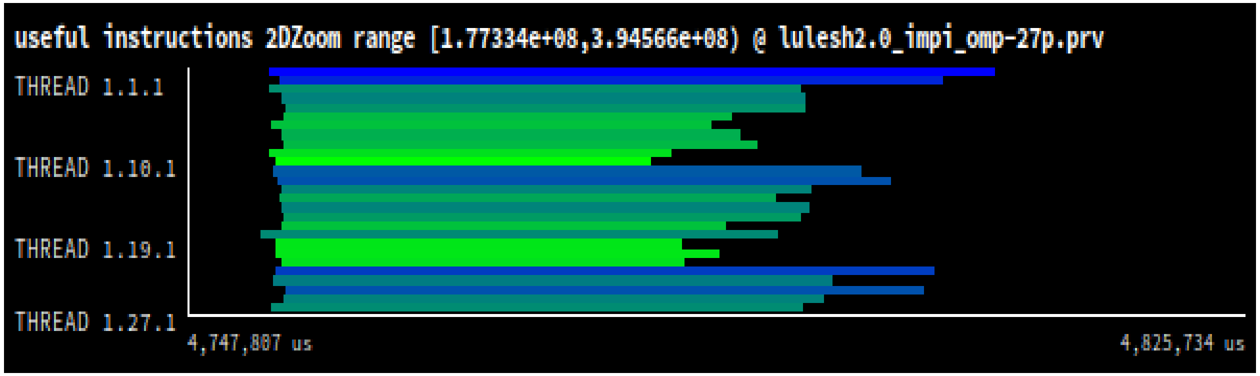
- Hints -> Call stack references -> Caller function


MPI caller @ lulesh2.0_impi_omp-27p.prv

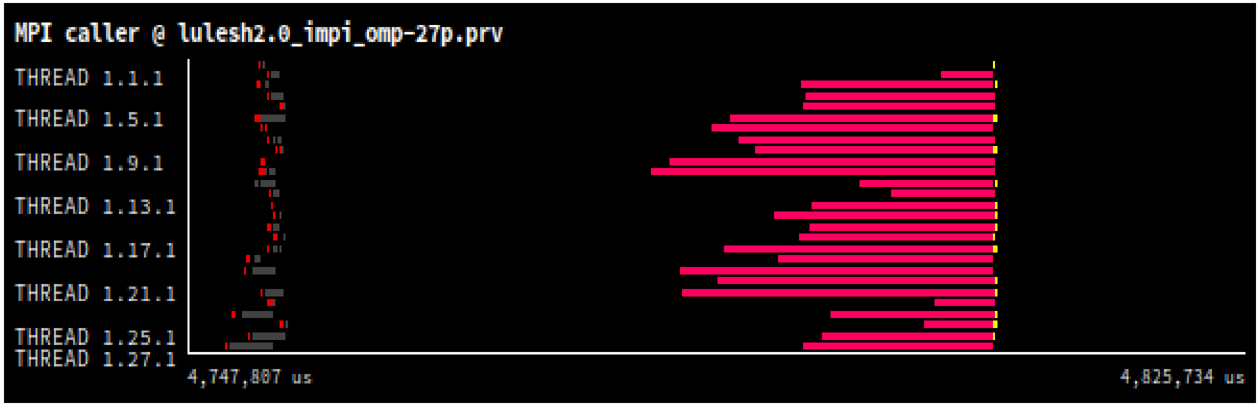
2. Right click -> Paste -> Time

Where does this happen?

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


useful instructions 2DZoom range [1.77334e+08,3.94566e+08) @ lulesh2.0_impi_omp-27p.prv
THREAD 1.1.1
THREAD 1.10.1
THREAD 1.19.1
THREAD 1.27.1
4,747,887 us
4,825,734 us

- **Hints -> Call stack references -> Caller function**

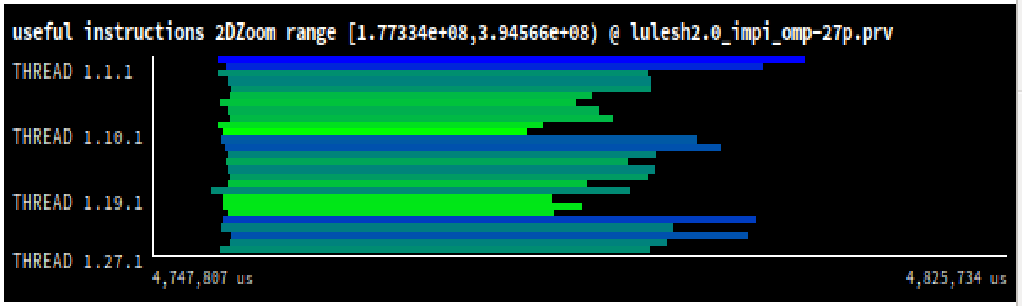


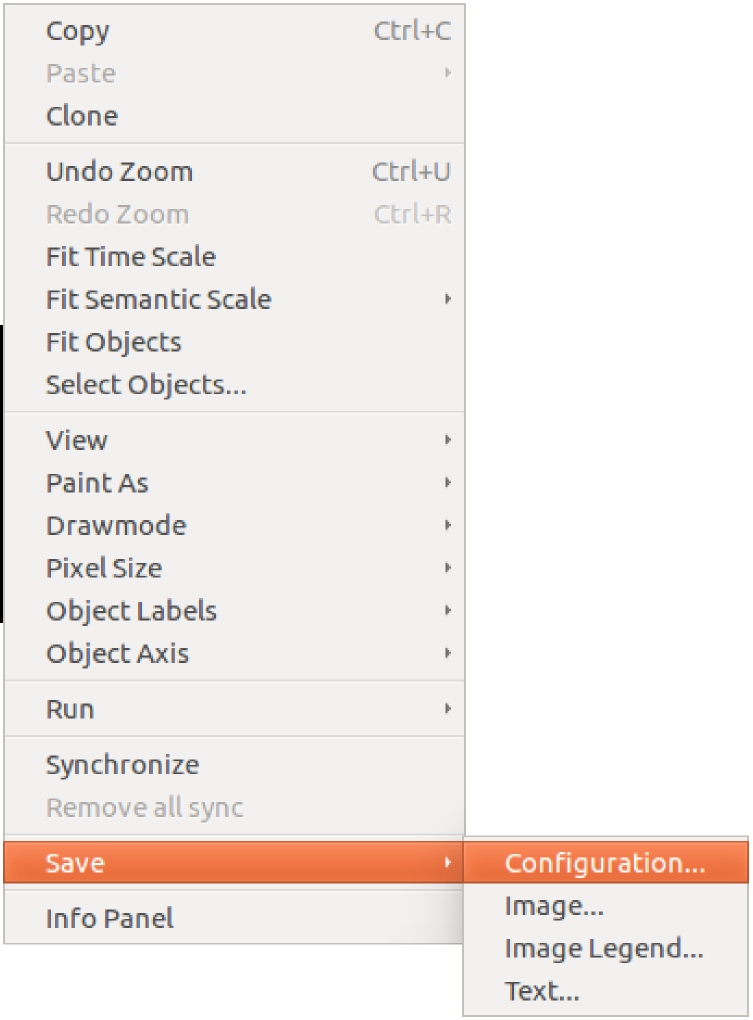
CommSend CommMonoQ

- ■ End
- 🟨 CommRecv
- 🟥 CommSend
- ■ CommMonoQ
- ■ _INTERNA..ncrement [_INTERNAL89f4daf0::TimeIncrement]

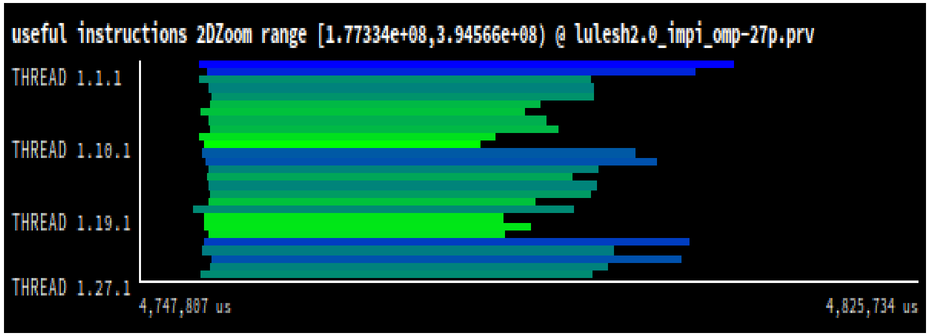
Save CFG's (method 1)

Right click on timeline

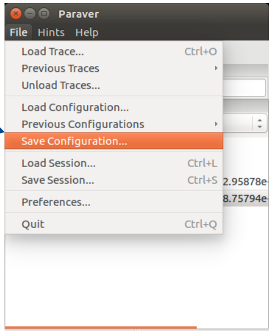

useful instructions 2DZoom range [1.77334e+08,3.94566e+08) @ lulesh2.0_impi_omp-27p.prv
THREAD 1.1.1
THREAD 1.10.1
THREAD 1.19.1
THREAD 1.27.1
4,747,807 us
4,825,734 us

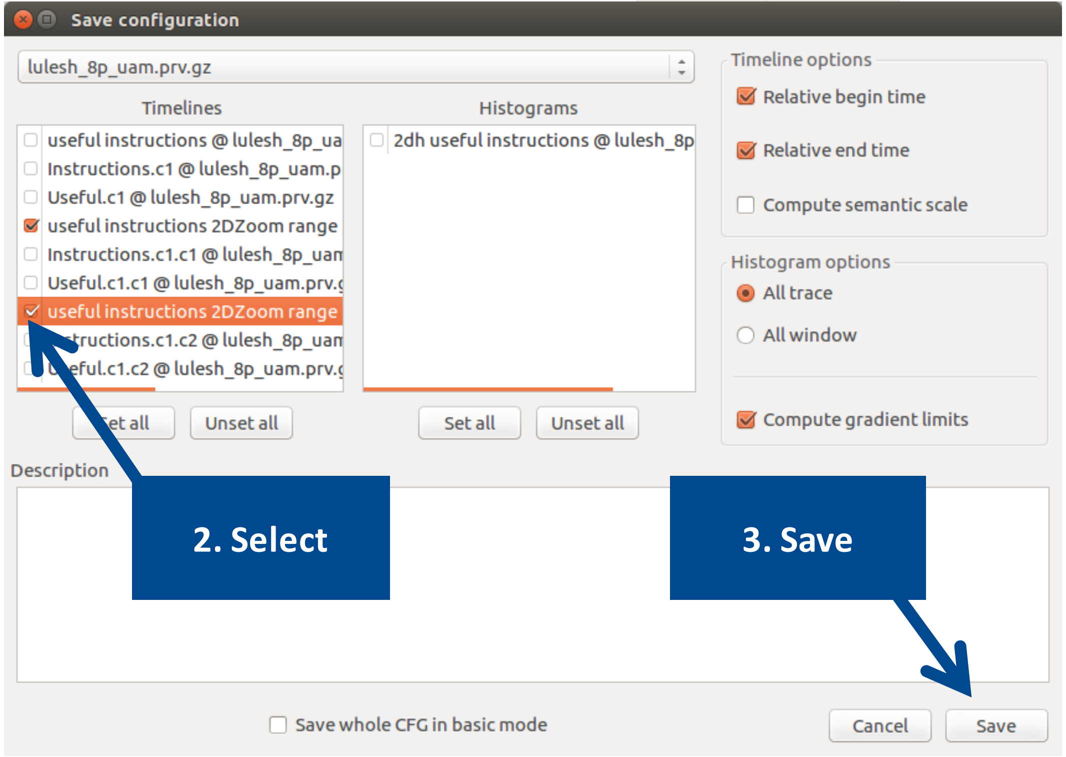


Save CFG's (method 2)



1. Main Paraver window → File → Save Configuration




Save configuration

lulesh_8p_uam.prv.gz

Timelines	Histograms
□ useful instructions @ lulesh_8p_ua	□ 2dh useful instructions @ lulesh_8p
□ 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	

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

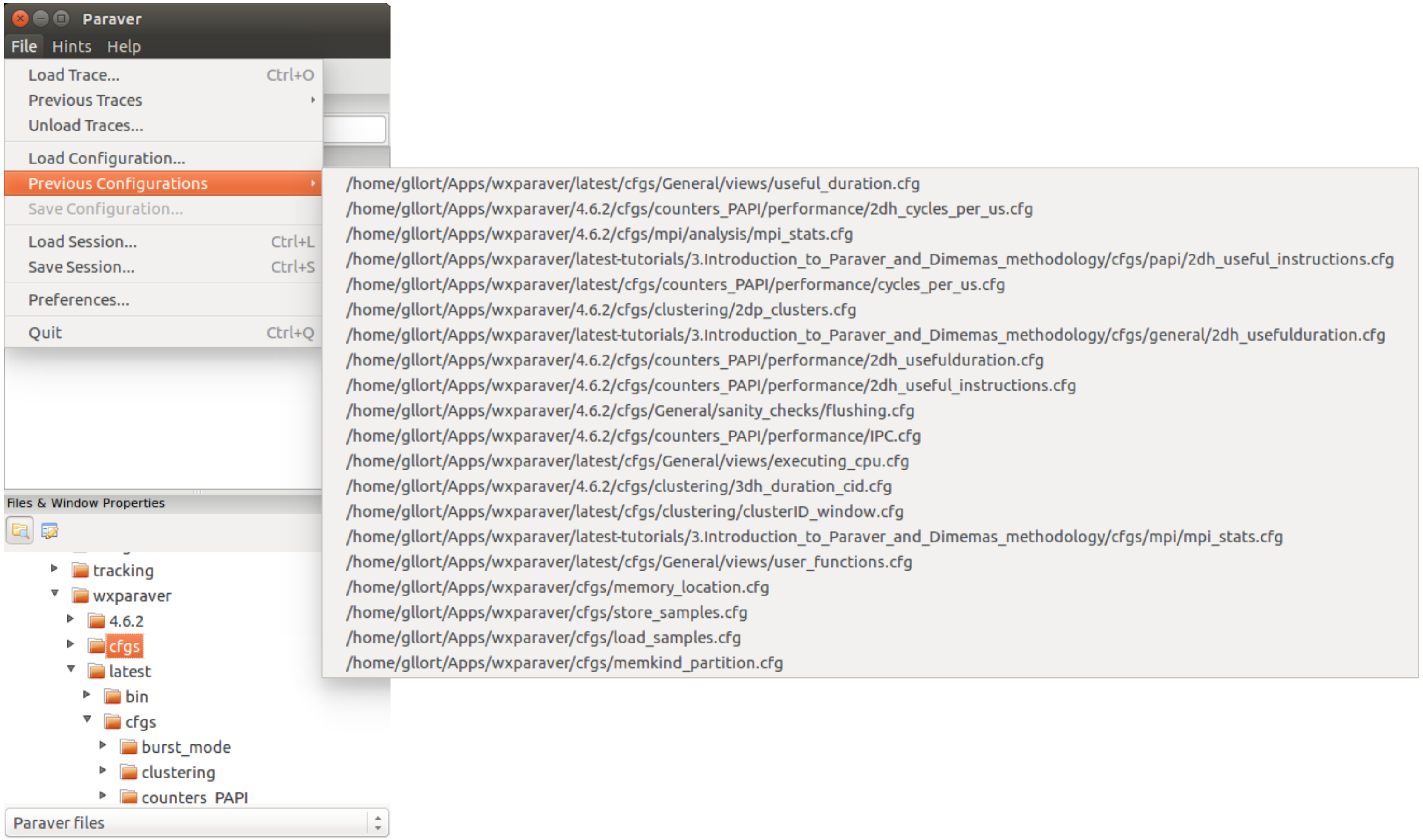
2. Select

3. Save

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

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
```

Basic Analysis tool

BasicAnalysis is a tool to extract POP efficiency metrics (BSC multiplicative model) from Paraver traces.

Installation

There is no installation required. Just copy the content of package into your preferred location and add such directory to the PATH environment variable.

Prerequisites

It relies on paramedir and Dimemas being installed and available through the PATH environment variable.

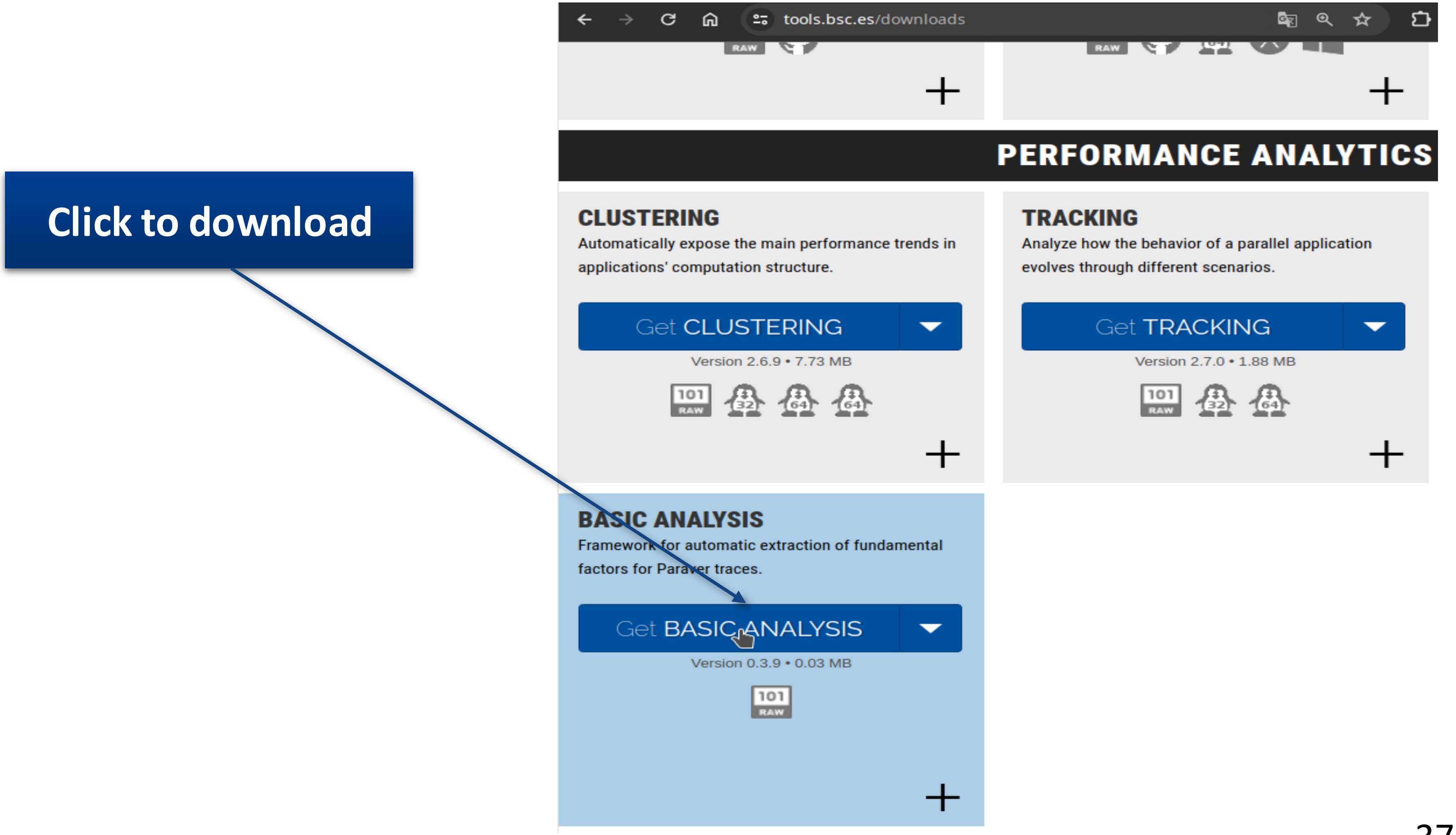
- **paramedir** available at https://tools.bsc.es/paraver
- **Dimemas (optional)** available at https://tools.bsc.es/dimemas

Usage example

- modelfactors.py <list-of-traces>
- modelfactors.py --help

Download BasicAnalysis

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



Click to download

tools.bsc.es/downloads

PERFORMANCE ANALYTICS

CLUSTERING

Automatically expose the main performance trends in applications' computation structure.

Version 2.6.9 • 7.73 MB

TRACKING

Analyze how the behavior of a parallel application evolves through different scenarios.

Version 2.7.0 • 1.88 MB

BASIC ANALYSIS

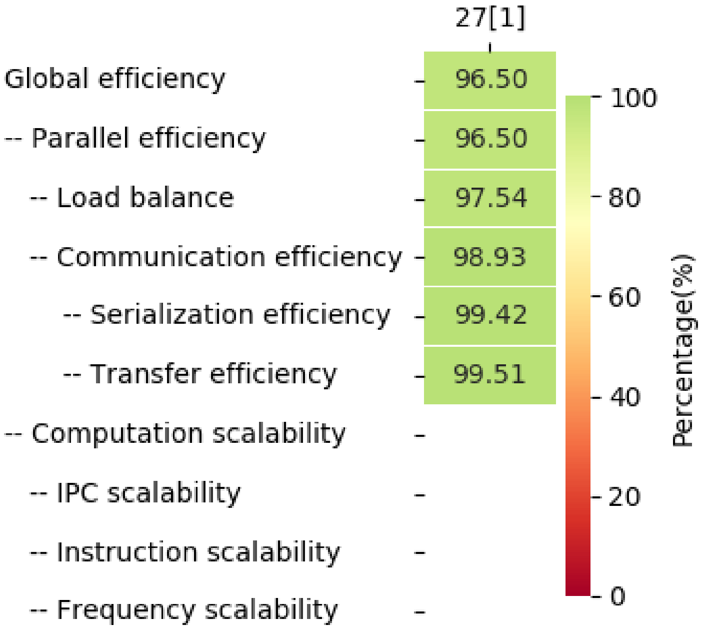
Framework for automatic extraction of fundamental factors for Paraver traces.

Version 0.3.9 • 0.03 MB

Basic Analysis Tool – Efficiency Table

```
modelfactors.py lulesh2.0_impi_omp-27p.prv
```

Full trace



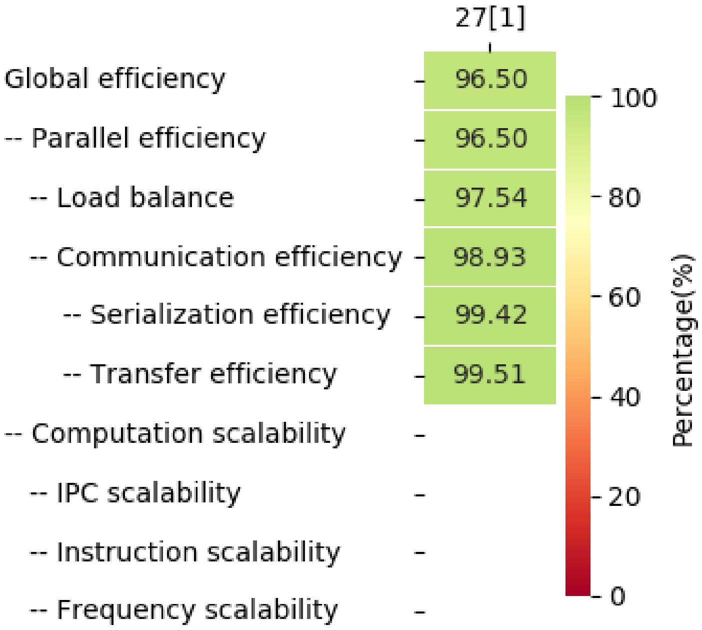
	27[1]
Global efficiency	96.50
-- Parallel efficiency	96.50
-- Load balance	97.54
-- Communication efficiency	98.93
-- Serialization efficiency	99.42
-- Transfer efficiency	99.51
-- Computation scalability	
-- IPC scalability	
-- Instruction scalability	
-- Frequency scalability	

- The higher the better
- Communication submetrics by using Dimemas simulator: Serialization and Transfer Efficiencies.
- User can find the simulated trace in scratch_out_basicanalysis folder. This trace can be analyzed with Paraver.

Basic Analysis Tool – Efficiency Table

```
modelfactors.py lulesh2.0_impi_omp-27p.prv
```

Full trace

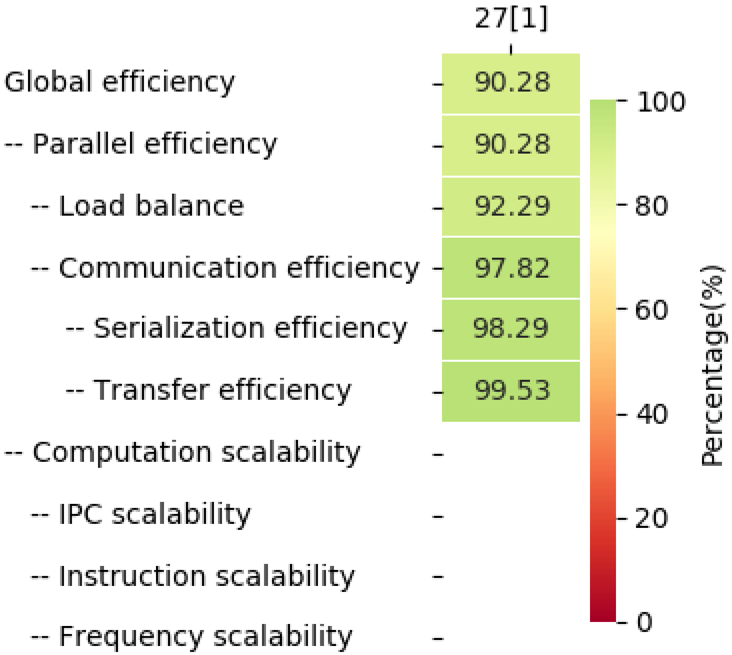


	27[1]
Global efficiency	96.50
-- Parallel efficiency	96.50
-- Load balance	97.54
-- Communication efficiency	98.93
-- Serialization efficiency	99.42
-- Transfer efficiency	99.51
-- Computation scalability	
-- IPC scalability	
-- Instruction scalability	
-- Frequency scalability	

100
80
60
40
20
0
Percentage(%)

It is possible to analyze only the iterative part.

Iterative part



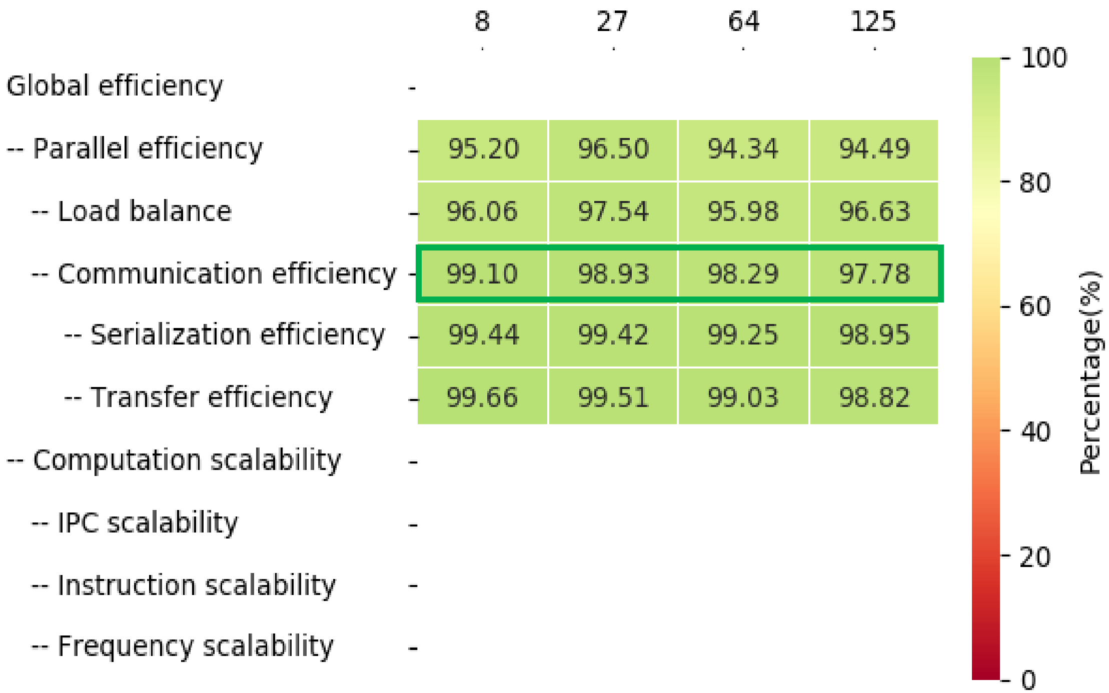
	27[1]
Global efficiency	90.28
-- Parallel efficiency	90.28
-- Load balance	92.29
-- Communication efficiency	97.82
-- Serialization efficiency	98.29
-- Transfer efficiency	99.53
-- Computation scalability	
-- IPC scalability	
-- Instruction scalability	
-- Frequency scalability	

100
80
60
40
20
0
Percentage(%)

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Basic Analysis Tool – Metrics from several traces

```
modelfactors.py lulesh2.0_impi_omp-*p.prv
```

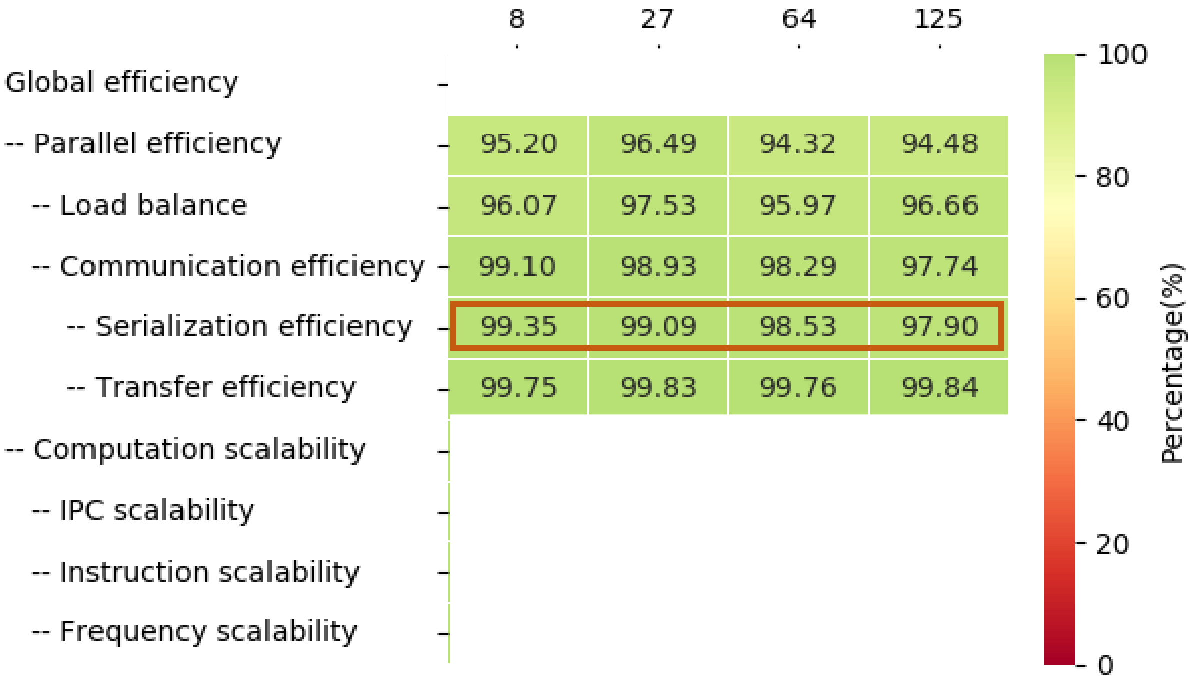


	8	27	64	125
Global efficiency				
-- Parallel efficiency	95.20	96.50	94.34	94.49
-- Load balance	96.06	97.54	95.98	96.63
-- Communication efficiency	99.10	98.93	98.29	97.78
-- Serialization efficiency	99.44	99.42	99.25	98.95
-- Transfer efficiency	99.66	99.51	99.03	98.82
-- Computation scalability				
-- IPC scalability				
-- Instruction scalability				
-- Frequency scalability				

Communication Efficiency seems will be a limiting factor but it is not clear if it serialization or transfer issues.

Basic Analysis Tool – Metrics from several traces (iterative part)

```
modelfactors.py lulesh2.0_impi_omp-*p-chop1.prv
```



	8	27	64	125
Global efficiency				
-- Parallel efficiency	95.20	96.49	94.32	94.48
-- Load balance	96.07	97.53	95.97	96.66
-- Communication efficiency	99.10	98.93	98.29	97.74
-- Serialization efficiency	99.35	99.09	98.53	97.90
-- Transfer efficiency	99.75	99.83	99.76	99.84
-- Computation scalability				
-- IPC scalability				
-- Instruction scalability				
-- Frequency scalability				

Analyzing only the iterative part we can see that at large scale it seems that a limiting factor is the serialization efficiency.

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

EXCELENCIA
SEVERO
OCHOA

Thank you!

Tutorial: Determining Parallel Application Execution Efficiency and Scaling using the POP Methodology

Sandra Mendez

✉ tools@bsc.es

12/05/2024