BSCTools
Hands-On

Judit Giménez, Jesús Labarta

Barcelona Supercomputing Center

Getting a trace with Extrae

Extrae features

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

No need to recompile / relink!

(*) GNU, Inttel, OMPT

How does Extrae work?

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

- Dynamic instrumentation (not yet available at leftraru)
  - Based on DynInst (developed by U.Wisconsin/U.Maryland)
    - Instrumentation in memory
    - Binary rewriting

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

How to use Extrae?

1. Adapt the job submission script

2. [Optional] Tune the Extrae XML configuration file
   - Examples distributed with Extrae at $EXTRAE_HOME/share/example

3. Run with instrumentation

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

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

Log in to pi and copy the example to your home directory

```
> ssh -X <uid>@pi.ircpi.kobe-u.ac.jp

> cp -r /home/S11505/shared/tutorial/ntchem-mini $HOME

> cd $HOME/ntchem-mini/run_h2o
```

Adapt the job script to load Extrae with LD_PRELOAD

PIcomputer.sh

```
#!/bin/bash -x
#
#PJM -o "h2o_10_rimp2.out"
#PJM -e "h2o_10_rimp2.err"
#PJM --rsc-list "rscgrp=small"
#PJM --rsc-list "node=2"
#PJM --rsc-list "elapse=0:10:00"
##PJM --mpi "proc=32"
#PJM -j
#

export FLIB_FASTOMP=FALSE
export FLIB_CNTL_BARRIER_ERR=FALSE
export OMP_NUM_THREADS=16

mpiexec ./rimp2.exe_debug
```

Adapt the job script to load Extrae with LD_PRELOAD

Picomputer_extrae.sh

```
#!/bin/bash -x
#
#PJM -o "h2o_10_rimp2.out"
#PJM -e "h2o_10_rimp2.err"
#PJM --rsc-list "rscgrp=small"
#PJM --rsc-list "node=2"
#PJM --rsc-list "elapse=0:10:00"
##PJM --mpi "proc=32"
#PJM -j
#

xospastop

export FLIB_FASTOMP=FALSE
export FLIB_CNTL_BARRIER_ERR=FALSE
export OMP_NUM_THREADS=16

source extrae/trace.sh
export TRACE=t_ntchem_h2o.prv

mpiexec ./rimp2.exe_debug
```

extrae/trace.sh

```
#!/bin/bash
export EXTRAE_HOME=/home/S11505/shared/tools/BSC/extrae-3.3.0/

export EXTRAE_CONFIG_FILE=extrae/extrae.xml

export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:
/home/S11505/judit/dependencies/binutils-2.24/lib

export LD_PRELOAD=$EXTRAE_HOME/lib/libmpitracef.so:libpapi.so
```

LD_PRELOAD library selection

- Choose depending on the application type

Library	Serial	MPI	OpenMP	pthread	CUDA
libseqtrace	✓				
libmpitrace[f][1]		✓			
libomptrace			✓		
libpttrace				✓	
libcudatrace					✓
libompitrace[f] [1]		✓	✓		
libptmpitrace[f] [1]		✓		✓	
libcudampitrace[f] [1]		✓			✓

[1] include suffix "f" in Fortran codes

Extrae XML configuration: extrae_config.xml

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

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

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

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

Trace MPI calls + HW counters

Trace call-stack events @ MPI calls

Extrae XML configuration: extrae_config.xml (II)

```
<counters enabled="yes">
  <cpu enabled="yes" starting-set-distribution="cyclic">
 <set enabled="yes" changeat-time="500000us" domain="all">
 PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_L1_DCM,PAPI_L2_TCM,PAPI_BR_MSP,PAPI_FP_INS
  </set>
 <set enabled="yes" changeat-time="500000us" domain="all">
 PAPI_TOT_INS,PAPI_TOT_CYC,PAPI_SR_INS,PAPI_LD_INS
 </set>
  </cpu>

  <network enabled="no" />

  <resource-usage enabled="no" />

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

Define which HW counters are measured

Extrae XML configuration: extrae_config.xml (III)

```
<buffer enabled="yes">
  <size enabled="yes">500000</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"
>
  $TRACE$
</merge>
```

Trace buffer size

Merge intermediate files into Paraver trace

Run with instrumentation

- Submit your job @pi.ircpi.kobe-u.ac.jp

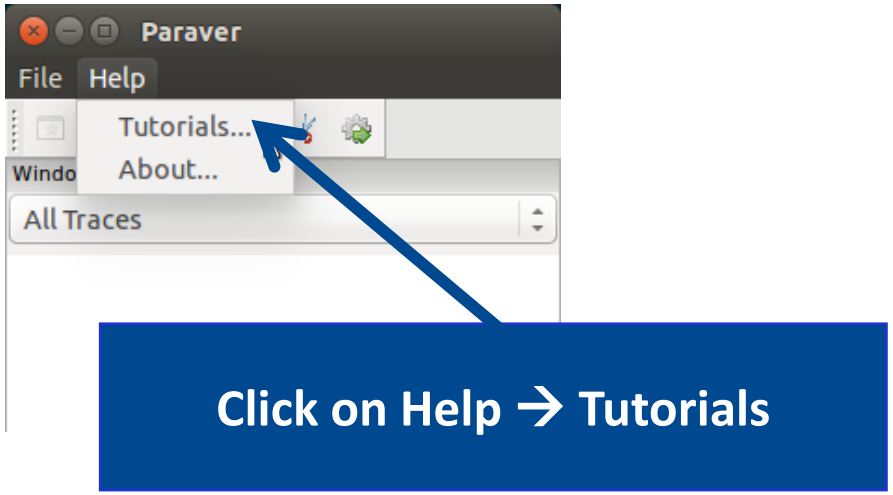
```
> cd $HOME/ntchem-mini/run_h2o
> pjsub PIcomputer_extrae.sh
```

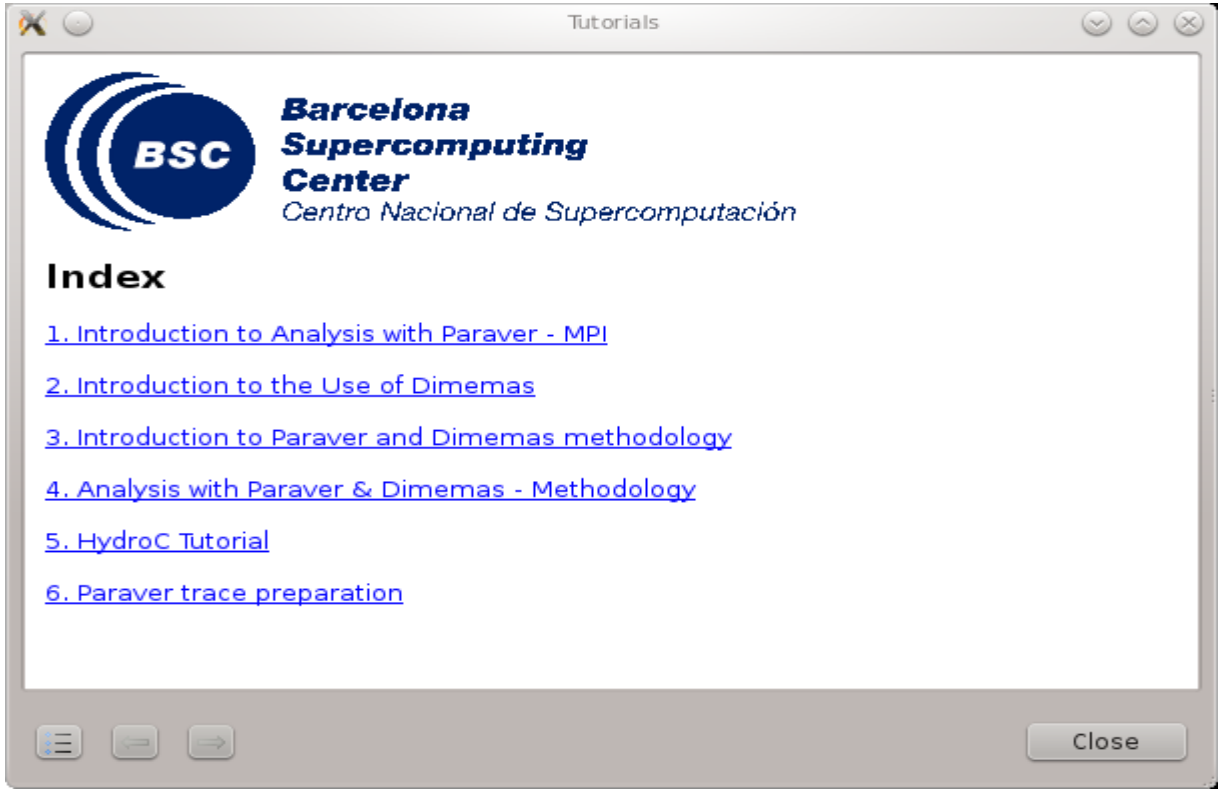
- Load the trace with Paraver

```
> wxparaver t_nchem_h2o.prv
```

Launch Paraver

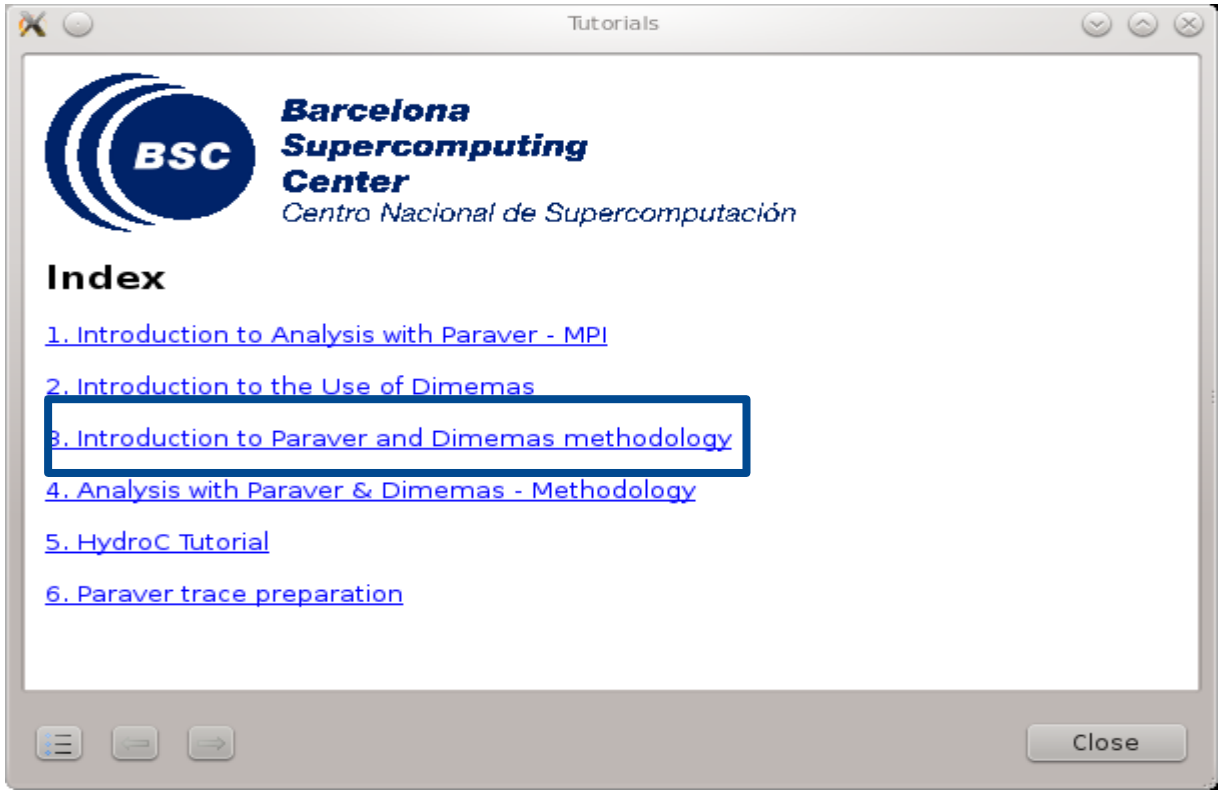
```
> source /home/S11505/shared/tools/setup.sh
> wxparaver
```





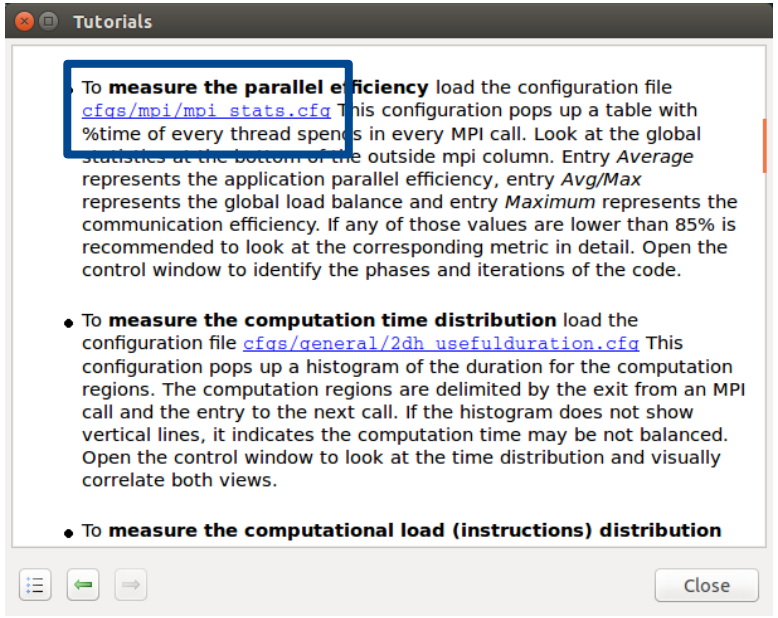
First steps of analysis

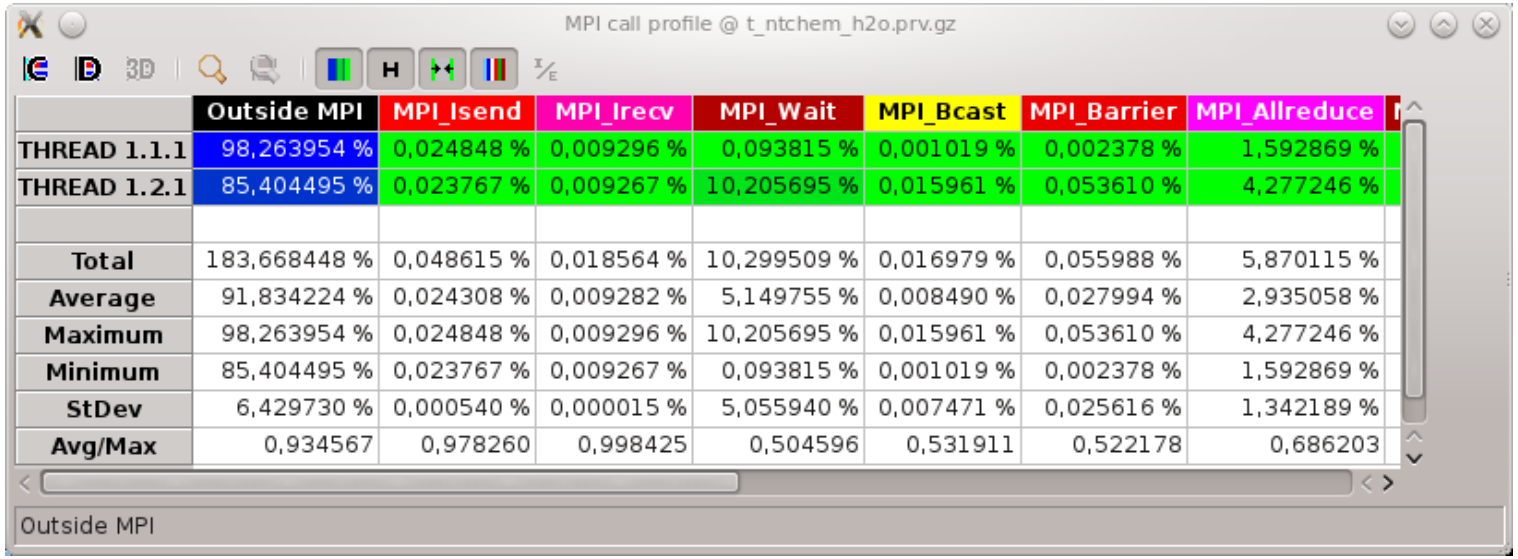
- Open Tutorial #3
  - Help → Tutorials



Measure the parallel efficiency

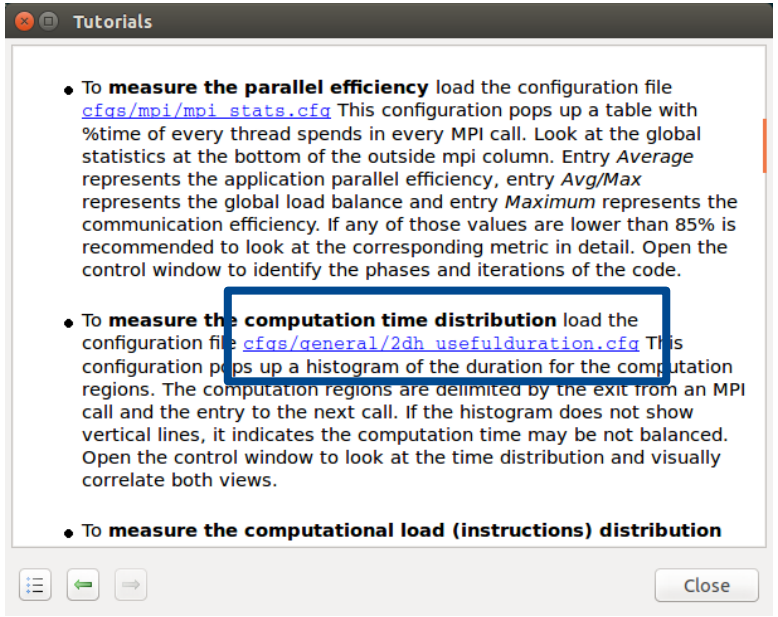
- Click on the “mpi_stats.cfg”





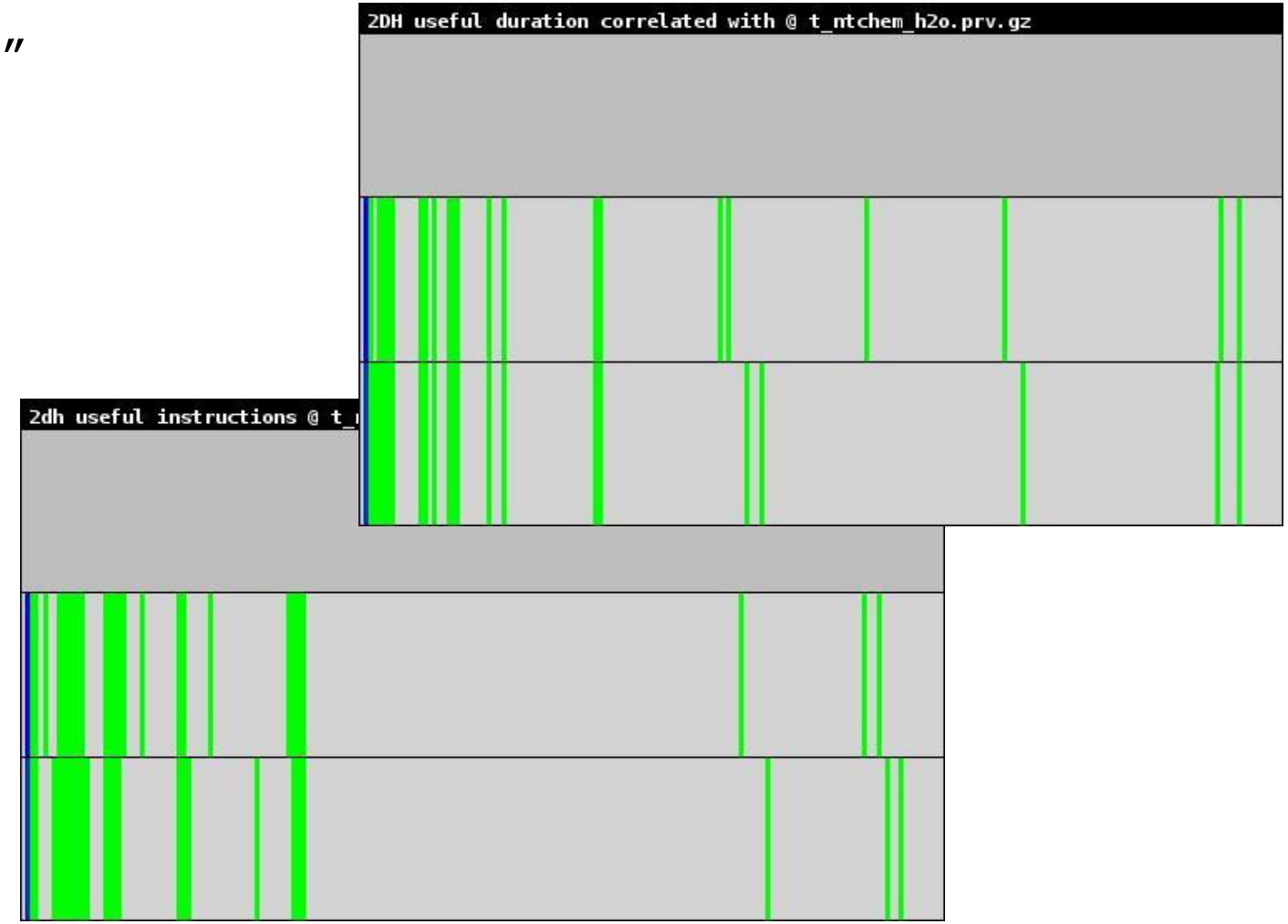
Measure the computation time distribution

- Click on the “2dh_usefulduration.cfg”



…and the “2dh_useful_instructions.cfg”

Zoom on the relevant (dark) area



Compare with other configurations

- Taxol test case (f.i. 32 nodes)
  - If you want to get the trace…

```
> cd $HOME/ntchem_mini/run_taxol
> pjsub PIcomputer_extrae.sh
```

  - It is already generated at

```
➢ cd $HOME/ntchem_mini/traces/taxol
```

Use clustering to analyse and compare both runs

- Run clustering

```
> cd $HOME/ntchem_mini/run_h2o
➢ clusterize.sh t_ntchem_h2o
```

- Look at the results

```
> gnuplot t_ntchem_h2o_clustered.IPC.PAPI_TOT_INS.gnuplot
```

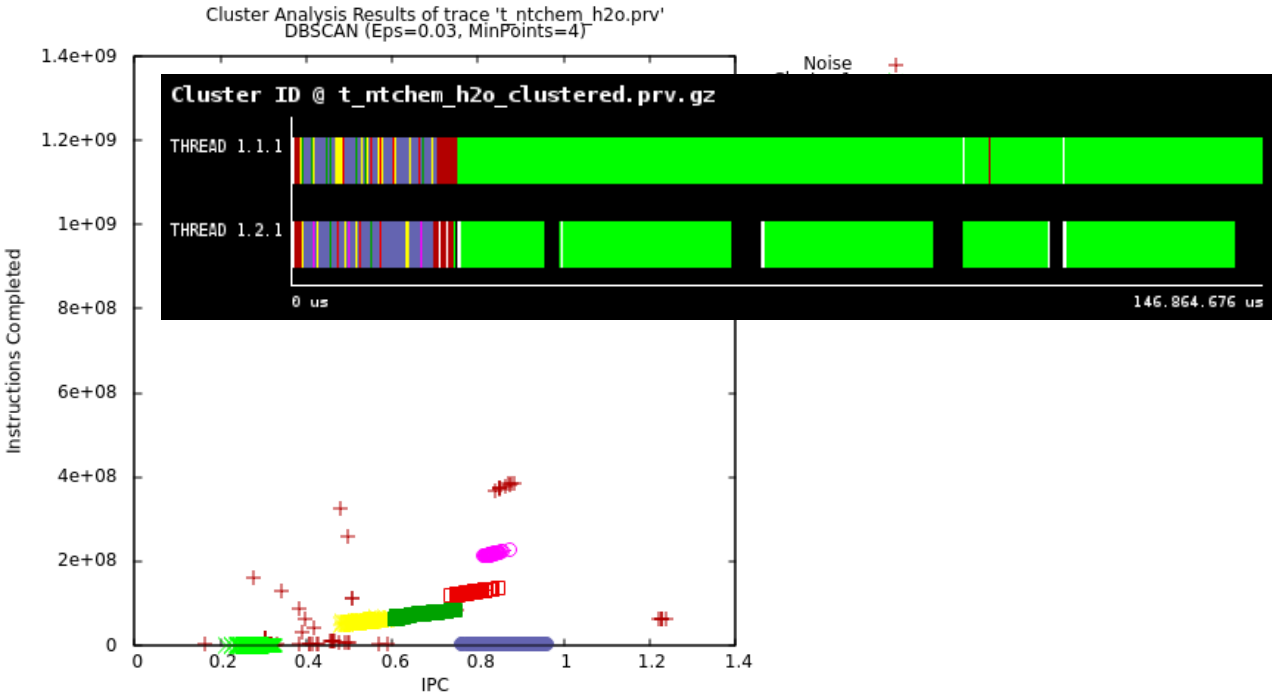
- Look at the resulting trace

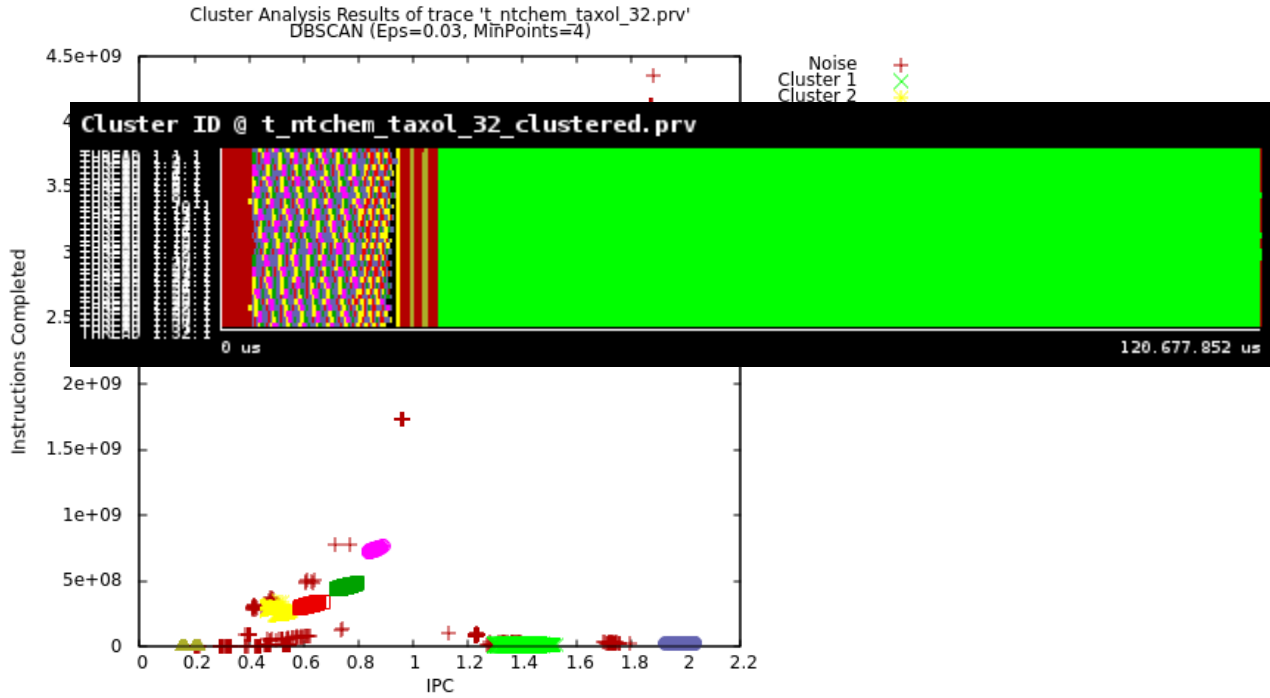
```
➢ wxparaver $HOME/ntchem_mini/run_h2o/t_ntchem_h2o_clustered.prv
```

- It is already generated at `$HOME/ntchem_mini/traces/h2o`

Looking at the clusters

- Click on the “cluster_id.cfg”





Installing Paraver

Install Paraver in your laptop

- Download the Paraver binaries from leftaru to your machine
- Built-in packages:
  - wxparaver-4.6.0-linux-x86_32.tar.gz
  - wxparaver-4.6.0-linux-x86_64.tar.gz wxparaver-4.6.0-linux_fc6-x86_64.tar-gz
  - wxparaver-4.6.0-mac.zip
  - wxparaver-4.6.0-win.zip

Example: Linux 64 bits:

```
scp <uid>@pi.ircpi.kobe-u.ac.jp:/home/S11505/shared/tools/BSC/paraver_pkgs/wxparaver-
4.5.8-linux-x86_64.tar.gz .
```

Install Paraver in your laptop (II)

- Uncompress into your home directory

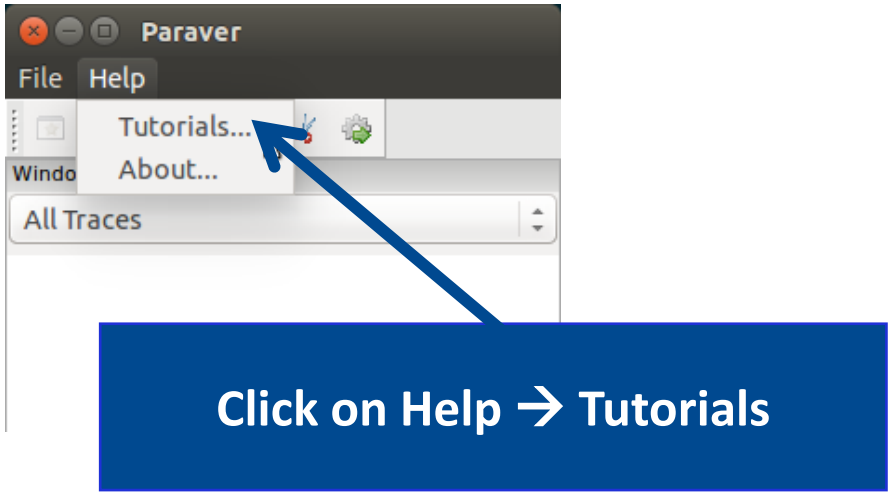
```
> tar xvfz wxparaver-4.5.6-linux-x86_64.tar.gz -C $HOME
> cd $HOME
> ln -s wxparaver-4.5.6-linux-x86_64 paraver
```

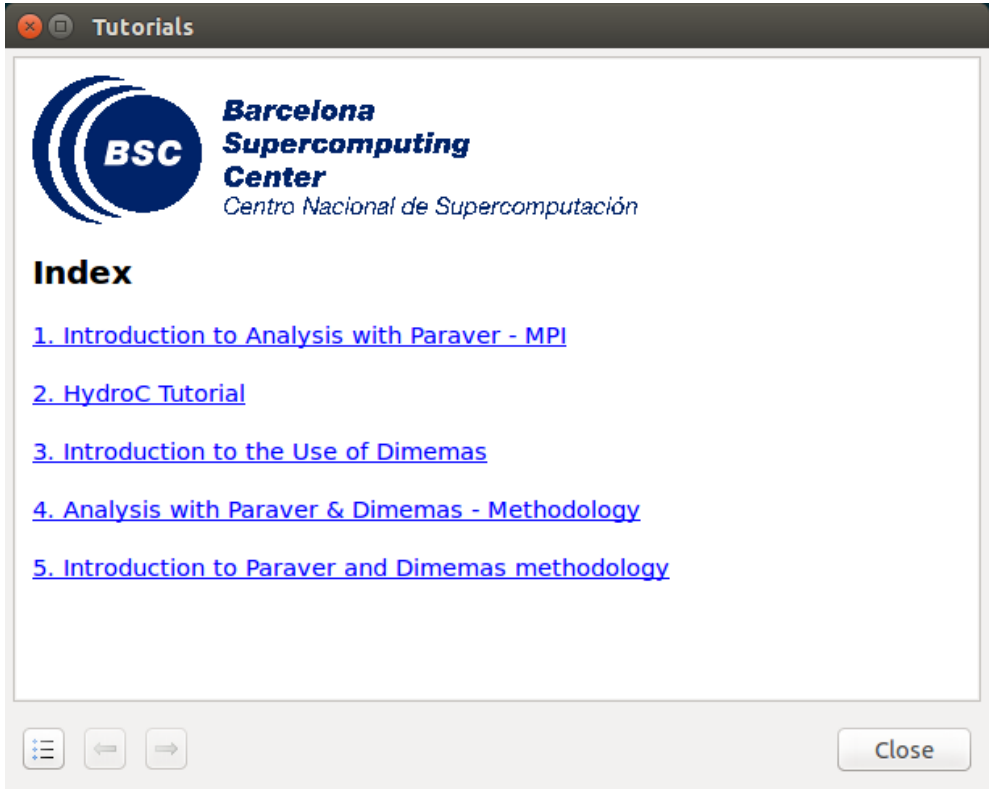
- Download Paraver tutorials and uncompress into the Paraver directory

```
> scp <uid>@pi.ircpi.kobe-
u.ac.jp:/home/S11505/shared/tools/BSC/paraver_pkgs/paraver-tutorials-
2015026.tar.gz .
> tar xvfz paraver-tutorials-20150526.tar.gz -C $HOME/paraver
```

Launch Paraver

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





Working with paraver locally

- Submit your job @pi.ircpi.kobe-u.ac.jp

```
> cd $HOME/ntchem-mini.taxol/run
> pjsub PIcomputer_extrae.sh
```

- Copy the resulting trace to your laptop and load it with Paraver **@your laptop**

```
> scp <uid>@pi.ircpi.kobe-u.ac.jp:$HOME/ntchem-mini/run_h2o/t_ntchem* $HOME
```

- Load the trace with Paraver **@your laptop**

```
> $HOME/paraver/bin/wxparaver $HOME/t_ntchem*prv*
```