Installing Paraver & first steps

Germán Llort
(tools@bsc.es)
Barcelona Supercomputing Center

Install Paraver in your laptop

- Download a binary for your OS
  - https://tools.bsc.es/downloads

wxparaver-4.10.0-win.zip

wxparaver-4.10.0-mac.zip

wxparaver-4.10.0-Linux_i686.tar.gz (32bit)
wxparaver-4.10.0-Linux_x86_64.tar.gz (64bit)

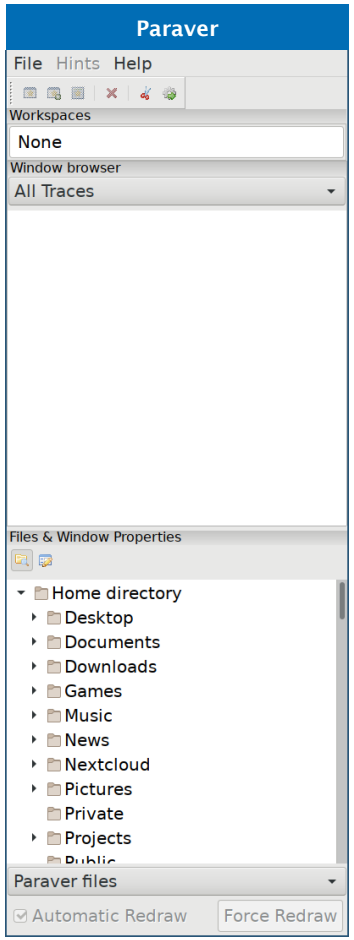
Install Paraver in your laptop

- Start Paraver:
  - Linux:

```
laptop> tar xf wxparaver-4.10.0-Linux_x86_64.tar.bz2
laptop> wxparaver-4.10.0-Linux_x86_64/bin/wxparaver
```

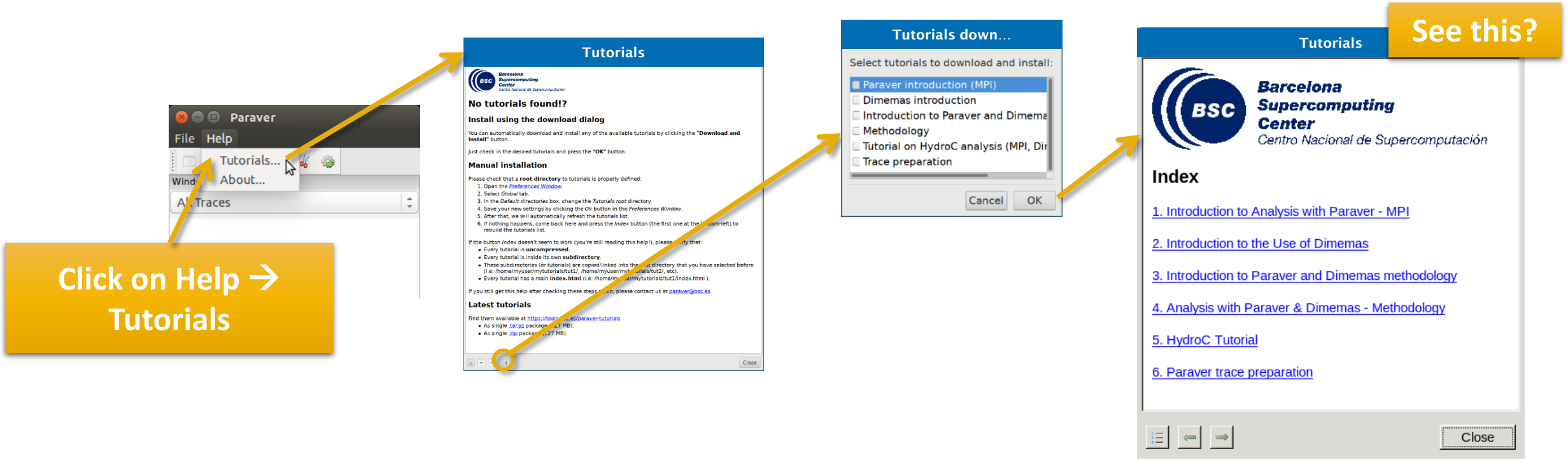
  - Windows: Unzip & double-click on `wxparaver-4.10.0-win/wxparaver.bat`
  - MAC: Unzip & double-click on `wxparaver.app`
  - Any issue? Remotely from JUWELS (`ssh -Y`):

```
juwels> cd /p/project/training2123/tools/paraver/4.10.0
juwels> LC_ALL=C ./bin/wxparaver
```



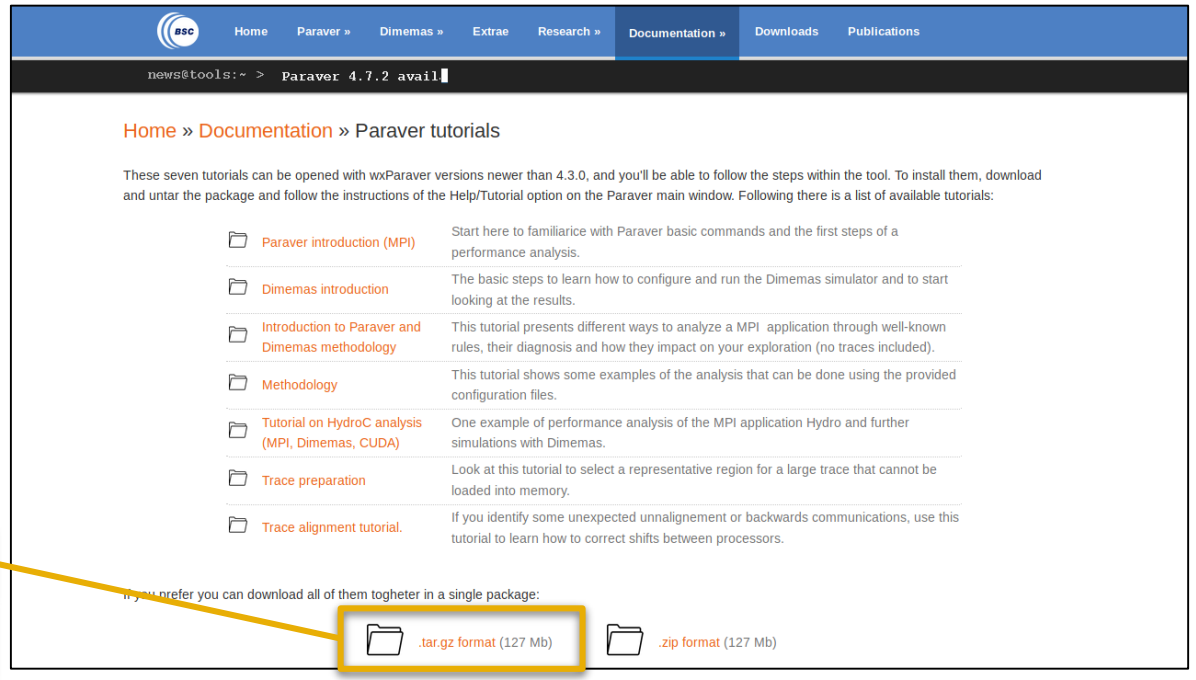
Install Paraver tutorials (Automatic download)

- Download tutorial #3 – Introduction to Paraver and Dimemas methodology



Install Paraver tutorials (Manual install)

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

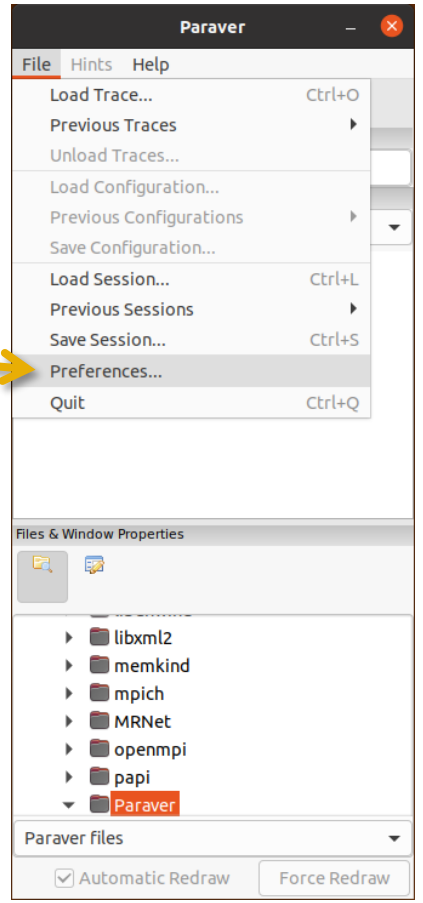


All tutorials

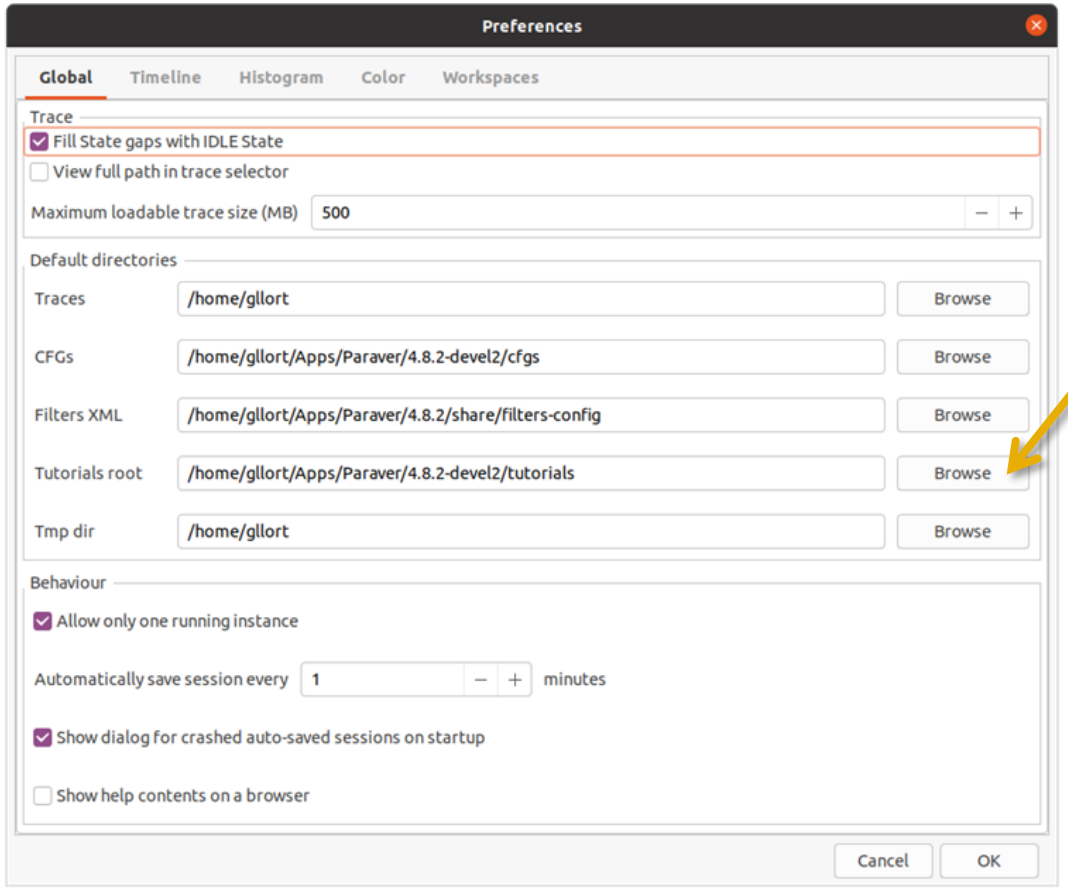
paraver-tutorials-20150526.tar.gz

Install Paraver tutorials (Manual install)

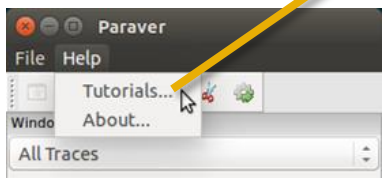
- Uncompress downloaded package
- Rename the folder:
  - paraver-tutorials-20150526 → tutorials
- Open File → Preferences



Install Paraver tutorials (Manual method)



Click on Tutorials root → Browse and point to your folder "tutorials"



See this?

Install CUDA workspace

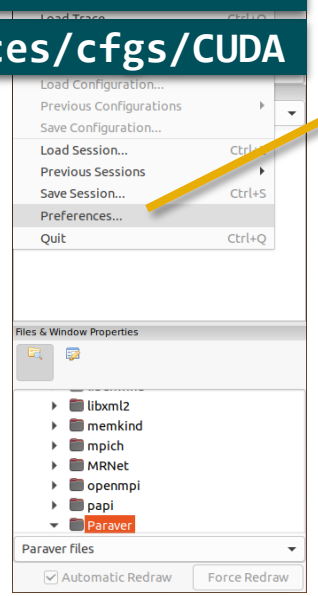
- Download from JUWELS & copy inside your Paraver folder:

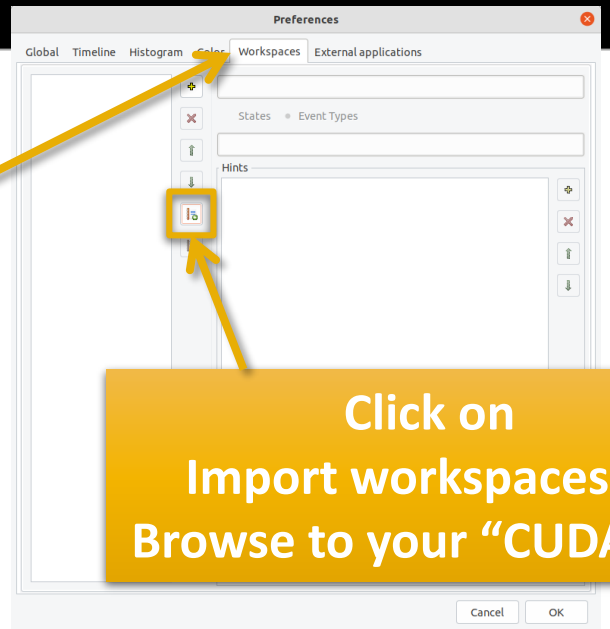
```
laptop> scp <USER>@juwels-booster.fz-juelich.de:/p/project/training2123/work/llort1/bsc-hands-on/ws/CUDA.ws .
laptop> scp <USER>@juwels-booster.fz-juelich.de:/p/project/training2123/work/llort1/bsc-hands-on/cfgs/*.cfg <…>/wxparaver-4.10-0-Linux_x86_64/cfgs/CUDA
```

wxparaver-4.10.0-win/cfgs/CUDA

wxparaver.app/Contents/Resources/cfgs/CUDA

- File →
  - Preferences →
    - Workspaces →
      - Import workspaces →
        - Browse to “CUDA.ws”





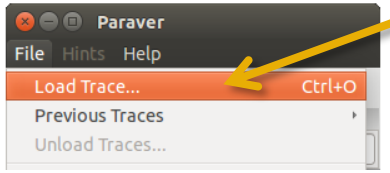
Click on
Import workspaces →
Browse to your “CUDA.ws”

First steps of analysis

- Download sample trace from JUWELS (3 files, pcf, prv, row):

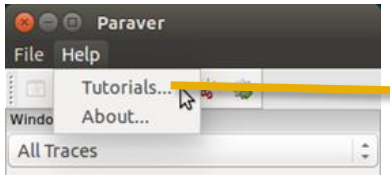
```
laptop> scp <USER>@juwels-booster.fz-juelich.de:/p/project/training2123/work/llort1/bsc-hands-on/traces/lulesh2.0_booster_27p.* .
```

- Load the trace in Paraver:



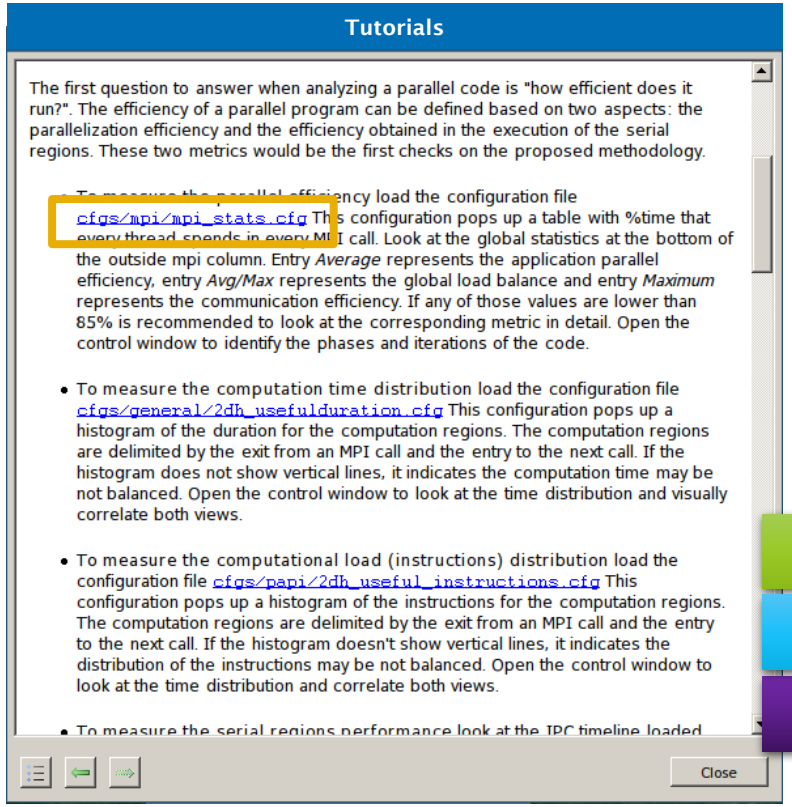
Click on File → Load Trace → Browse to “lulesh2.0_booster_27p.prv”

- Follow Tutorial #3:



Measure the parallel efficiency

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



Parallel efficiency (Avg)

Comm efficiency (Max)

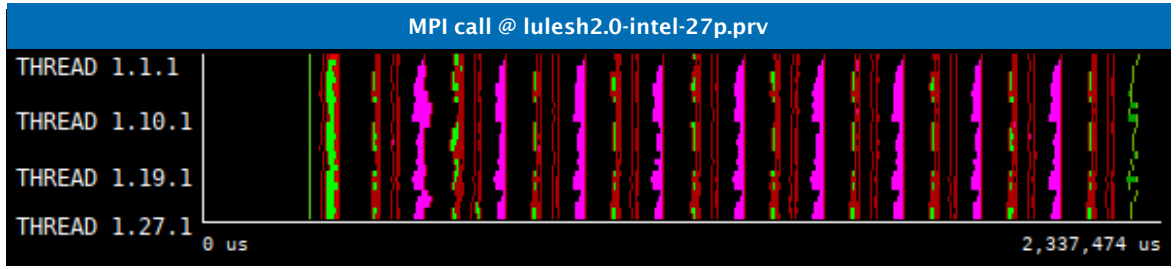
Load balance (Avg/Max)

MPI call profile @ lulesh2.0_booster_27p.prv

Default

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.15.1	84.80 %	0.17 %	0.15 %	0.97 %	6.63 %	0.28 %	0.06 %
THREAD 1.16.1	85.33 %	0.12 %	0.10 %	1.17 %	6.45 %	0.44 %	0.00 %
THREAD 1.17.1	91.74 %	0.17 %	0.16 %	0.77 %	1.50 %	0.12 %	0.27 %
THREAD 1.18.1	89.33 %	0.12 %	0.11 %	0.12 %	1.95 %	0.34 %	0.00 %
THREAD 1.19.1	92.12 %	0.08 %	0.07 %	0.09 %	1.81 %	0.01 %	0.00 %
THREAD 1.20.1	82.52 %	0.12 %	0.10 %	0.21 %	7.21 %	0.32 %	0.00 %
THREAD 1.21.1	80.60 %	0.09 %	0.07 %	3.20 %	3.53 %	0.38 %	0.98 %
THREAD 1.22.1	81.81 %	0.13 %	0.10 %	0.56 %	7.10 %	0.00 %	0.00 %
THREAD 1.23.1	89.49 %	0.18 %	0.14 %	3.98 %	2.71 %	0.32 %	0.00 %
THREAD 1.24.1	86.98 %	0.14 %	0.09 %	3.55 %	3.34 %	0.35 %	0.00 %
THREAD 1.25.1	81.88 %	0.11 %	0.07 %	4.01 %	5.07 %	0.36 %	0.01 %
THREAD 1.26.1	81.47 %	0.15 %	0.09 %	0.30 %	9.63 %	0.12 %	0.01 %
THREAD 1.27.1	80.67 %	0.11 %	0.06 %	0.21 %	9.70 %	0.40 %	0.00 %
Total	2,334.46 %	3.39 %	3.06 %	31.36 %	125.25 %	8.43 %	2.89 %
Average	86.46 %	0.13 %	0.11 %	1.16 %	4.64 %	0.31 %	0.11 %
Maximum	95.83 %	0.24 %	0.23 %	5.09 %	9.70 %	1.02 %	0.98 %
Minimum	80.60 %	0.06 %	0.06 %	0.09 %	1.24 %	0.00 %	0.00 %
StDev	4.22 %	0.04 %	0.04 %	1.42 %	2.63 %	0.19 %	0.25 %
Avg/Max	0.90	0.51	0.49	0.23	0.48	0.31	0.11

Focus on the iterative part



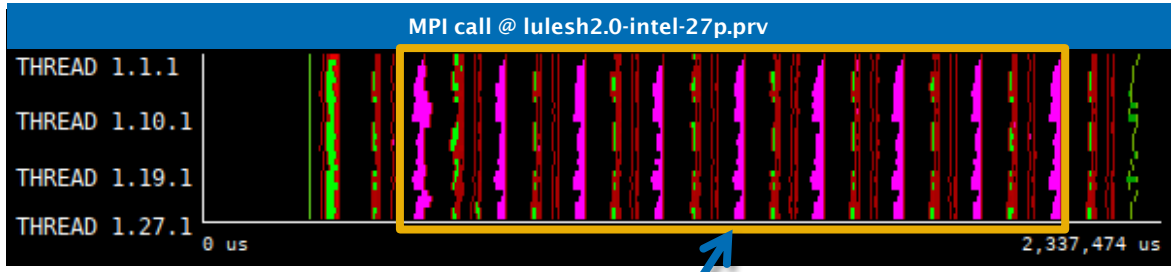
Click on
“Open Control Window”

MPI call profile @ lulesh2.0_booster_27p.prv

Default

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.15.1	84.80 %	0.17 %	0.15 %	0.97 %	6.63 %	0.28 %	0.06 %
THREAD 1.16.1	85.33 %	0.12 %	0.10 %	1.17 %	6.45 %	0.44 %	0.00 %
THREAD 1.17.1	91.74 %	0.17 %	0.16 %	0.77 %	1.50 %	0.12 %	0.27 %
THREAD 1.18.1	89.33 %	0.12 %	0.11 %	0.12 %	1.95 %	0.34 %	0.00 %
THREAD 1.19.1	92.12 %	0.08 %	0.07 %	0.09 %	1.81 %	0.01 %	0.00 %
THREAD 1.20.1	82.52 %	0.12 %	0.10 %	0.21 %	7.21 %	0.32 %	0.00 %
THREAD 1.21.1	80.60 %	0.09 %	0.07 %	3.20 %	3.53 %	0.38 %	0.98 %
THREAD 1.22.1	81.81 %	0.13 %	0.10 %	0.56 %	7.10 %	0.00 %	0.00 %
THREAD 1.23.1	89.49 %	0.18 %	0.14 %	3.98 %	2.71 %	0.32 %	0.00 %
THREAD 1.24.1	86.98 %	0.14 %	0.09 %	3.55 %	3.34 %	0.35 %	0.00 %
THREAD 1.25.1	81.88 %	0.11 %	0.07 %	4.01 %	5.07 %	0.36 %	0.01 %
THREAD 1.26.1	81.47 %	0.15 %	0.09 %	0.30 %	9.63 %	0.12 %	0.01 %
THREAD 1.27.1	80.67 %	0.11 %	0.06 %	0.21 %	9.70 %	0.40 %	0.00 %
Total	2,334.46 %	3.39 %	3.06 %	31.36 %	125.25 %	8.43 %	2.89 %
Average	86.46 %	0.13 %	0.11 %	1.16 %	4.64 %	0.31 %	0.11 %
Maximum	95.83 %	0.24 %	0.23 %	5.09 %	9.70 %	1.02 %	0.98 %
Minimum	80.60 %	0.06 %	0.06 %	0.09 %	1.24 %	0.00 %	0.00 %
StDev	4.22 %	0.04 %	0.04 %	1.42 %	2.63 %	0.19 %	0.25 %
Avg/Max	0.90	0.51	0.49	0.23	0.48	0.31	0.11

Focus on the iterative part



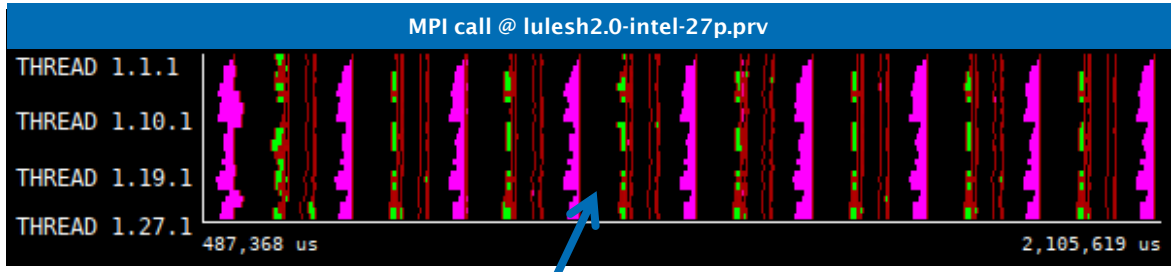
Drag & drop on this area
to zoom on the iterative region

MPI call profile @ lulesh2.0_booster_27p.prv

Default

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.15.1	84.80 %	0.17 %	0.15 %	0.97 %	6.63 %	0.28 %	0.06 %
THREAD 1.16.1	85.33 %	0.12 %	0.10 %	1.17 %	6.45 %	0.44 %	0.00 %
THREAD 1.17.1	91.74 %	0.17 %	0.16 %	0.77 %	1.50 %	0.12 %	0.27 %
THREAD 1.18.1	89.33 %	0.12 %	0.11 %	0.12 %	1.95 %	0.34 %	0.00 %
THREAD 1.19.1	92.12 %	0.08 %	0.07 %	0.09 %	1.81 %	0.01 %	0.00 %
THREAD 1.20.1	82.52 %	0.12 %	0.10 %	0.21 %	7.21 %	0.32 %	0.00 %
THREAD 1.21.1	80.60 %	0.09 %	0.07 %	3.20 %	3.53 %	0.38 %	0.98 %
THREAD 1.22.1	81.81 %	0.13 %	0.10 %	0.56 %	7.10 %	0.00 %	0.00 %
THREAD 1.23.1	89.49 %	0.18 %	0.14 %	3.98 %	2.71 %	0.32 %	0.00 %
THREAD 1.24.1	86.98 %	0.14 %	0.09 %	3.55 %	3.34 %	0.35 %	0.00 %
THREAD 1.25.1	81.88 %	0.11 %	0.07 %	4.01 %	5.07 %	0.36 %	0.01 %
THREAD 1.26.1	81.47 %	0.15 %	0.09 %	0.30 %	9.63 %	0.12 %	0.01 %
THREAD 1.27.1	80.67 %	0.11 %	0.06 %	0.21 %	9.70 %	0.40 %	0.00 %
Total	2,334.46 %	3.39 %	3.06 %	31.36 %	125.25 %	8.43 %	2.89 %
Average	86.46 %	0.13 %	0.11 %	1.16 %	4.64 %	0.31 %	0.11 %
Maximum	95.83 %	0.24 %	0.23 %	5.09 %	9.70 %	1.02 %	0.98 %
Minimum	80.60 %	0.06 %	0.06 %	0.09 %	1.24 %	0.00 %	0.00 %
StDev	4.22 %	0.04 %	0.04 %	1.42 %	2.63 %	0.19 %	0.25 %
Avg/Max	0.90	0.51	0.49	0.23	0.48	0.31	0.11

Recalculate efficiency of iterative region



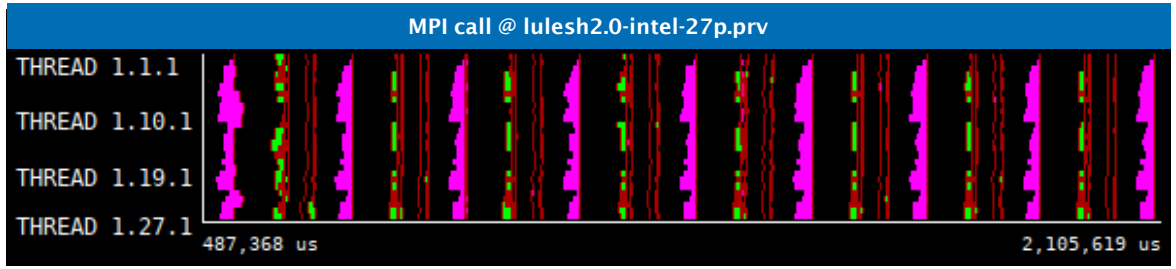
Right click → Copy

MPI call profile @ lulesh2.0_booster_27p.prv

Default

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce
THREAD 1.15.1	84.80 %	0.17 %	0.15 %	0.97 %	6.63 %	0.28 %	0.06 %
THREAD 1.16.1	85.33 %	0.12 %	0.10 %	1.17 %	6.45 %	0.44 %	0.00 %
THREAD 1.17.1	91.74 %	0.17 %	0.16 %	0.77 %	1.50 %	0.12 %	0.27 %
THREAD 1.18.1	89.33 %	0.12 %	0.11 %	0.12 %	1.95 %	0.34 %	0.00 %
THREAD 1.19.1	92.12 %	0.08 %	0.07 %	0.09 %	1.81 %	0.01 %	0.00 %
THREAD 1.20.1	82.52 %	0.12 %	0.10 %	0.21 %	7.21 %	0.32 %	0.00 %
THREAD 1.21.1	80.60 %	0.09 %	0.07 %	3.20 %	3.53 %	0.38 %	0.98 %
THREAD 1.22.1	81.81 %	0.13 %	0.10 %	0.56 %	7.10 %	0.00 %	0.00 %
THREAD 1.23.1	89.49 %	0.18 %	0.14 %	3.98 %	2.71 %	0.32 %	0.00 %
THREAD 1.24.1	86.98 %	0.14 %	0.09 %	3.55 %	3.34 %	0.35 %	0.00 %
THREAD 1.25.1	81.88 %	0.11 %	0.07 %	4.01 %	5.07 %	0.36 %	0.01 %
THREAD 1.26.1	81.47 %	0.15 %	0.09 %	0.30 %	9.63 %	0.12 %	0.01 %
THREAD 1.27.1	80.67 %	0.11 %	0.06 %	0.21 %	9.70 %	0.40 %	0.00 %
Total	2,334.46 %	3.39 %	3.06 %	31.36 %	125.25 %	8.43 %	2.89 %
Average	86.46 %	0.13 %	0.11 %	1.16 %	4.64 %	0.31 %	0.11 %
Maximum	95.83 %	0.24 %	0.23 %	5.09 %	9.70 %	1.02 %	0.98 %
Minimum	80.60 %	0.06 %	0.06 %	0.09 %	1.24 %	0.00 %	0.00 %
StDev	4.22 %	0.04 %	0.04 %	1.42 %	2.63 %	0.19 %	0.25 %
Avg/Max	0.90	0.51	0.49	0.23	0.48	0.31	0.11

Recalculate efficiency of iterative region



Right click → Paste → Time

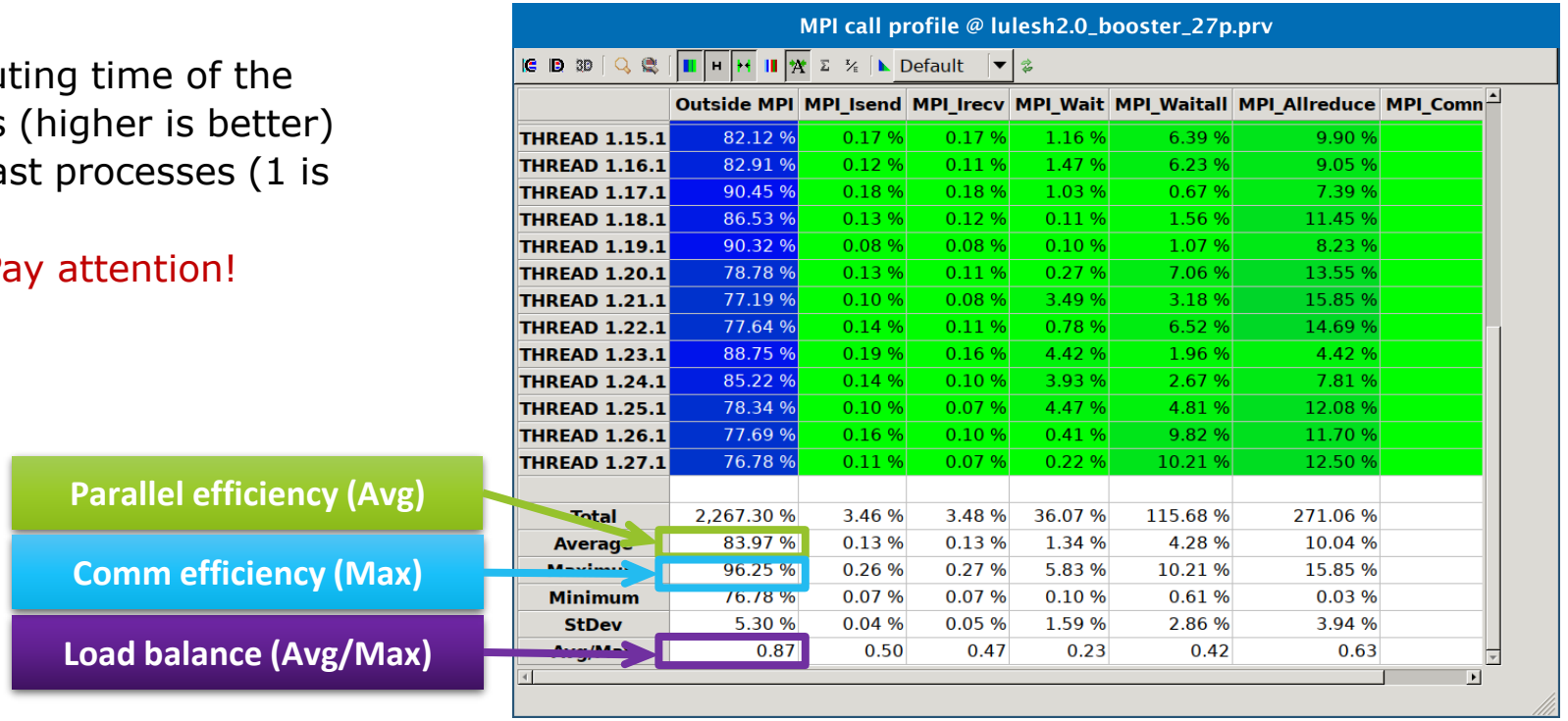
MPI call profile @ lulesh2.0_booster_27p.prv

Default

	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Allreduce	MPI_Comm
THREAD 1.15.1	82.12 %	0.17 %	0.17 %	1.16 %	6.39 %	9.90 %	
THREAD 1.16.1	82.91 %	0.12 %	0.11 %	1.47 %	6.23 %	9.05 %	
THREAD 1.17.1	90.45 %	0.18 %	0.18 %	1.03 %	0.67 %	7.39 %	
THREAD 1.18.1	86.53 %	0.13 %	0.12 %	0.11 %	1.56 %	11.45 %	
THREAD 1.19.1	90.32 %	0.08 %	0.08 %	0.10 %	1.07 %	8.23 %	
THREAD 1.20.1	78.78 %	0.13 %	0.11 %	0.27 %	7.06 %	13.55 %	
THREAD 1.21.1	77.19 %	0.10 %	0.08 %	3.49 %	3.18 %	15.85 %	
THREAD 1.22.1	77.64 %	0.14 %	0.11 %	0.78 %	6.52 %	14.69 %	
THREAD 1.23.1	88.75 %	0.19 %	0.16 %	4.42 %	1.96 %	4.42 %	
THREAD 1.24.1	85.22 %	0.14 %	0.10 %	3.93 %	2.67 %	7.81 %	
THREAD 1.25.1	78.34 %	0.10 %	0.07 %	4.47 %	4.81 %	12.08 %	
THREAD 1.26.1	77.69 %	0.16 %	0.10 %	0.41 %	9.82 %	11.70 %	
THREAD 1.27.1	76.78 %	0.11 %	0.07 %	0.22 %	10.21 %	12.50 %	
Total	2,267.30 %	3.46 %	3.48 %	36.07 %	115.68 %	271.06 %	
Average	83.97 %	0.13 %	0.13 %	1.34 %	4.28 %	10.04 %	
Maximum	96.25 %	0.26 %	0.27 %	5.83 %	10.21 %	15.85 %	
Minimum	76.78 %	0.07 %	0.07 %	0.10 %	0.61 %	0.03 %	
StDev	5.30 %	0.04 %	0.05 %	1.59 %	2.86 %	3.94 %	
Avg/Max	0.87	0.50	0.47	0.23	0.42	0.63	

Efficiency of iterative region

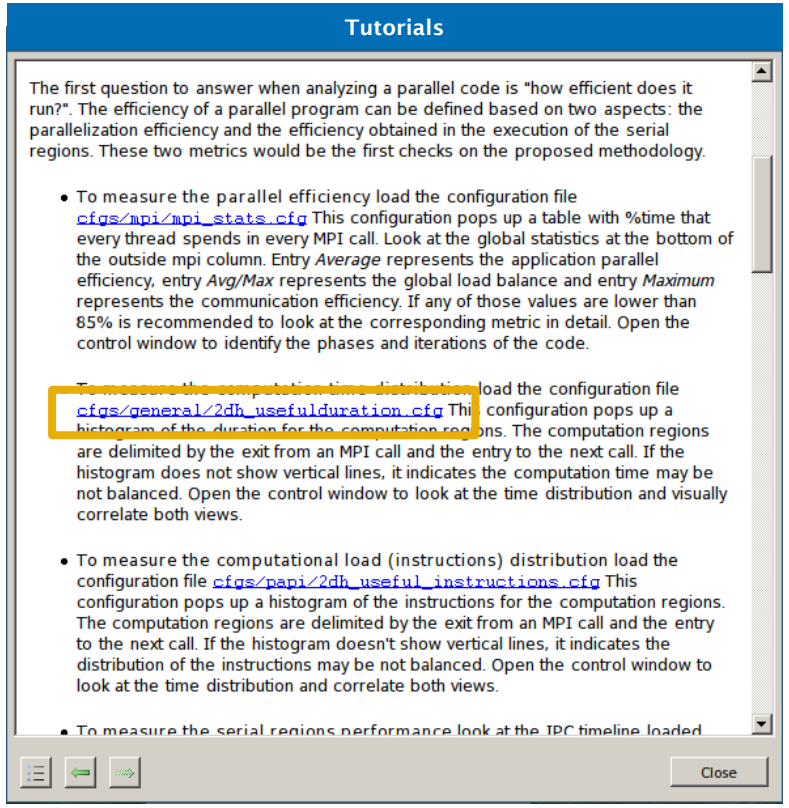
- 3 numbers to quickly describe the efficiency of your code
  - Parallel efficiency → % of time my program is computing (higher is better)
  - Comm efficiency → % of computing time of the process that communicates less (higher is better)
  - Load balance → Ratio of slow/fast processes (1 is perfectly balanced)
  - Any value below 85% (0.85)? Pay attention!

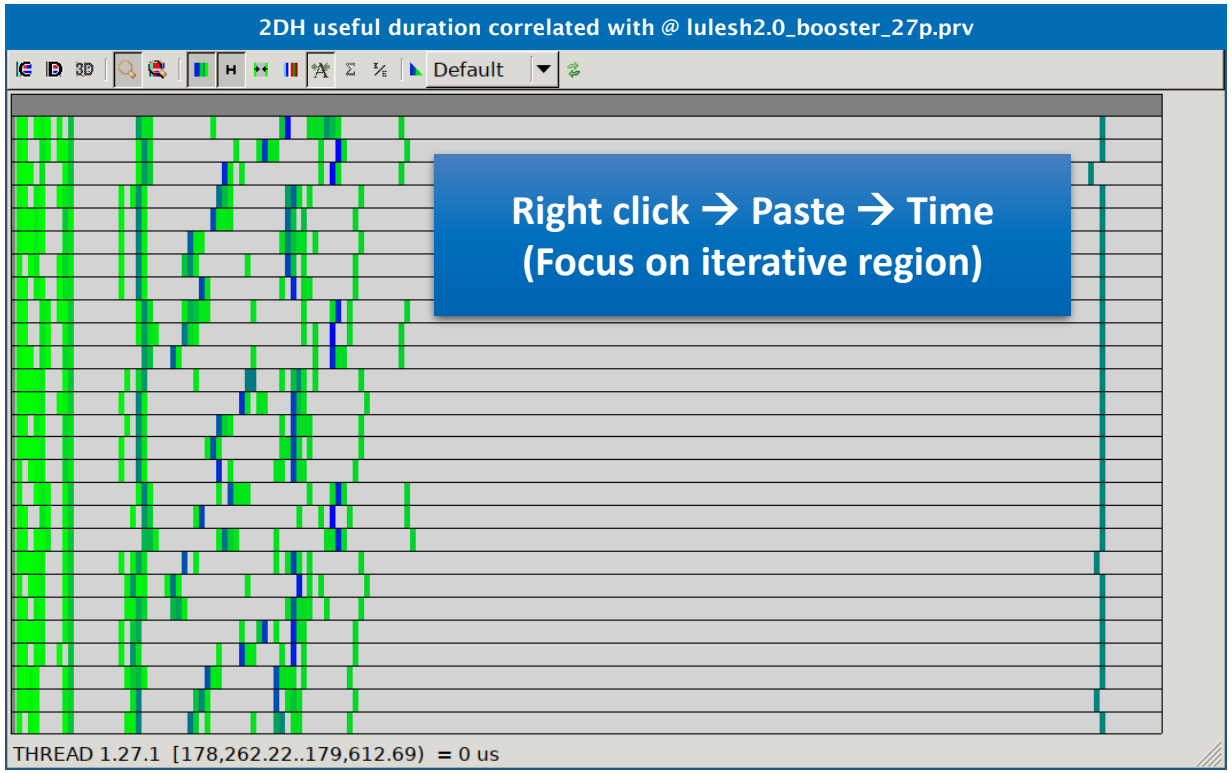


	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Allreduce	MPI_Comm
THREAD 1.15.1	82.12 %	0.17 %	0.17 %	1.16 %	6.39 %	9.90 %	
THREAD 1.16.1	82.91 %	0.12 %	0.11 %	1.47 %	6.23 %	9.05 %	
THREAD 1.17.1	90.45 %	0.18 %	0.18 %	1.03 %	0.67 %	7.39 %	
THREAD 1.18.1	86.53 %	0.13 %	0.12 %	0.11 %	1.56 %	11.45 %	
THREAD 1.19.1	90.32 %	0.08 %	0.08 %	0.10 %	1.07 %	8.23 %	
THREAD 1.20.1	78.78 %	0.13 %	0.11 %	0.27 %	7.06 %	13.55 %	
THREAD 1.21.1	77.19 %	0.10 %	0.08 %	3.49 %	3.18 %	15.85 %	
THREAD 1.22.1	77.64 %	0.14 %	0.11 %	0.78 %	6.52 %	14.69 %	
THREAD 1.23.1	88.75 %	0.19 %	0.16 %	4.42 %	1.96 %	4.42 %	
THREAD 1.24.1	85.22 %	0.14 %	0.10 %	3.93 %	2.67 %	7.81 %	
THREAD 1.25.1	78.34 %	0.10 %	0.07 %	4.47 %	4.81 %	12.08 %	
THREAD 1.26.1	77.69 %	0.16 %	0.10 %	0.41 %	9.82 %	11.70 %	
THREAD 1.27.1	76.78 %	0.11 %	0.07 %	0.22 %	10.21 %	12.50 %	
Total	2,267.30 %	3.46 %	3.48 %	36.07 %	115.68 %	271.06 %	
Average	83.97 %	0.13 %	0.13 %	1.34 %	4.28 %	10.04 %	
Maximum	96.25 %	0.26 %	0.27 %	5.83 %	10.21 %	15.85 %	
Minimum	76.78 %	0.07 %	0.07 %	0.10 %	0.61 %	0.03 %	
StDev	5.30 %	0.04 %	0.05 %	1.59 %	2.86 %	3.94 %	
Avg/Max	0.87	0.50	0.47	0.23	0.42	0.63	

Computation time distribution

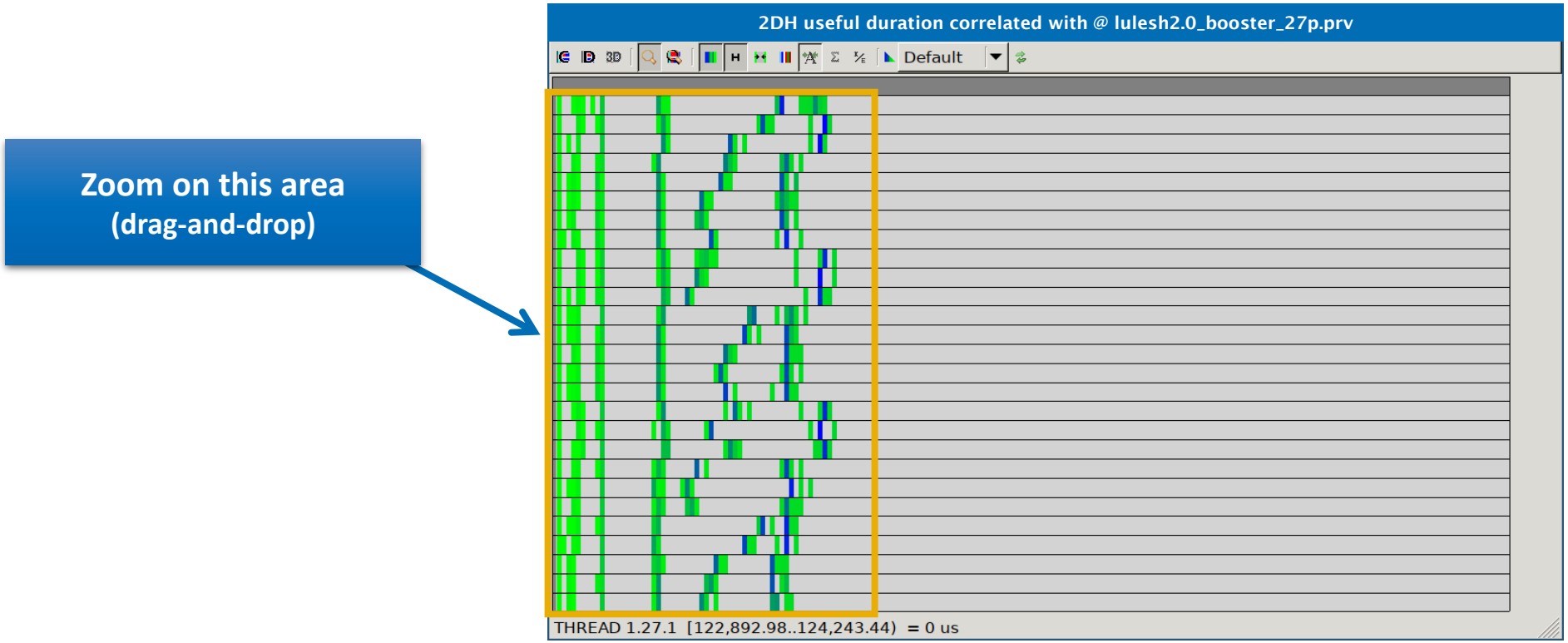
- Click on "2dh_usefulduration.cfg" (2nd link) → Shows **time computing**





Focus on the iterative part

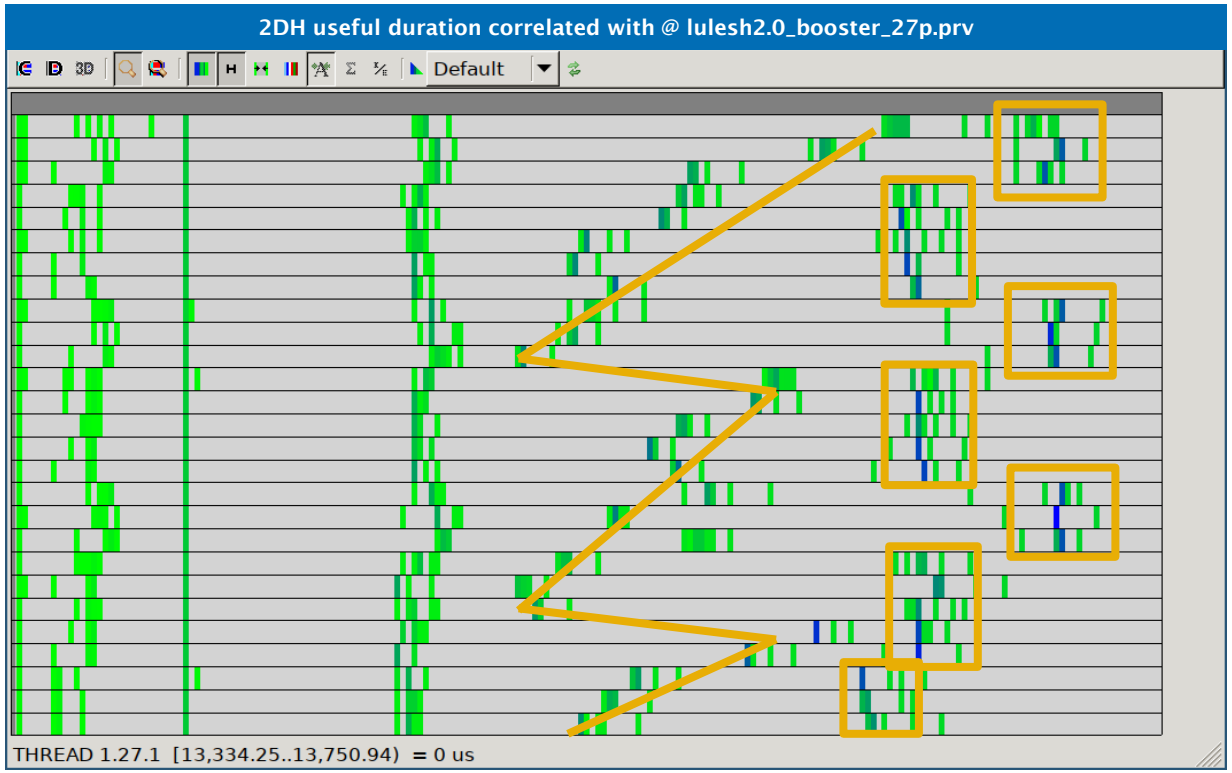
- Click on "2dh_usefulduration.cfg" (2nd link) → Shows **time computing**



Computation time distribution

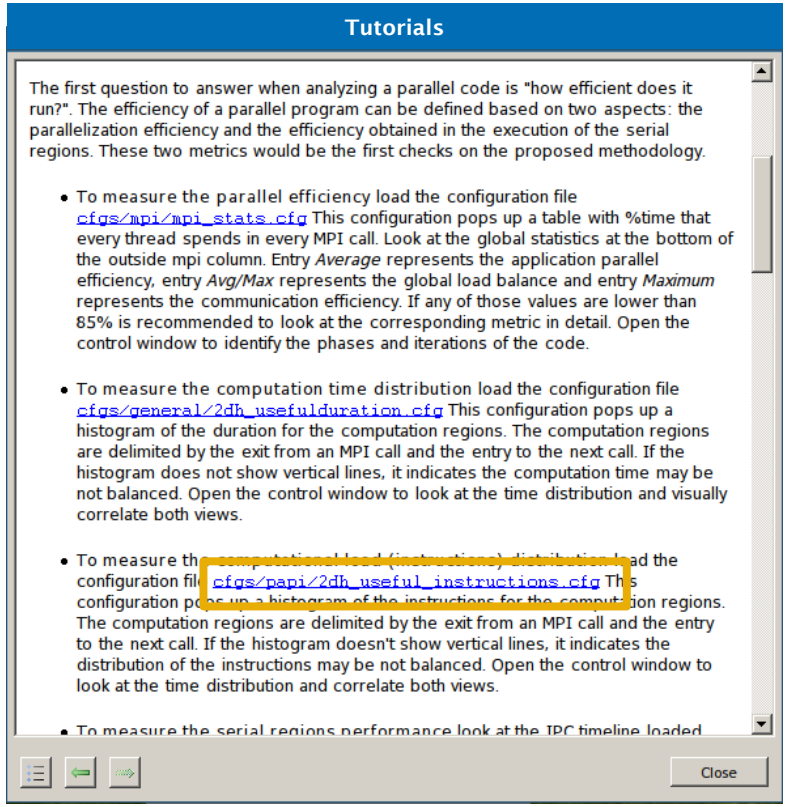
- Click on "2dh_usefulduration.cfg" (2nd link) → Shows **time computing**

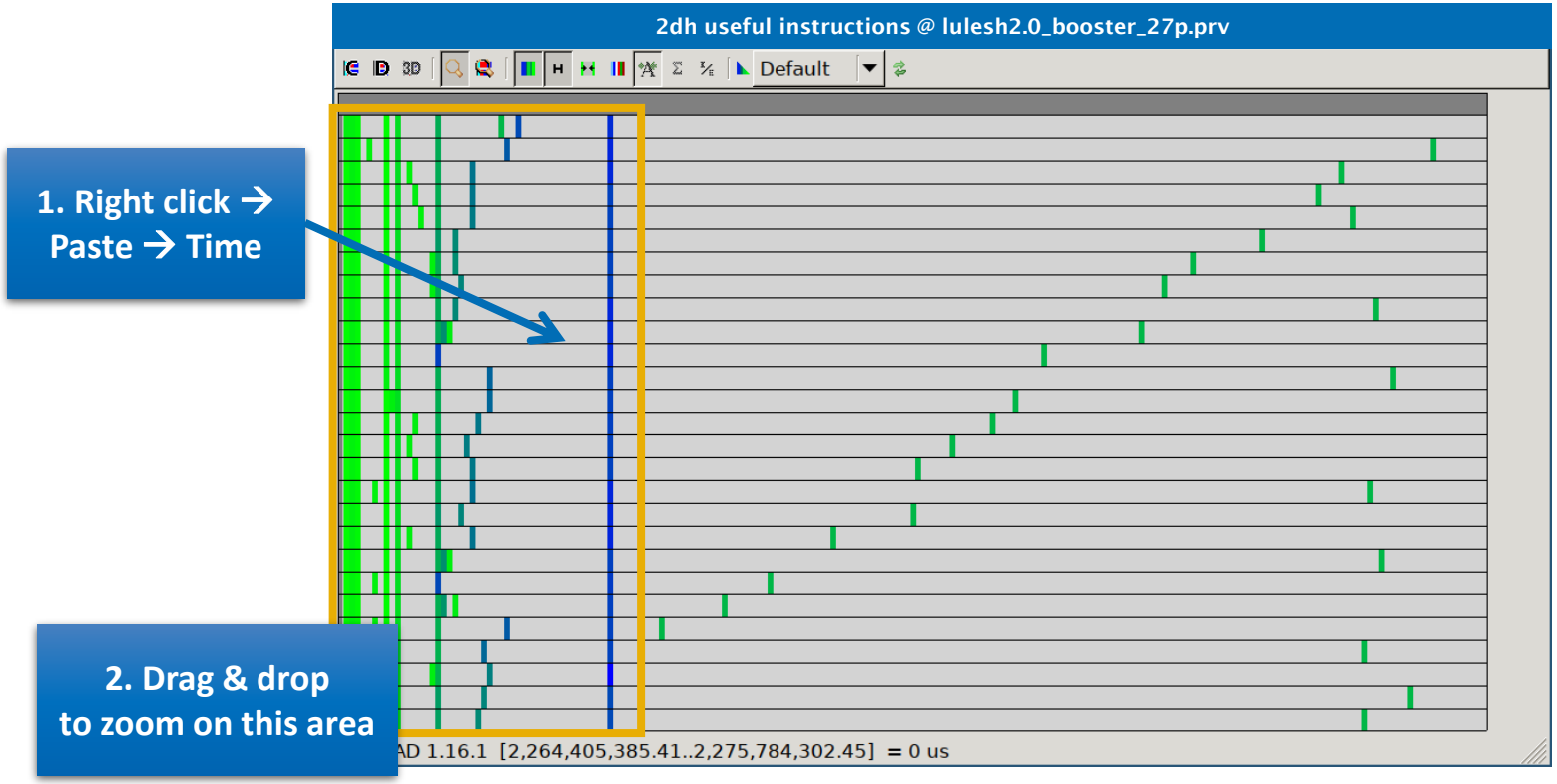
Duration imbalance (zigzag = some processes are taking more time than others)



Computation load distribution

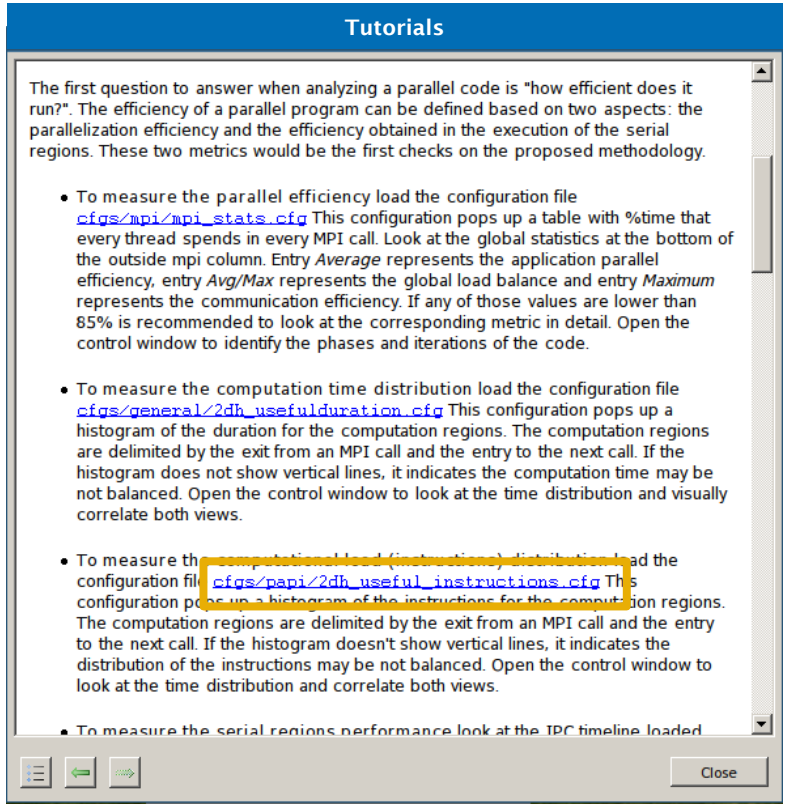
- Click on "2dh_useful_instructions.cfg" (3rd link) → Shows **amount of work**

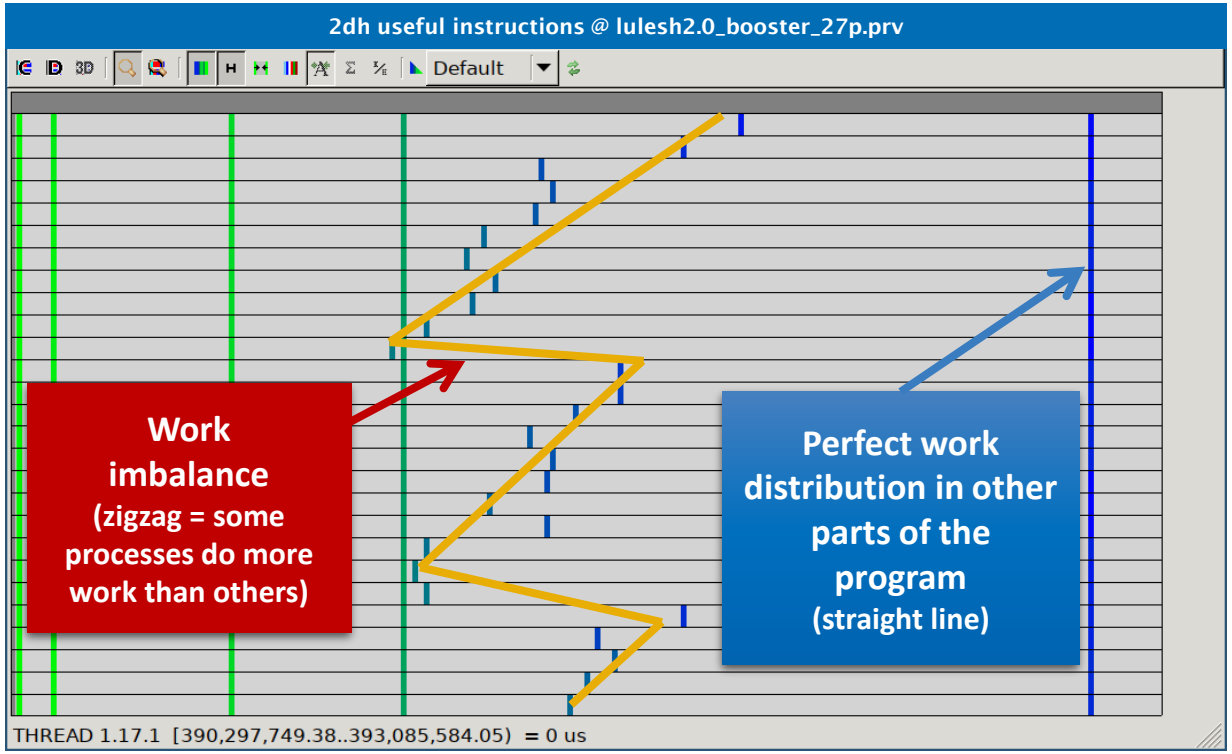




Computation load distribution

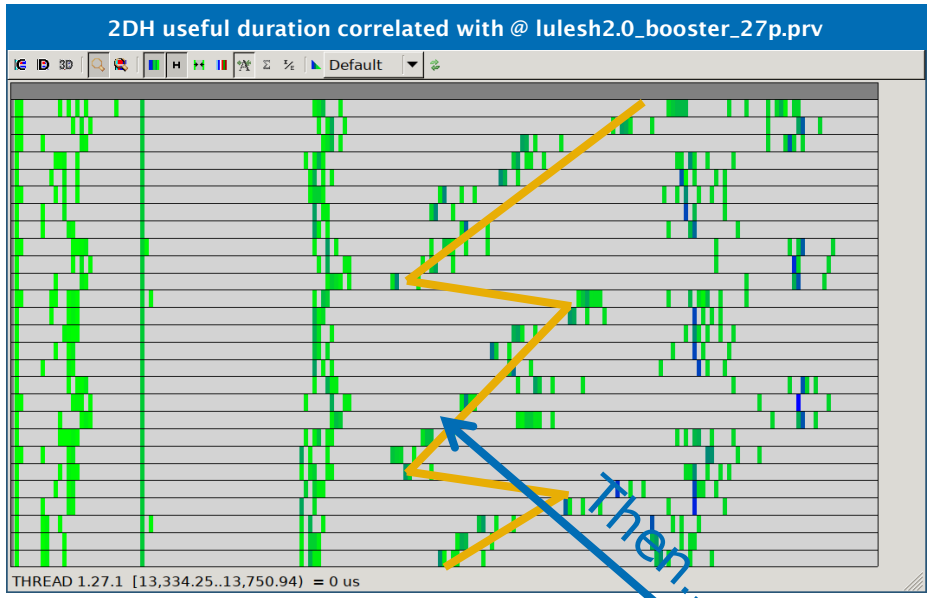
- Click on "2dh_useful_instructions.cfg" (3rd link) → Shows **amount of work**





Computation load distribution

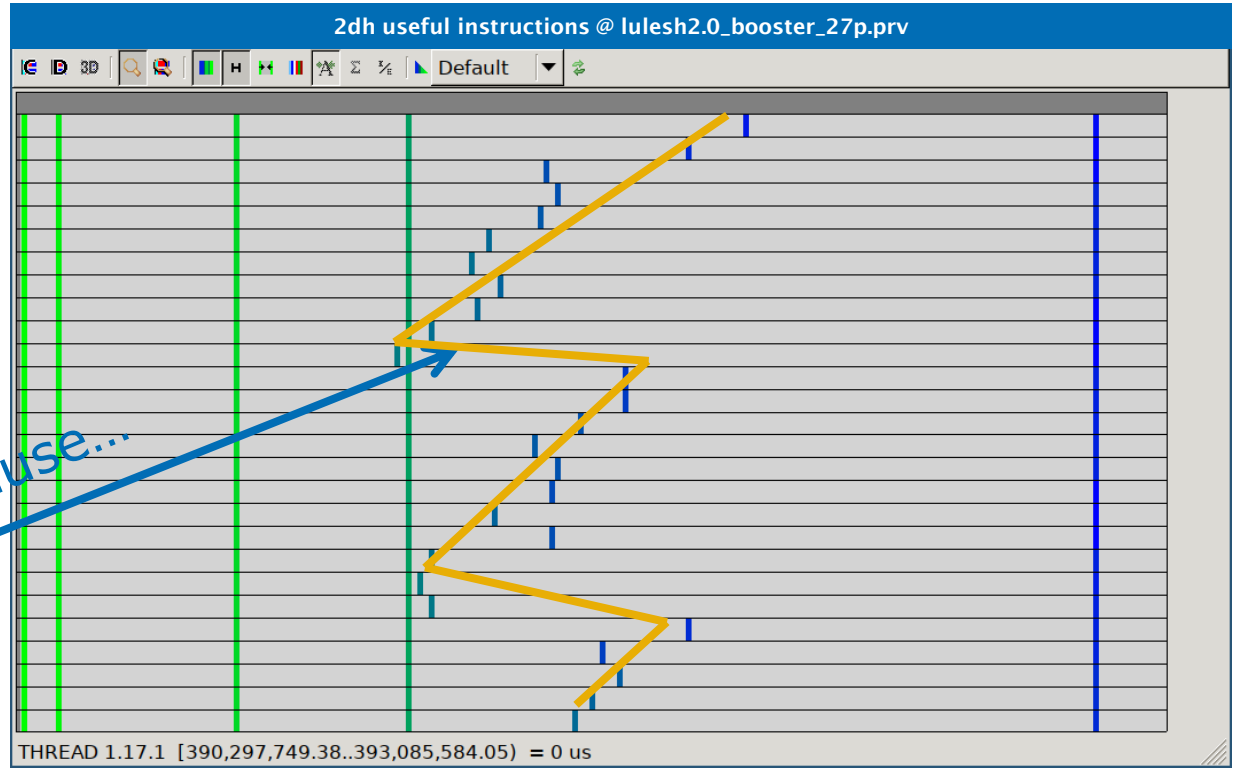
- Correlate the two histograms → **Similar shapes** → Work distribution determines time computing



Then...

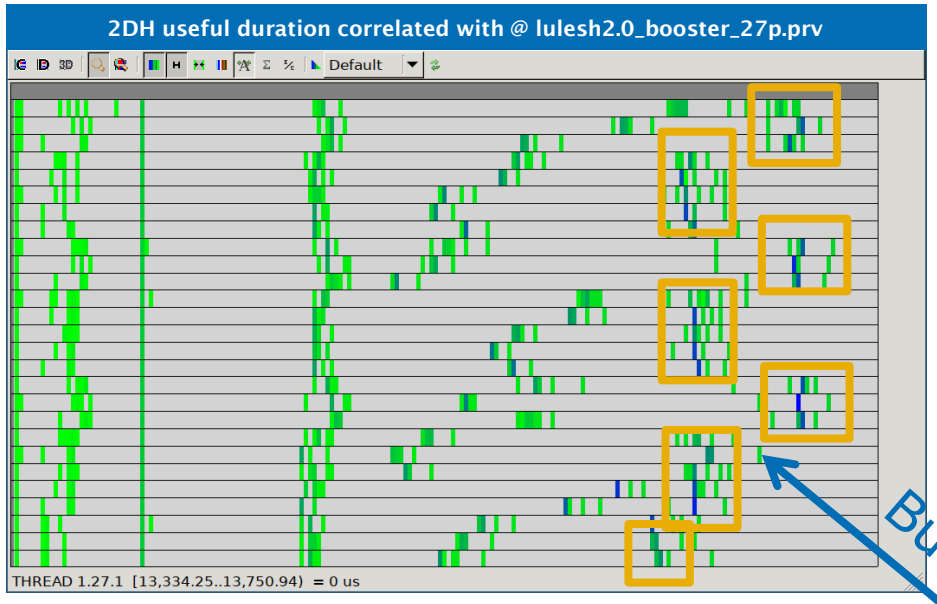
Because...

Work imbalance → Slow/fast processes

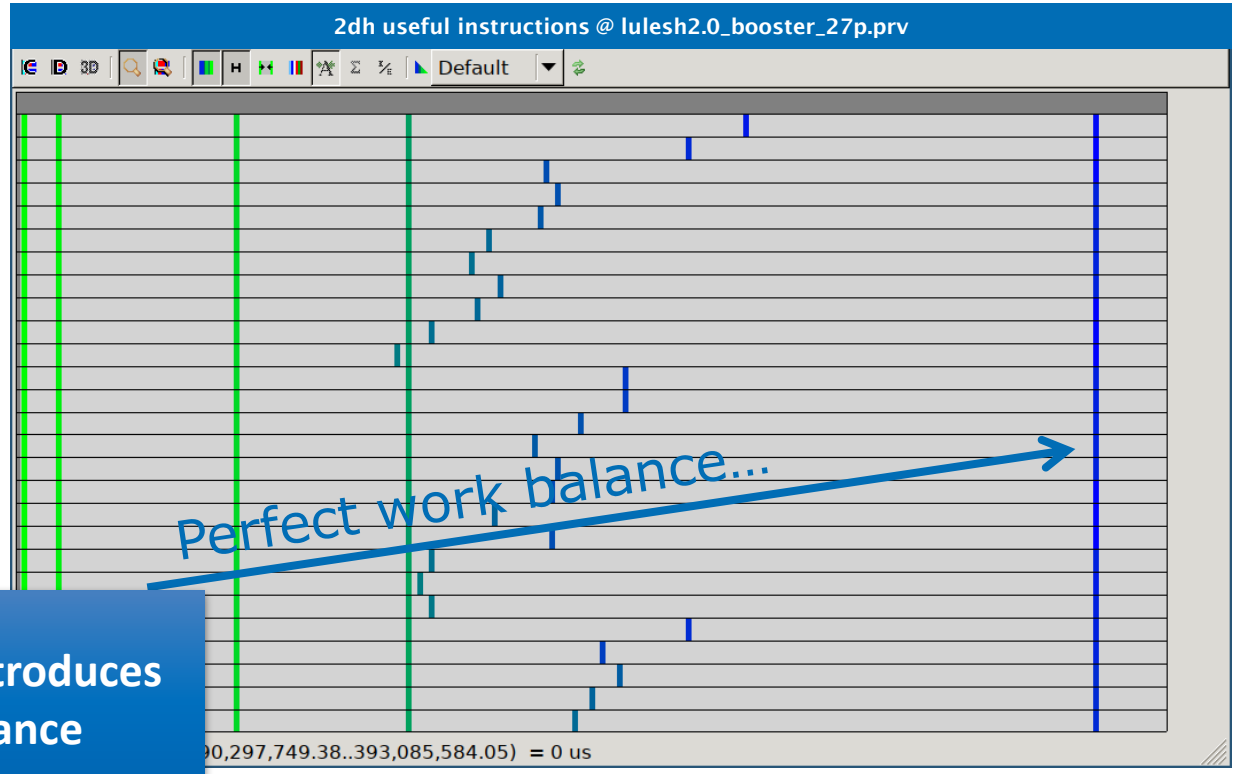


Computation load distribution

- Correlate the two histograms → **Different shapes**



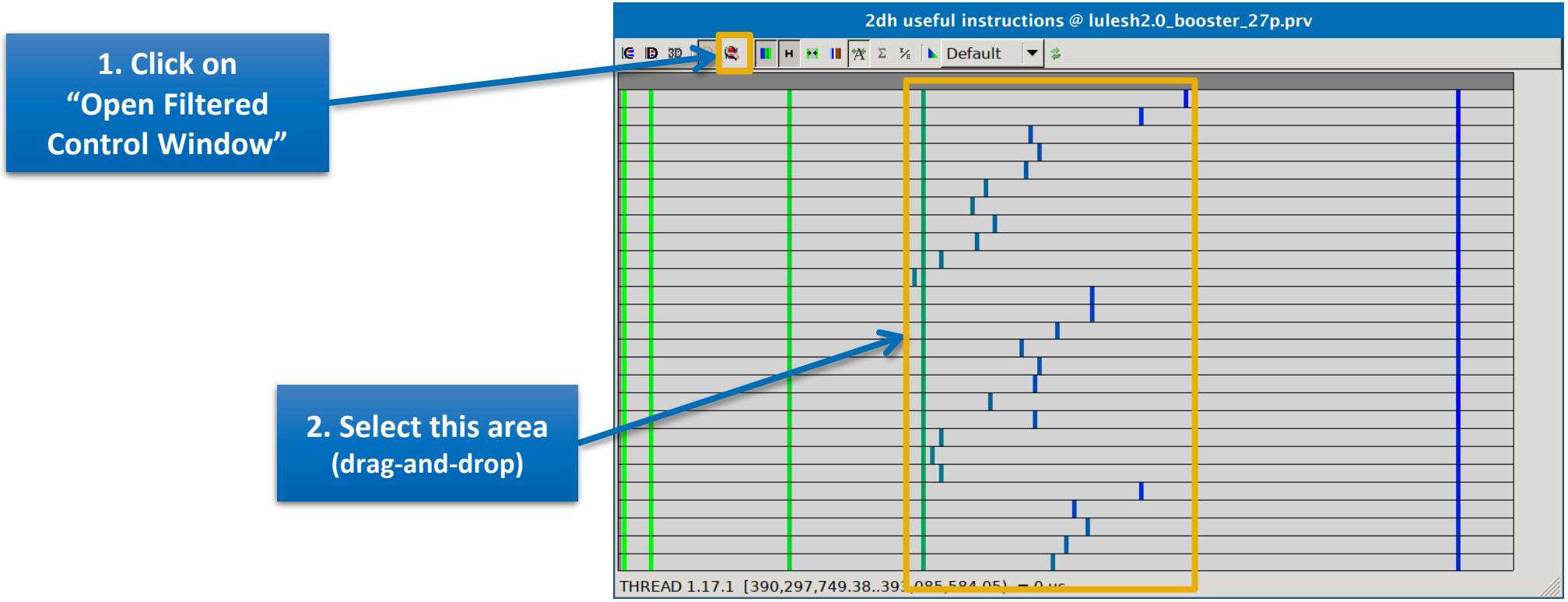
But...



System introduces imbalance

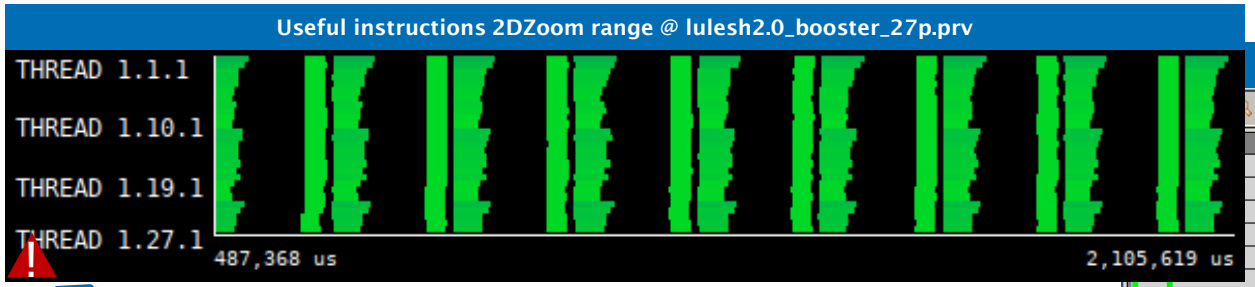
Where does this happen?

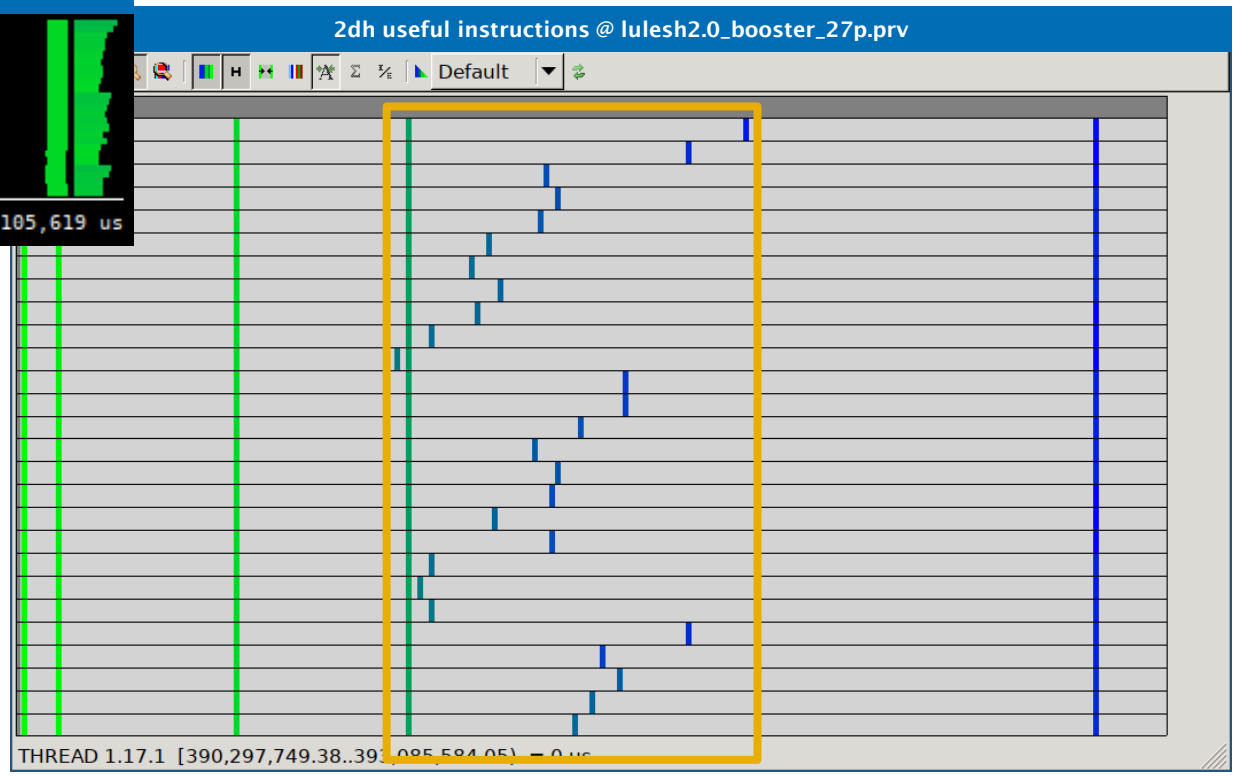
- Go from the table to the timeline



Where does this happen?

- Go from the table to the timeline

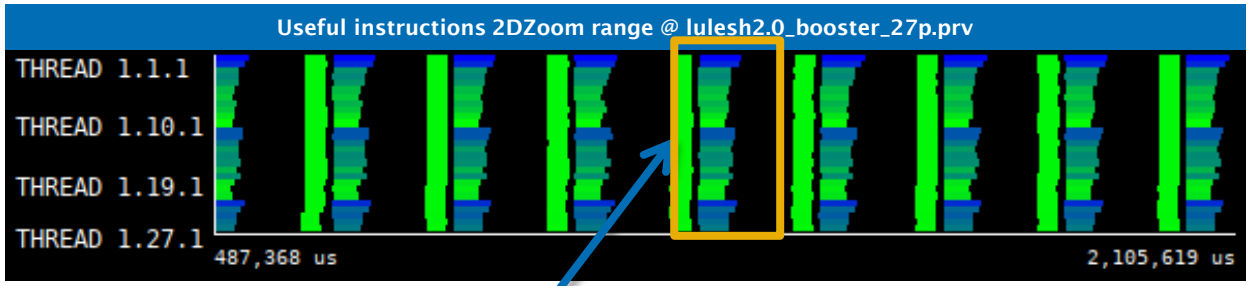




Clicking here rescales.
Same as…
Right click →
Fit Semantic Scale → Fit Both

Where does this happen?

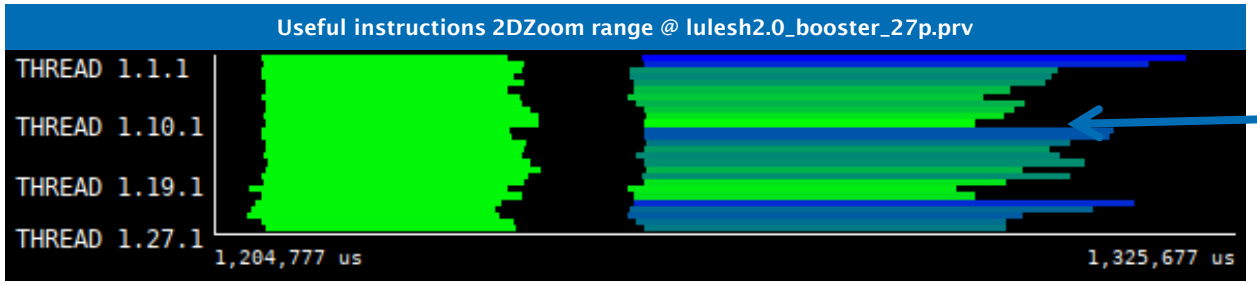
- **Slow** & **Fast** at the same time? → Imbalance paid later in MPI comms



Zoom into
1 of the iterations
(drag-and-drop)

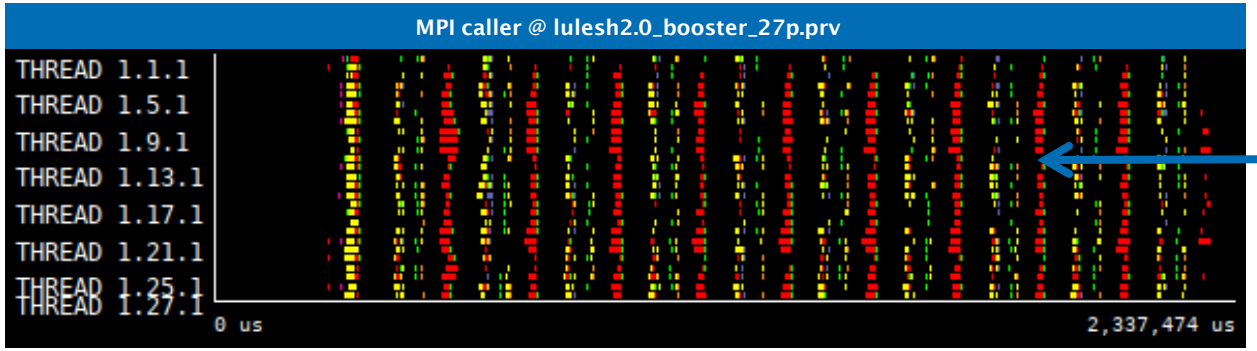
Where does this happen?

- **Slow** & **Fast** at the same time? → Imbalance paid later in MPI comms



1. Right click → Copy

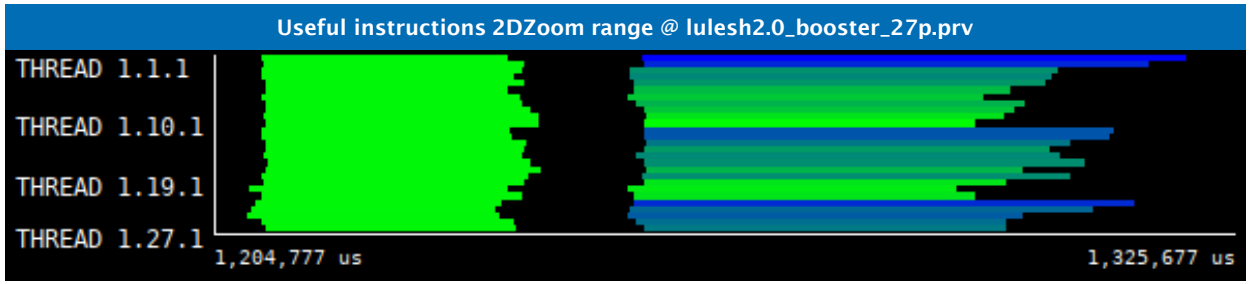
- **Hints → Call stack references → Caller function**



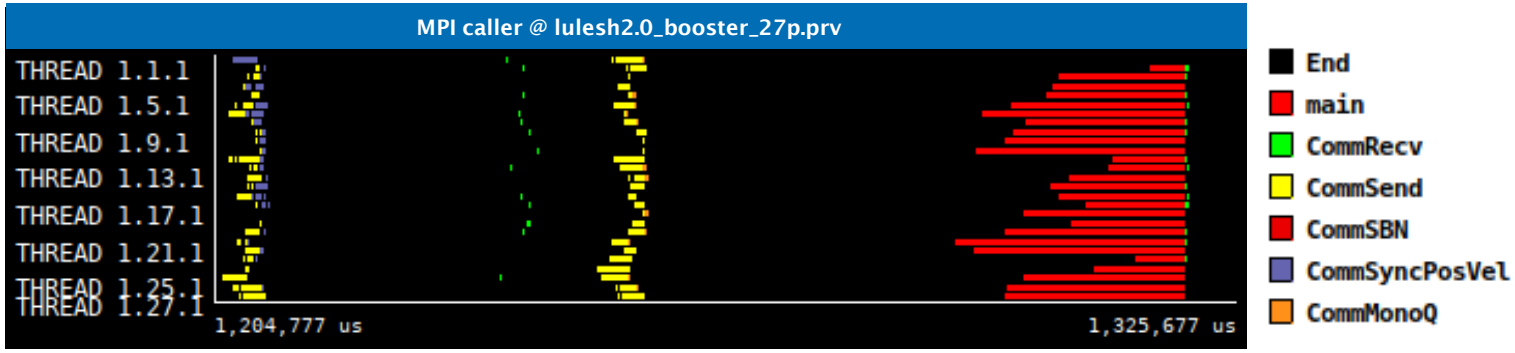
2. Right click → Paste → Time

Where does this happen?

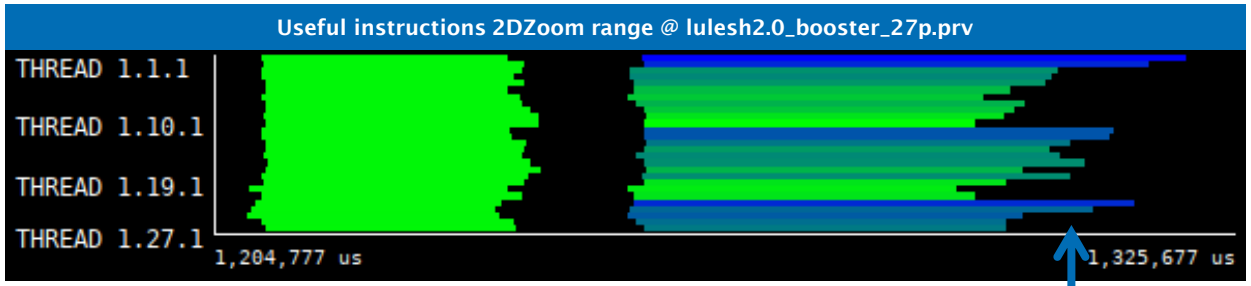
- **Slow** & **Fast** at the same time? → Imbalance paid later in MPI comms



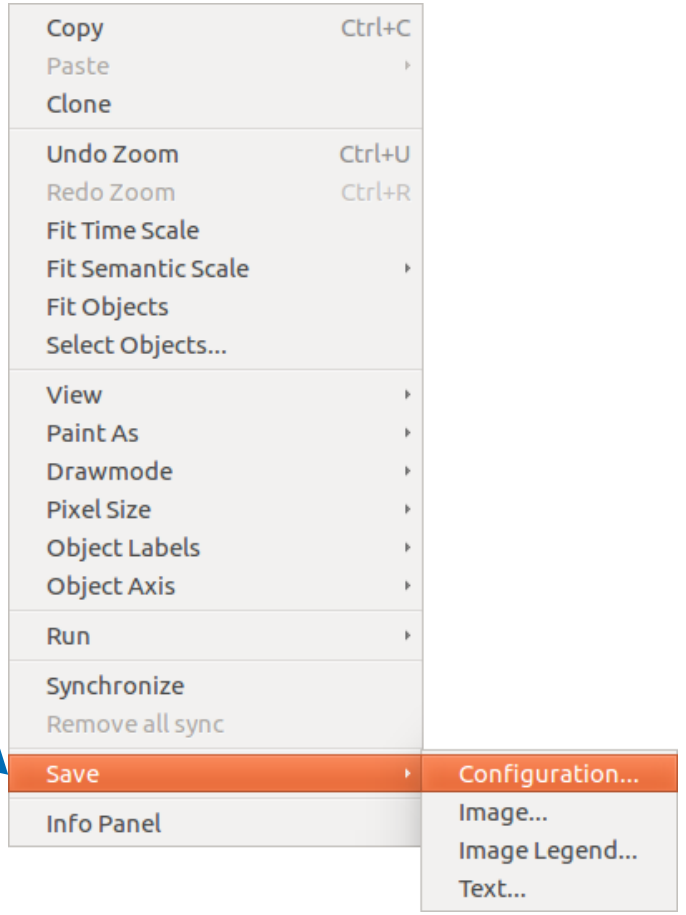
- **Hints → Call stack references → Caller function**



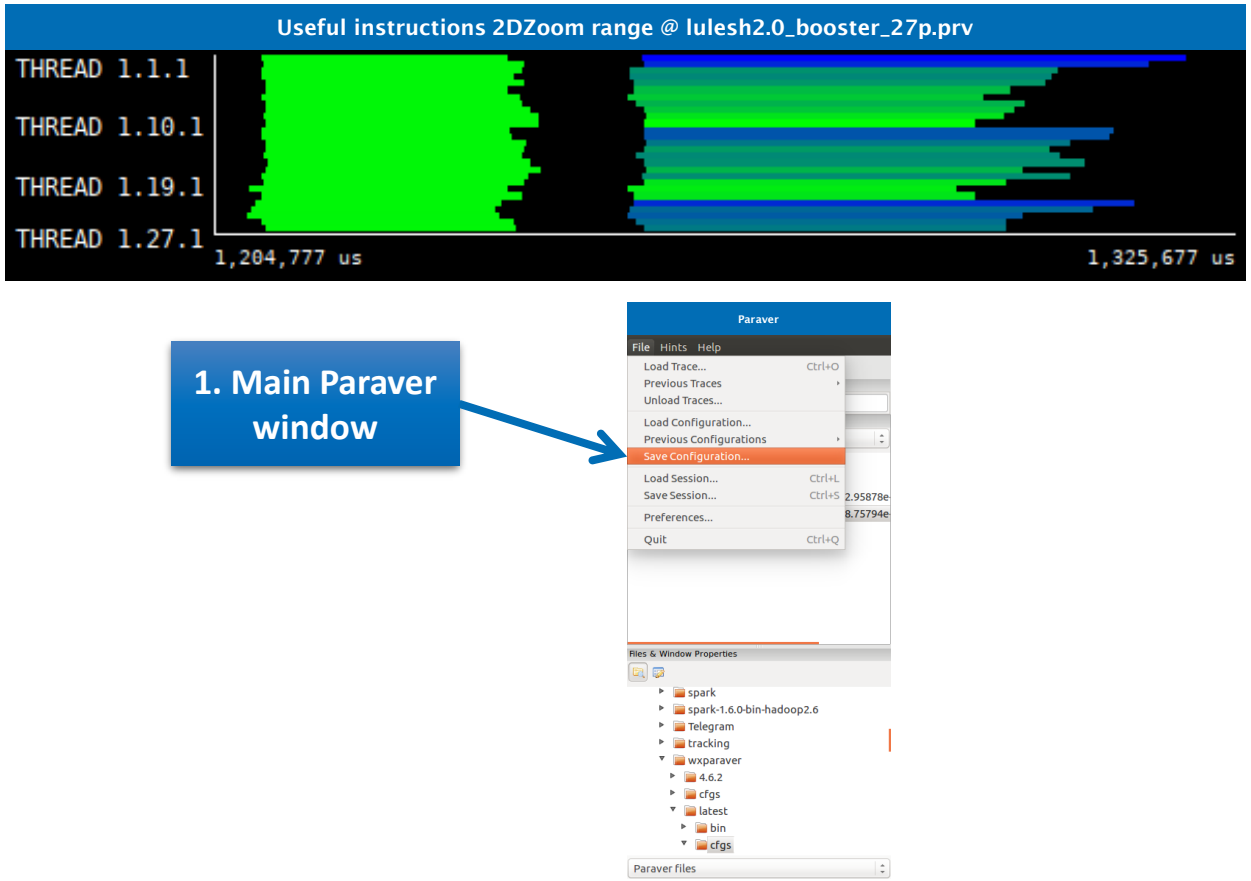
Save CFG's (method 1)



Right click on timeline

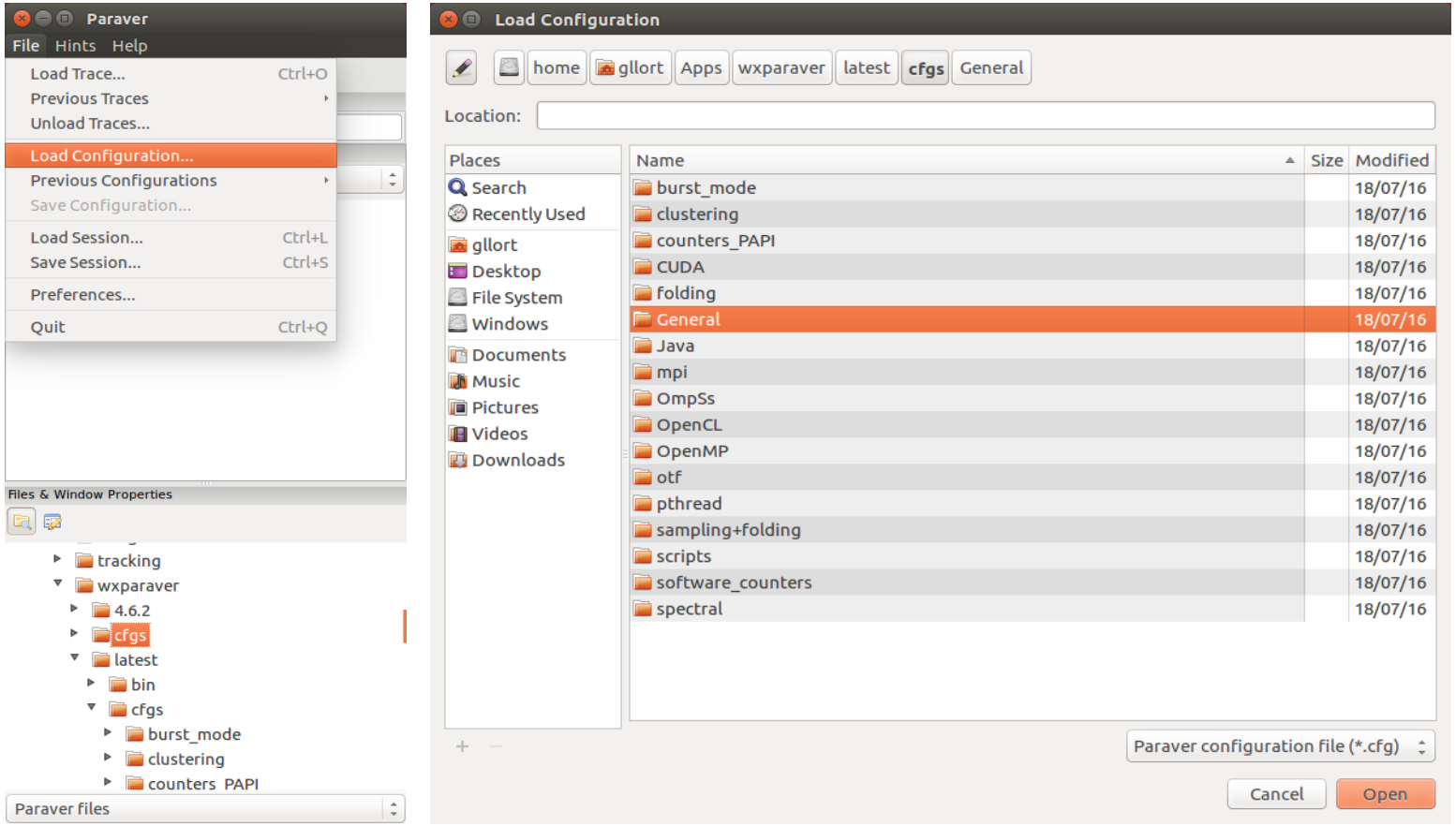


Save CFG's (method 2)



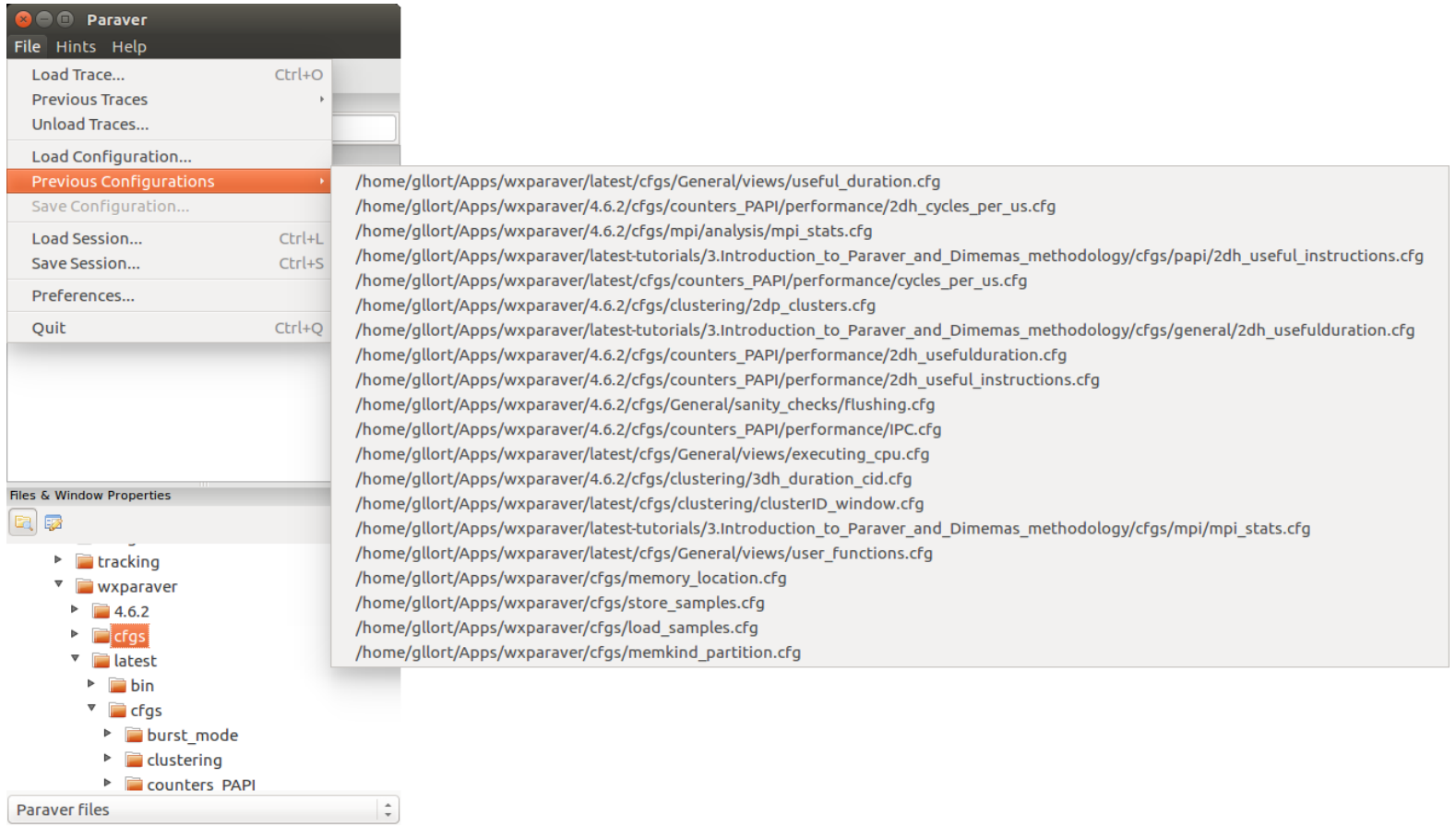
CFG's distribution

- Paraver comes with many included CFG's ➔ Apply any CFG to any trace!



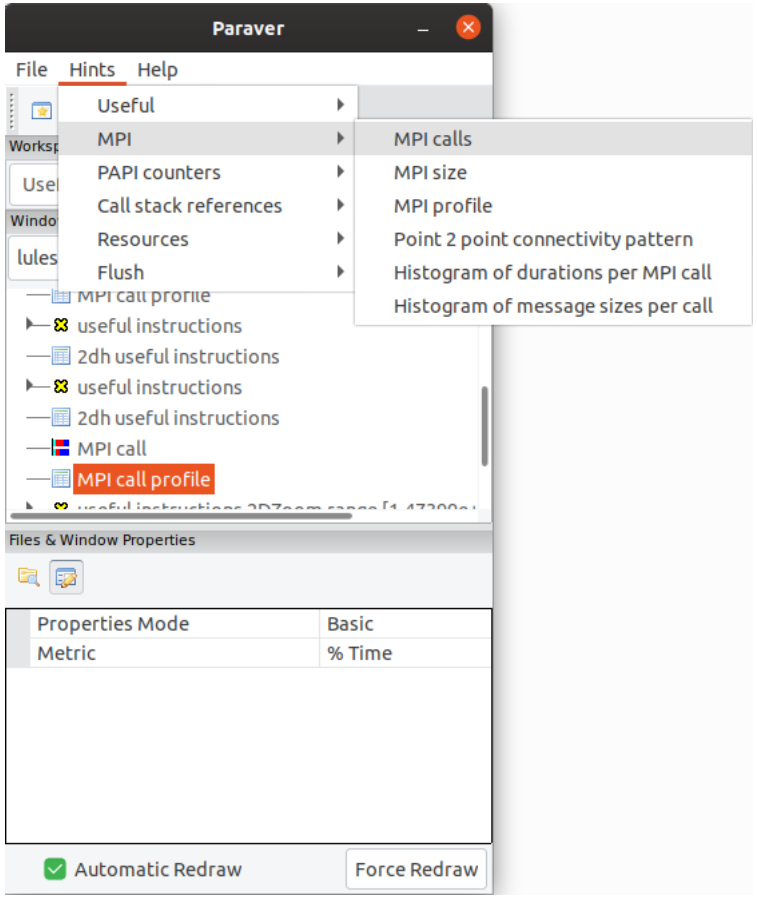
CFG's distribution

- Paraver comes with many included CFG's ➔ Apply any CFG to any trace!



Hints: a good place to start!

- Suggested CFG's based on the contents of the trace



Extrae
Hands-On

Germán Llort
(tools@bsc.es)
Barcelona Supercomputing Center

Extrae features

- Platforms
  - Intel, Cray, BlueGene, MIC, ARM, Android, Fujitsu Sparc, RISC-V …
- 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
- Periodic sampling

- User events (Extrae API)

No need to recompile nor relink!

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
- Compiler instrumentation (`-finstrument-functions`)
- Static link (PMPI, Extrae API)

BSC-Tools on JUWELS-BOOSTER

- Log into JUWELS-BOOSTER:

```
laptop> ssh -Y <USER>@juwels-booster.fz-juelich.de
```

- Extrae is available under:

```
juwels> ls /p/project/training2123/tools/extrae/3.8.3_psmpi5.4.9-1
```

- … and Paraver under:

```
juwels> ls /p/project/training2123/tools/paraver/4.10.0
```

Getting your first trace

- Provided folder `/p/project/training2123/work/llort1/bsc-hands-on` contains:
  - Sample application compiled with NVHPC + ParaStationMPI toolchain (`tea_leaf`)
  - Jobscripts to execute and trace (`run.sbatch, trace.sh`)
  - Configuration of the tracing tool (`extrae.xml`)
  - Already generated tracefiles (`traces/*.{pcf,prv,row}`)

- Copy this folder and you are ready to follow this hands-on tutorial

```
juwels> cp -r /p/project/training2123/work/llort1/bsc-hands-on /p/project/training2123/work/$USER
```

Using Extrae in 3 steps

1. **Adapt** your job submission script

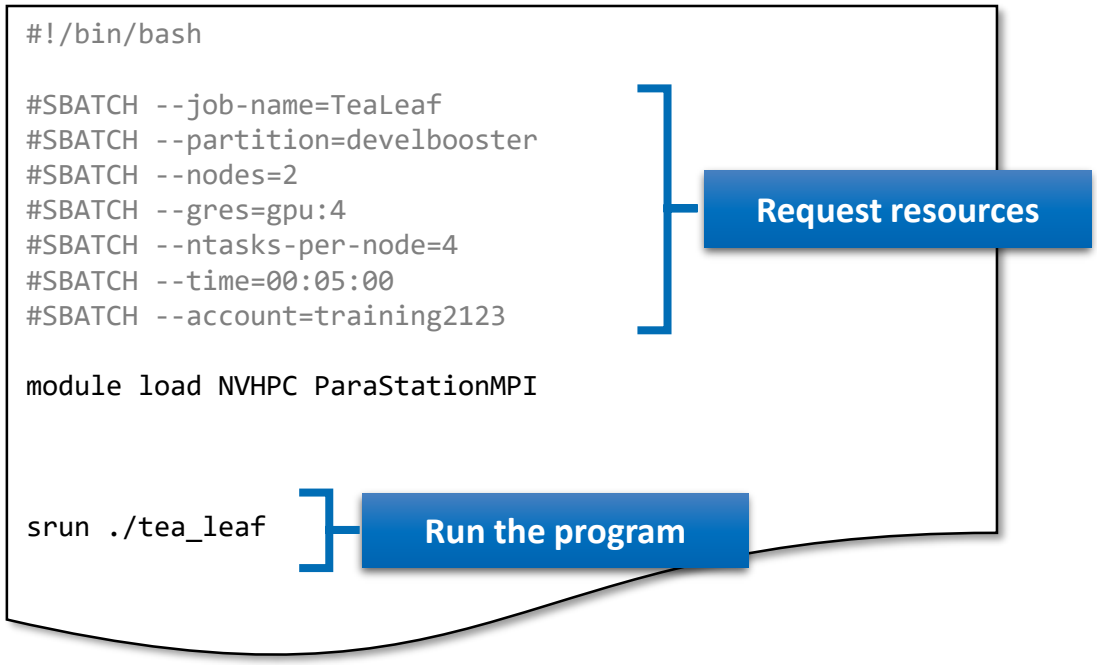
2. **Configure** what to trace
   - XML configuration file
   - Example configurations at `$EXTRAE_HOME/share/example`

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`

Step 1: Adapt the job script to load Extrae

- Example of a standard jobscript (without tracing)



Step 1: Adapt the job script to load Extrae

- Jobscript modified to load Extrae

```
juwels> cat /p/project/training2123/work/$USER/bsc-hands-on/extrae/run.sbatch
```

```
#!/bin/bash

#SBATCH --job-name=TeaLeaf
#SBATCH --partition=develbooster
#SBATCH --nodes=2
#SBATCH --gres=gpu:4
#SBATCH --ntasks-per-node=4
#SBATCH --time=00:05:00
#SBATCH --account=training2123

module load NVHPC ParaStationMPI

export TRACE_NAME=tealeaf_8p_4gpu_maxit6.prv

srun ./trace.sh ./tea_leaf
```

Run with Extrae

Step 1: Adapt the job script to load Extrae

- Tracing launcher helper script

```
juwels> cat /p/project/training2123/work/$USER/bsc-hands-on/extrae/trace.sh
```

```
#!/bin/bash

#SBATCH --job-name=TeaLeaf
#SBATCH --partition=develbooster
#SBATCH --nodes=2
#SBATCH --gres=gpu:4
#SBATCH --ntasks-per-node=4
#SBATCH --time=00:05:00
#SBATCH --account=training2123

module load NVHPC ParaStationMPI

export TRACE_NAME=tealeaf_8p_4gpu_maxit6.prv

srun ./trace.sh ./tea_leaf
```

```
#!/bin/bash

export EXTRAE_HOME=/p/project/training2123/tools/extrae/3.8.3_psmpi5.4.9-1

export EXTRAE_CONFIG_FILE=./extrae.xml

# Keep this if your app uses MPI, remove otherwise
export EXTRAE_SKIP_AUTO_LIBRARY_INITIALIZE=1

# Select tracing library
export LD_PRELOAD=${EXTRAE_HOME}/lib/libcudampitrace.so

# Run the program
$*
```

What to trace?

Choose tracing library depending on the app type (see next slide)

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] add suffix "f" if code is Fortran and default lib misses MPI activity

Step 2: Extrae XML configuration

```
juwels> cat /p/project/training2123/work/$USER/bsc-hands-on/extrae/extrae.xml
```

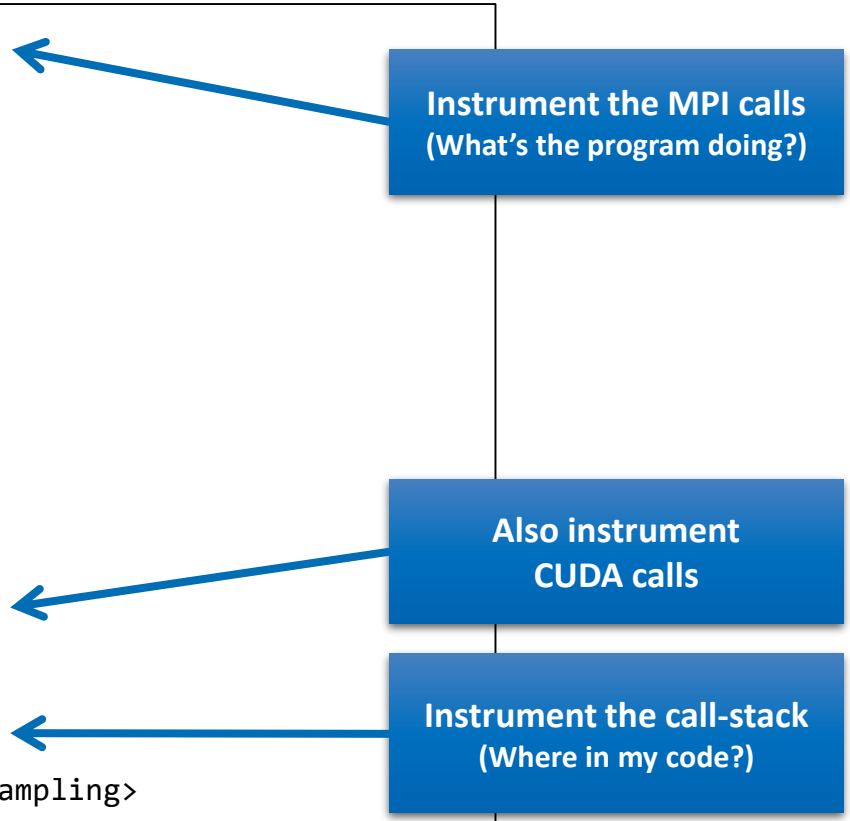
```
<mpi enabled="yes">
  <counters enabled="yes" />
</mpi>

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

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

<cuda enabled="yes">

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



Step 2: Extrae XML configuration (II)

```
juwels> cat /p/project/training2123/work/$USER/bsc-hands-on/extrae/extrae.xml
```

```
<counters enabled="yes">
  <cpu enabled="yes" starting-set-distribution="1">
    <set enabled="yes" domain="all" changeat-time="0">
      PAPI_TOT_INS,PAPI_TOT_CYC
    </set>
  </cpu>
  <network enabled="no" />
  <resource-usage enabled="no" />
  <memory-usage enabled="no" />
</counters>
```

Select which HW counters are measured
(How's the machine doing?)

```
<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>
```

Extrae buffer size
(Flush/memory trade-off)

Additional sampling
(Want more details?)

Automatic post-processing to generate the Paraver trace
(Very handy!)

Step 3: Run it!

- Submit your job as usual

```
juwels> cd /p/project/training2123/work/$USER/bsc-hands-on/extrae
juwels> sbatch run.sbatch
```

- Once finished (check with "`squeue -u $USER`") you will have the trace in the same folder (3 files):

```
juwels> ls
… tealeaf_8p_4gpu_maxit6.pcf tealeaf_8p_4gpu_maxit6.prv  tealeaf_8p_4gpu_maxit6.row
```

- Any trouble? There's a trace already generated under folder "`bsc-hands-on/traces`"
- Now let's look into it!

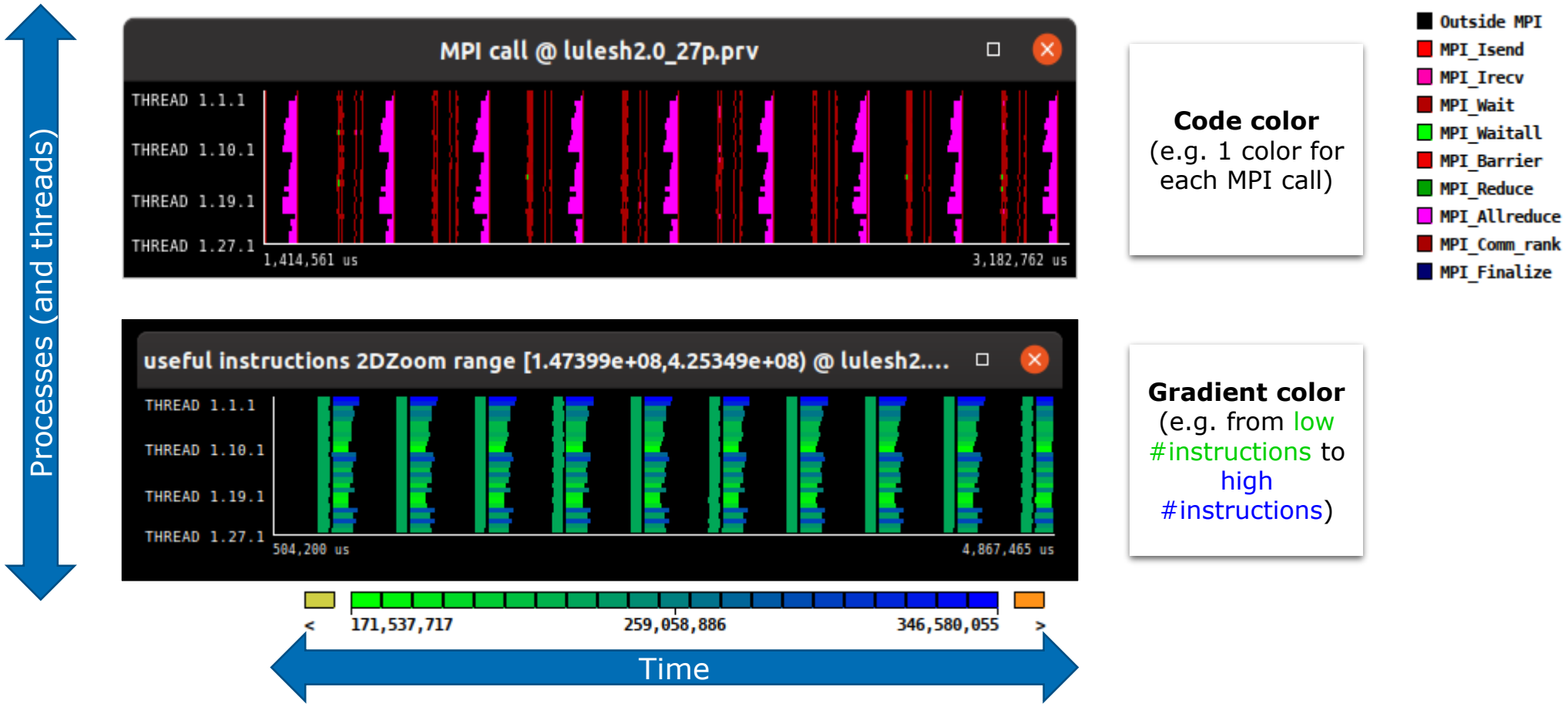
```
laptop> scp <USER>@juwels-booster.fz-juelich.de:/p/project/training2123/
work/$USER/bsc-hands-on/extrae/tealeaf*.{pcf,prv,row} .
```

Thank you!

Germán Llort
(tools@bsc.es)
Barcelona Supercomputing Center

Cheatsheet: 3 main views of Paraver (I)

- Timeline



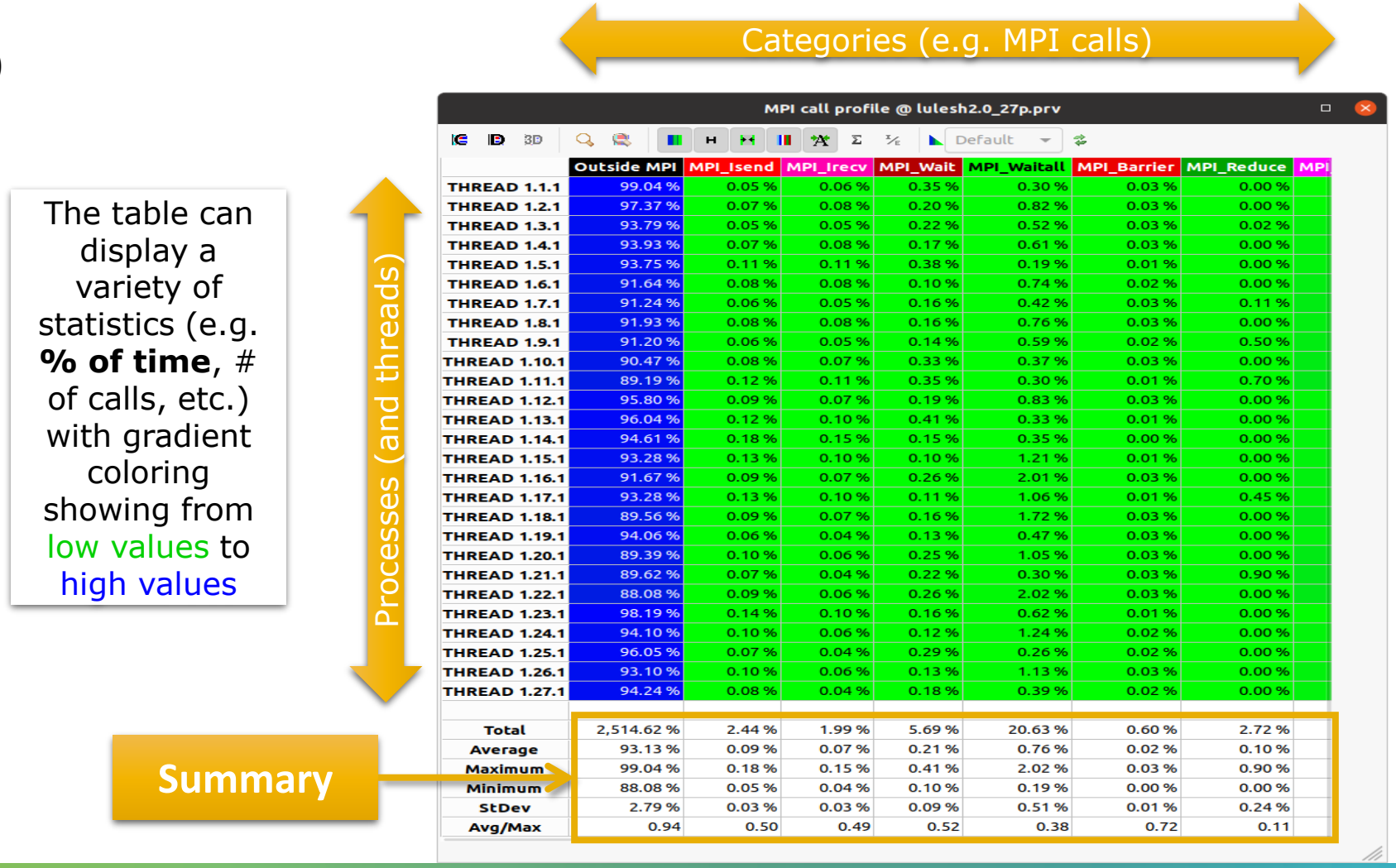
Code color
(e.g. 1 color for each MPI call)

- Outside MPI
- MPI_Isend
- MPI_Irecv
- MPI_Wait
- MPI_Waitall
- MPI_Barrier
- MPI_Reduce
- MPI_Allreduce
- MPI_Comm_rank
- MPI_Finalize

Gradient color
(e.g. from low #instructions to high #instructions)

Cheatsheet: 3 main views of Paraver (II)

- Table (Profile)



	Outside MPI	MPI_Isend	MPI_Irecv	MPI_Wait	MPI_Waitall	MPI_Barrier	MPI_Reduce	MPI
THREAD 1.1.1	99.04 %	0.05 %	0.06 %	0.35 %	0.30 %	0.03 %	0.00 %	
THREAD 1.2.1	97.37 %	0.07 %	0.08 %	0.20 %	0.82 %	0.03 %	0.00 %	
THREAD 1.3.1	93.79 %	0.05 %	0.05 %	0.22 %	0.52 %	0.03 %	0.02 %	
THREAD 1.4.1	93.93 %	0.07 %	0.08 %	0.17 %	0.61 %	0.03 %	0.00 %	
THREAD 1.5.1	93.75 %	0.11 %	0.11 %	0.38 %	0.19 %	0.01 %	0.00 %	
THREAD 1.6.1	91.64 %	0.08 %	0.08 %	0.10 %	0.74 %	0.02 %	0.00 %	
THREAD 1.7.1	91.24 %	0.06 %	0.05 %	0.16 %	0.42 %	0.03 %	0.11 %	
THREAD 1.8.1	91.93 %	0.08 %	0.08 %	0.16 %	0.76 %	0.03 %	0.00 %	
THREAD 1.9.1	91.20 %	0.06 %	0.05 %	0.14 %	0.59 %	0.02 %	0.50 %	
THREAD 1.10.1	90.47 %	0.08 %	0.07 %	0.33 %	0.37 %	0.03 %	0.00 %	
THREAD 1.11.1	89.19 %	0.12 %	0.11 %	0.35 %	0.30 %	0.01 %	0.70 %	
THREAD 1.12.1	95.80 %	0.09 %	0.07 %	0.19 %	0.83 %	0.03 %	0.00 %	
THREAD 1.13.1	96.04 %	0.12 %	0.10 %	0.41 %	0.33 %	0.01 %	0.00 %	
THREAD 1.14.1	94.61 %	0.18 %	0.15 %	0.15 %	0.35 %	0.00 %	0.00 %	
THREAD 1.15.1	93.28 %	0.13 %	0.10 %	0.10 %	1.21 %	0.01 %	0.00 %	
THREAD 1.16.1	91.67 %	0.09 %	0.07 %	0.26 %	2.01 %	0.03 %	0.00 %	
THREAD 1.17.1	93.28 %	0.13 %	0.10 %	0.11 %	1.06 %	0.01 %	0.45 %	
THREAD 1.18.1	89.56 %	0.09 %	0.07 %	0.16 %	1.72 %	0.03 %	0.00 %	
THREAD 1.19.1	94.06 %	0.06 %	0.04 %	0.13 %	0.47 %	0.03 %	0.00 %	
THREAD 1.20.1	89.39 %	0.10 %	0.06 %	0.25 %	1.05 %	0.03 %	0.00 %	
THREAD 1.21.1	89.62 %	0.07 %	0.04 %	0.22 %	0.30 %	0.03 %	0.90 %	
THREAD 1.22.1	88.08 %	0.09 %	0.06 %	0.26 %	2.02 %	0.03 %	0.00 %	
THREAD 1.23.1	98.19 %	0.14 %	0.10 %	0.16 %	0.62 %	0.01 %	0.00 %	
THREAD 1.24.1	94.10 %	0.10 %	0.06 %	0.12 %	1.24 %	0.02 %	0.00 %	
THREAD 1.25.1	96.05 %	0.07 %	0.04 %	0.29 %	0.26 %	0.02 %	0.00 %	
THREAD 1.26.1	93.10 %	0.10 %	0.06 %	0.13 %	1.13 %	0.03 %	0.00 %	
THREAD 1.27.1	94.24 %	0.08 %	0.04 %	0.18 %	0.39 %	0.02 %	0.00 %	
Total	2,514.62 %	2.44 %	1.99 %	5.69 %	20.63 %	0.60 %	2.72 %	
Average	93.13 %	0.09 %	0.07 %	0.21 %	0.76 %	0.02 %	0.10 %	
Maximum	99.04 %	0.18 %	0.15 %	0.41 %	2.02 %	0.03 %	0.90 %	
Minimum	88.08 %	0.05 %	0.04 %	0.10 %	0.19 %	0.00 %	0.00 %	
StDev	2.79 %	0.03 %	0.03 %	0.09 %	0.51 %	0.01 %	0.24 %	
Avg/Max	0.94	0.50	0.49	0.52	0.38	0.72	0.11	

Cheatsheet: 3 main views of Paraver (III)

- Histogram

Displays continuous metrics (e.g. **instructions executed**, duration of computations, bytes sent/received, etc.)

Gradient color represents low to high values of selected statistic (**time %**, # instances, etc.)

General tip: straight lines are good (all processes show same behavior), while variabilities usually indicate imbalances