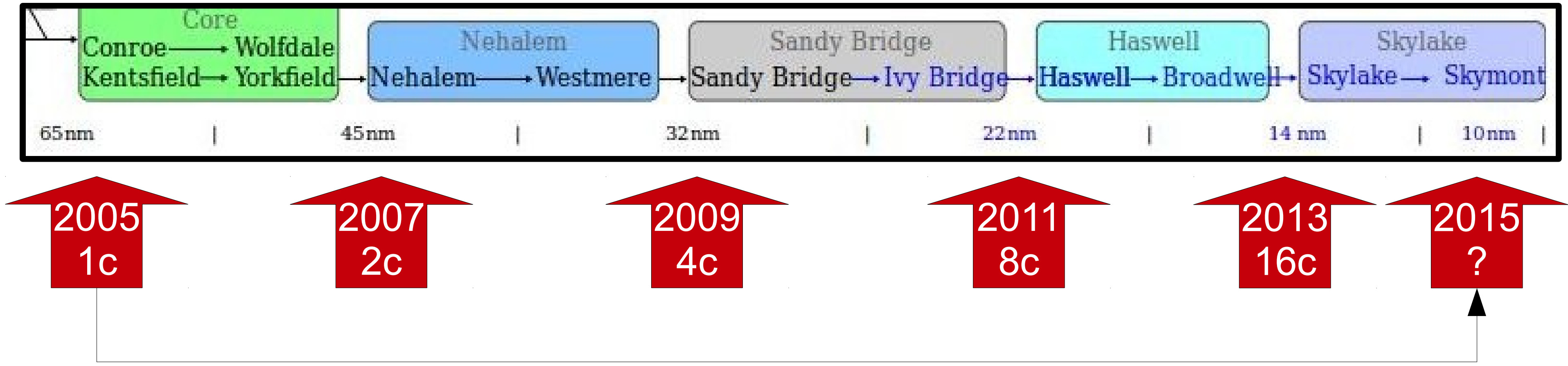
CALCUL PARALLELE et APPLICATION AUX PLASMAS FROIDS

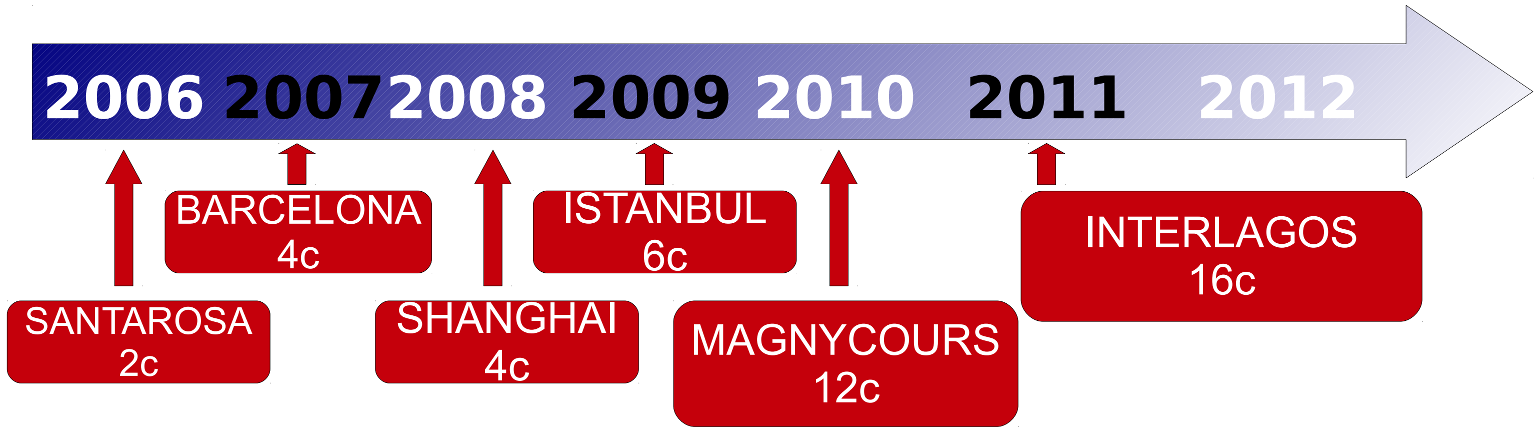
- Introduction
- Performance
- TPs

Motivation

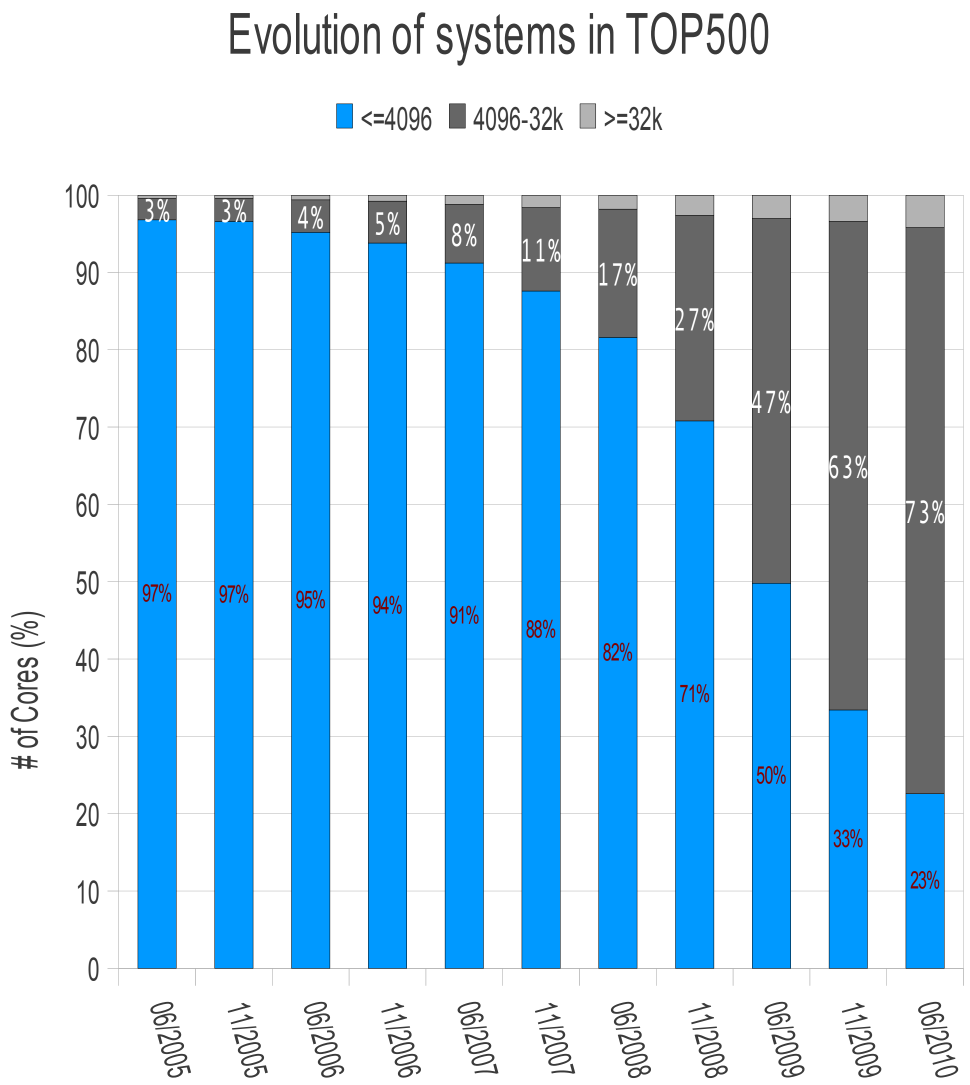
INTEL ROADMAP

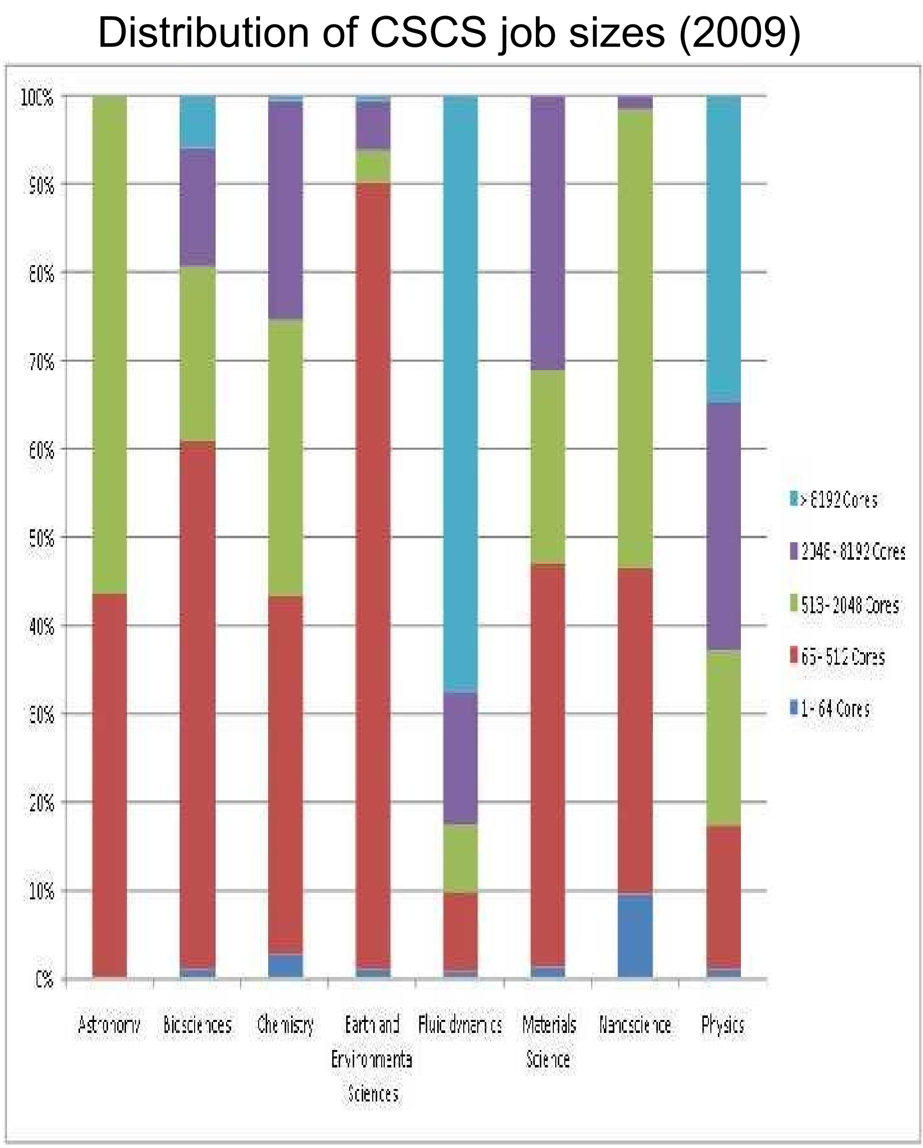


AMD ROADMAP



Motivation


Evolution of systems in TOP500
<=4096
4096-32k
>=32k
# of Cores (%)
100
90
80
70
60
50
40
30
20
10
0
3%
3%
4%
5%
8%
11%
17%
27%
47%
63%
73%
97%
97%
95%
94%
91%
88%
82%
71%
50%
33%
23%
06/2005
11/2005
06/2006
11/2006
06/2007
11/2007
06/2008
11/2008
06/2009
11/2009
06/2010


Distribution of CSCS job sizes (2009)
100%
90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
> 8192 Cores
2048 - 8192 Cores
513 - 2048 Cores
65 - 512 Cores
1 - 64 Cores
Astronomy
Biosciences
Chemistry
Earth and Environmental Sciences
Fluid dynamics
Materials Science
Nanoscience
Physics

Example : Earth Weather forecasting (1 km resolution)

$$cells \approx 5.1E10 \text{ with } \begin{cases} radius_{earth} & = & 5.1\,10^{8}\,km \\ levels & = & 100 \\ cells & = & 4\pi r^{2} * levels \end{cases}$$

X

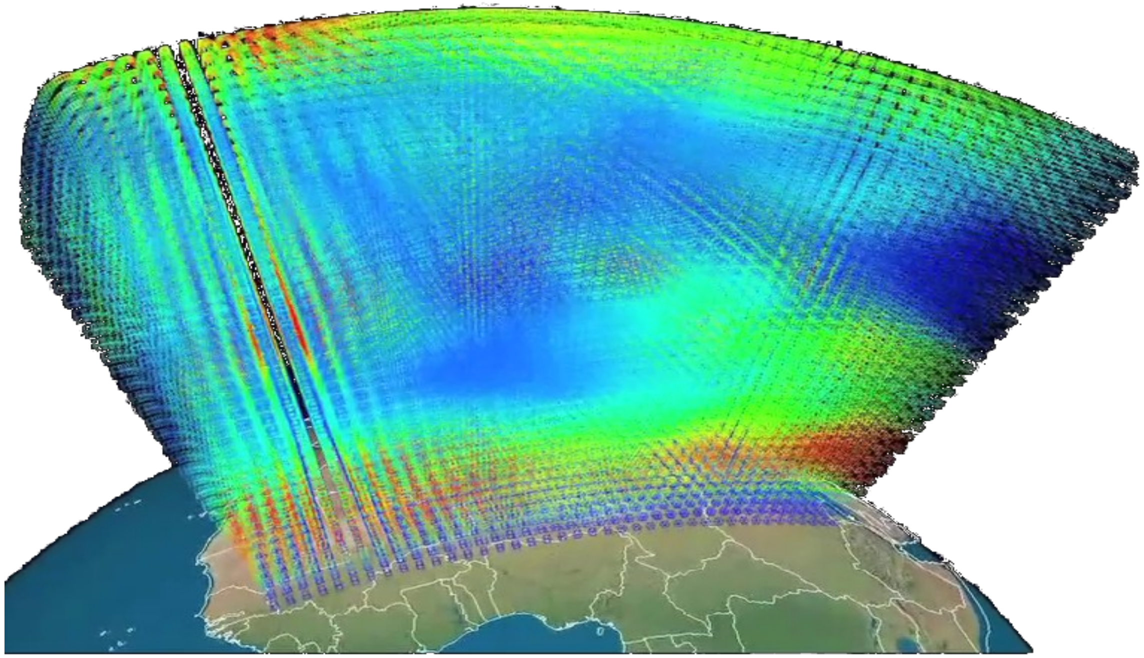
$$steps \approx 3.0E03 \text{ with } \begin{cases} forecast & = & 86400\,sec\,(24h) \\ timeperstep & = & 30\,sec \\ steps & = & forecast/timeperstep \end{cases}$$

X

$$op/cell/step \approx 1.0E03$$

1 simulated day = 1.5E17 flop

How much (real) time would it take to simulate 1 day ?



ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

CSCS CRAY systems

	ROSA CRAY XE6 (10/2011)
Cores	47872 cores AMD Interlagos 2.1 Ghz 32 c/cnode 1496 cnodes
Memory	47 TB (DDR3) 32 GB/cnode >= 1 GB/core
RPeak Perf	402 Tflops 8.4 Gflops/core Water cooled

PALU CRAY XE6 (today)
4224 cores AMD MagnyCours 2.1 Ghz 24 c/cnode 176 cnodes
5.6 TB (DDR3) 32 GB/cnode >= 1.3 GB/core
35 Tflops 8.4 Gflops/core Air cooled

Example : Earth Weather forecasting (1 km resolution)

1 simulated day = 1.5E17 flop

Computational Performance	
1 PetaFLOPS	10^15 flop/sec
1 TeraFLOPS	10^12 flop/sec
1 GigaFLOPS	10^09 flop/sec

1 simu. day	Rpeak performance	Time2solution
LAPTOP	0.022 TFlops	80 days
AUTRANS	3.2 TFlops	13 hours
ROSA	212 TFlops	12 min
JAGUAR	2330 TFlops	1.1 min
RIKKEN (#1)	8000 TFlops	19.1 sec

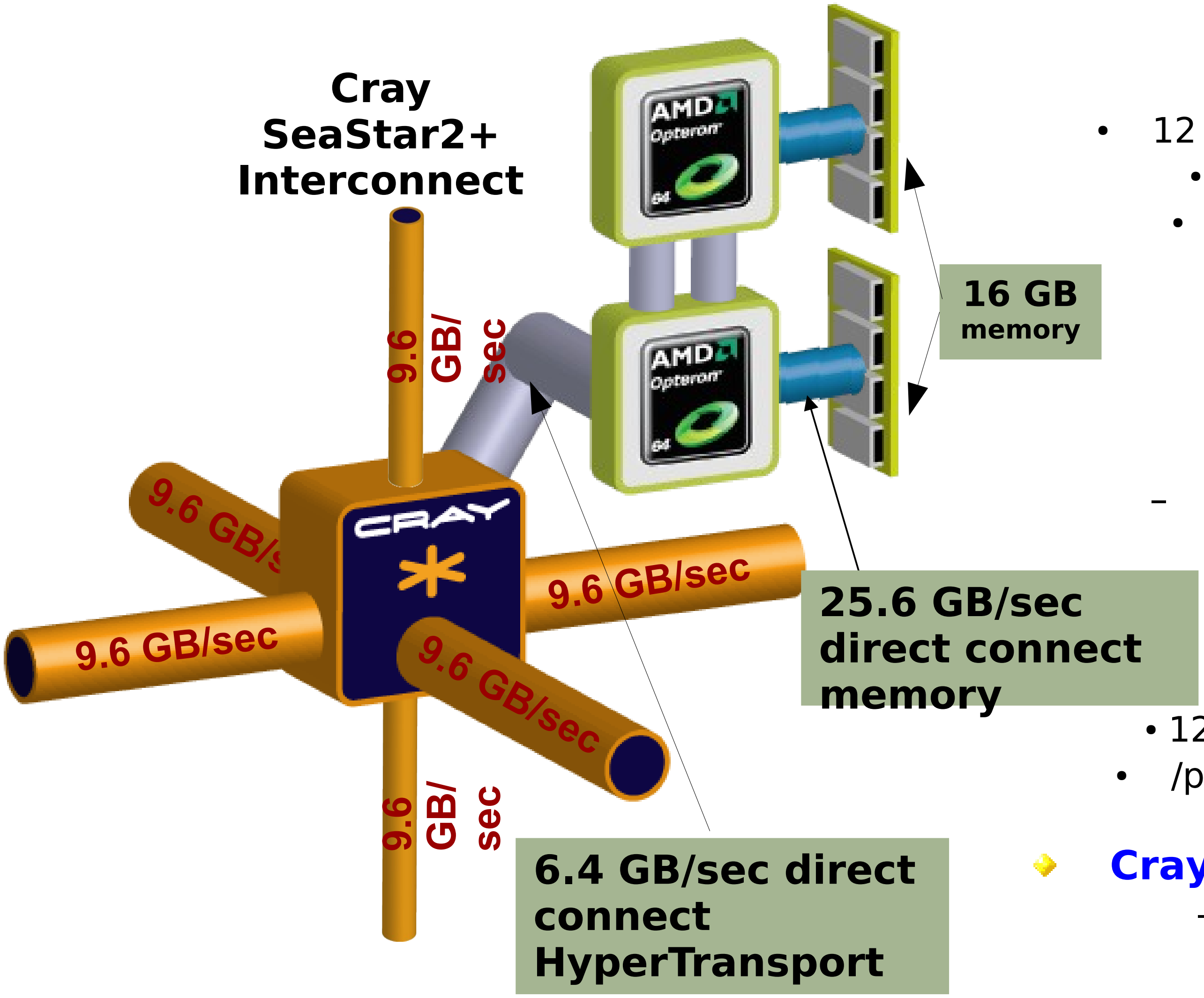
Compared to my laptop :

Rosa is ~10 000 times faster,

Rikken is more than 360 000 times faster !

CRAY XT5

CSCS production machine is a 20 cabinets Cray XT5


Cray SeaStar2+ Interconnect
9.6 GB/sec
9.6 GB/sec
9.6 GB/sec
9.6 GB/sec
9.6 GB/sec
9.6 GB/sec
CRAY
AMD Opteron
AMD Opteron
16 GB memory
25.6 GB/sec direct connect memory
6.4 GB/sec direct connect HyperTransport

CPU

- 12 cores per compute node (cnode)
  - *2.4 GHz AMD Opteron Istanbul*
- A total of 22128 compute cores
- PEAK perf = 212 Tflop/s

Memory

- *1,33 GB/core, 16 GB/cnode*
- A total of 29.5TB
- 9.6 GB/s interconnect bandwidth

I/O subsystem

- /scratch (287TB)
  - LUSTRE
  - 12 GB/s sustained write bandwidth
- /project (400 TB) and /home (5.5T)

Cray Linux Environment (CLE)

- Linux based operating system
- Supports MPI and OpenMP
- Designed to run large scale apps

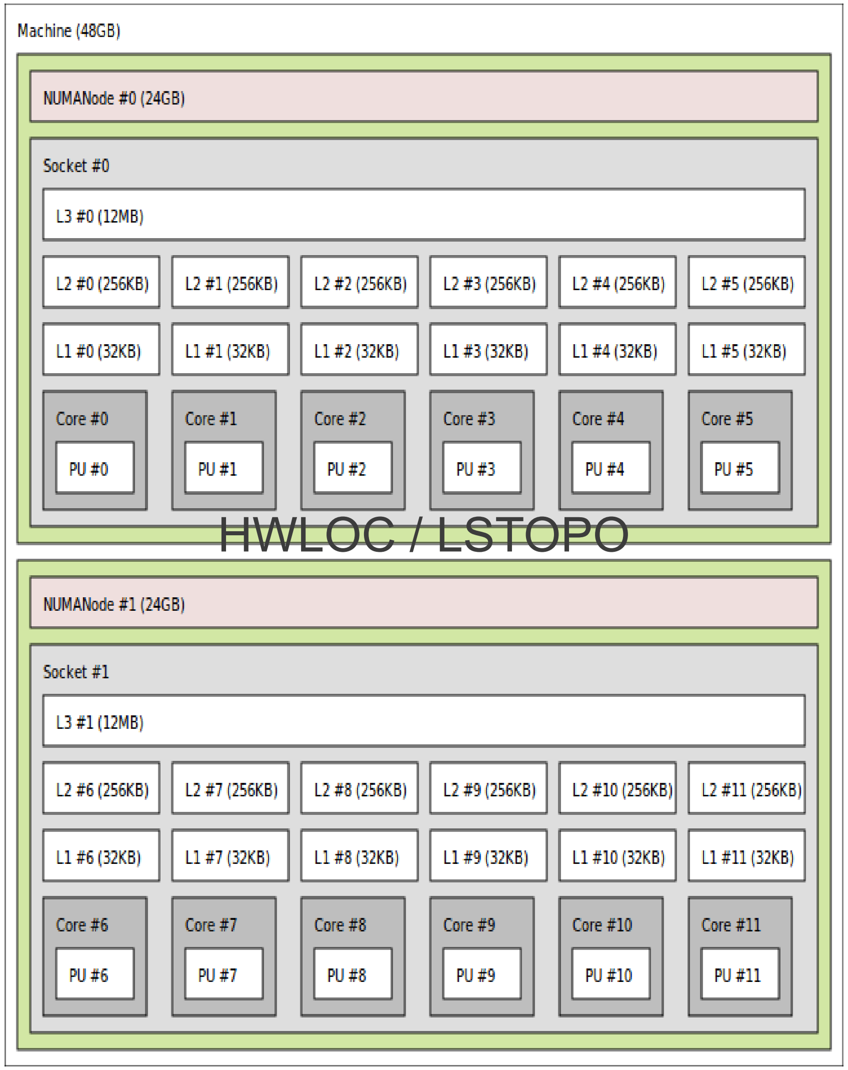
Performance model

- Application name, version, language, parallelization, algo (?), etc...
- Machine name, configuration, etc...
- Programming env. name, version, etc... (compiler + opt flags), module list
- Scalability : Timings / # of cores => speedup, efficiency, etc...
- Profiling :
  - %MPI / %User code (CPU) / %IO
  - Memory usage
  - I/O : Outputs and inputs
    - File sizes / Number of files / File format, etc...
  - Hardware counters ==> gflops, % of peak, Cache hit/misses, TLB, etc...
  - Load imbalance,communication patterns, obstacles to scaling
  - More ? : OpenMP, tracing...

Better know your machine : cpu/mpi

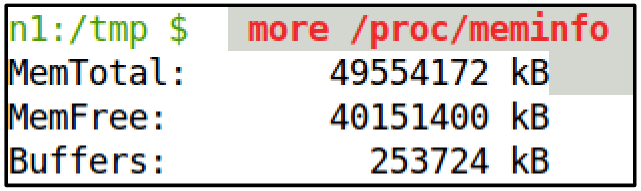
```
n1:/tmp $  more /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 44
model name      : Intel(R) Xeon(R) CPU           X5660  @ 2.80GHz
stepping        : 2
cpu MHz         : 1600.000
cache size      : 12288 KB
```

```
n1:/tmp $  grep processor /proc/cpuinfo
processor       : 0
processor       : 1
processor       : 2
processor       : 3
processor       : 4
processor       : 5
processor       : 6
processor       : 7
processor       : 8
processor       : 9
processor       : 10
processor       : 11
```

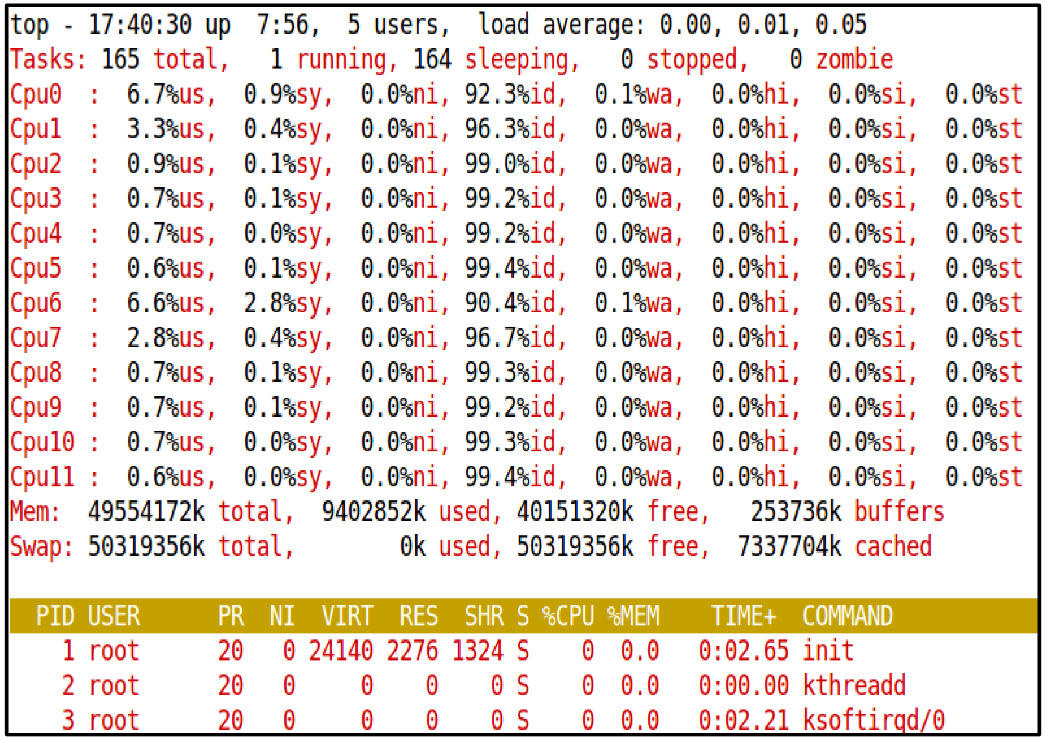

Machine (48GB)
NUMANode #0 (24GB)
Socket #0
L3 #0 (12MB)
L2 #0 (256KB) L2 #1 (256KB) L2 #2 (256KB) L2 #3 (256KB) L2 #4 (256KB) L2 #5 (256KB)
L1 #0 (32KB) L1 #1 (32KB) L1 #2 (32KB) L1 #3 (32KB) L1 #4 (32KB) L1 #5 (32KB)
Core #0 Core #1 Core #2 Core #3 Core #4 Core #5
PU #0 PU #1 PU #2 PU #3 PU #4 PU #5
NUMANode #1 (24GB)
Socket #1
L3 #1 (12MB)
L2 #6 (256KB) L2 #7 (256KB) L2 #8 (256KB) L2 #9 (256KB) L2 #10 (256KB) L2 #11 (256KB)
L1 #6 (32KB) L1 #7 (32KB) L1 #8 (32KB) L1 #9 (32KB) L1 #10 (32KB) L1 #11 (32KB)
Core #6 Core #7 Core #8 Core #9 Core #10 Core #11
PU #6 PU #7 PU #8 PU #9 PU #10 PU #11

HWLOC / LSTOPO

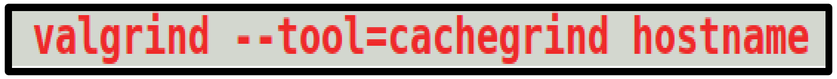
Better know your machine : memory



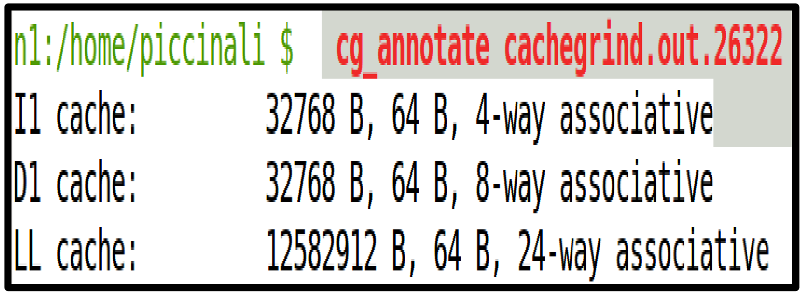
```
n1:/tmp $  more /proc/meminfo
MemTotal:        49554172 kB
MemFree:         40151400 kB
Buffers:           253724 kB
```



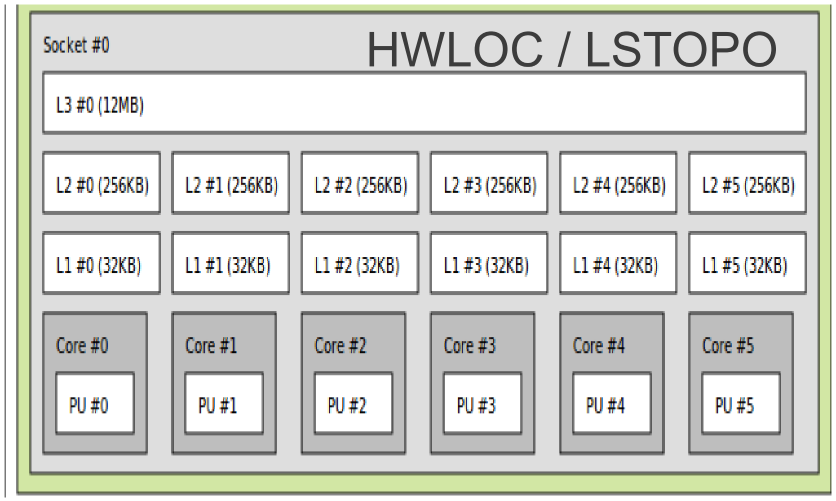
```
top - 17:40:30 up  7:56,  5 users,  load average: 0.00, 0.01, 0.05
Tasks: 165 total,   1 running, 164 sleeping,   0 stopped,   0 zombie
Cpu0  :  6.7%us,  0.9%sy,  0.0%ni, 92.3%id,  0.1%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  3.3%us,  0.4%sy,  0.0%ni, 96.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu2  :  0.9%us,  0.1%sy,  0.0%ni, 99.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.7%us,  0.1%sy,  0.0%ni, 99.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu4  :  0.7%us,  0.0%sy,  0.0%ni, 99.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5  :  0.6%us,  0.1%sy,  0.0%ni, 99.4%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6  :  6.6%us,  2.8%sy,  0.0%ni, 90.4%id,  0.1%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7  :  2.8%us,  0.4%sy,  0.0%ni, 96.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu8  :  0.7%us,  0.1%sy,  0.0%ni, 99.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu9  :  0.7%us,  0.1%sy,  0.0%ni, 99.2%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu10 :  0.7%us,  0.0%sy,  0.0%ni, 99.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu11 :  0.6%us,  0.0%sy,  0.0%ni, 99.4%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  49554172k total,  9402852k used, 40151320k free,   253736k buffers
Swap: 50319356k total,        0k used, 50319356k free,  7337704k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
    1 root      20   0 24140 2276 1324 S    0  0.0   0:02.65 init
    2 root      20   0     0    0    0 S    0  0.0   0:00.00 kthreadd
    3 root      20   0     0    0    0 S    0  0.0   0:02.21 ksoftirqd/0
```



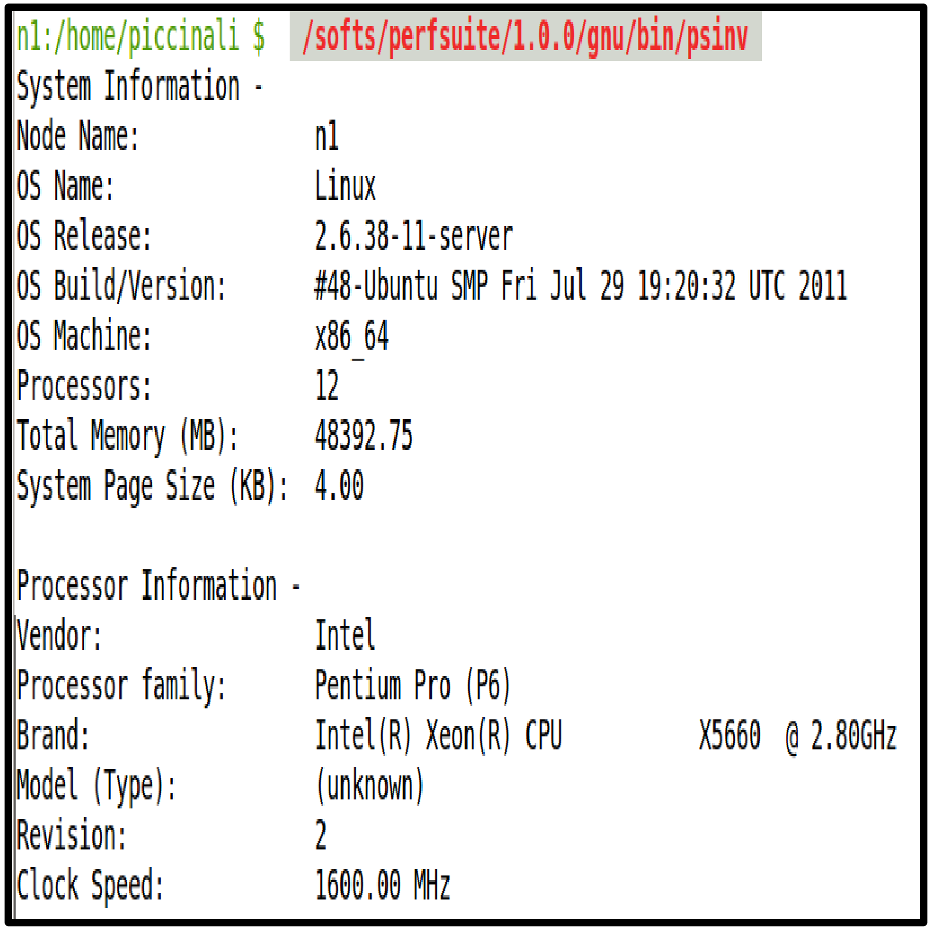
```
valgrind --tool=cachegrind hostname
```



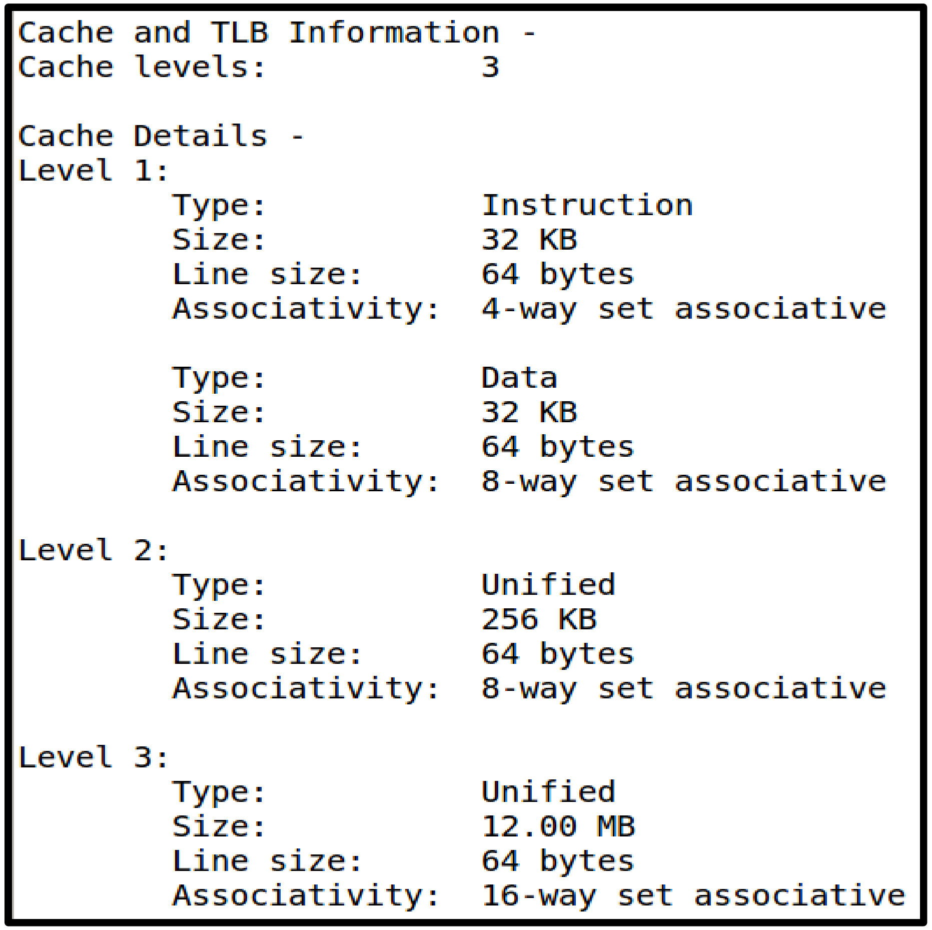
```
n1:/home/piccinali $  cg_annotate cachegrind.out.26322
I1 cache:         32768 B, 64 B, 4-way associative
D1 cache:         32768 B, 64 B, 8-way associative
LL cache:         12582912 B, 64 B, 24-way associative
```



Better know your machine : memory caches (psinv / INTEL)

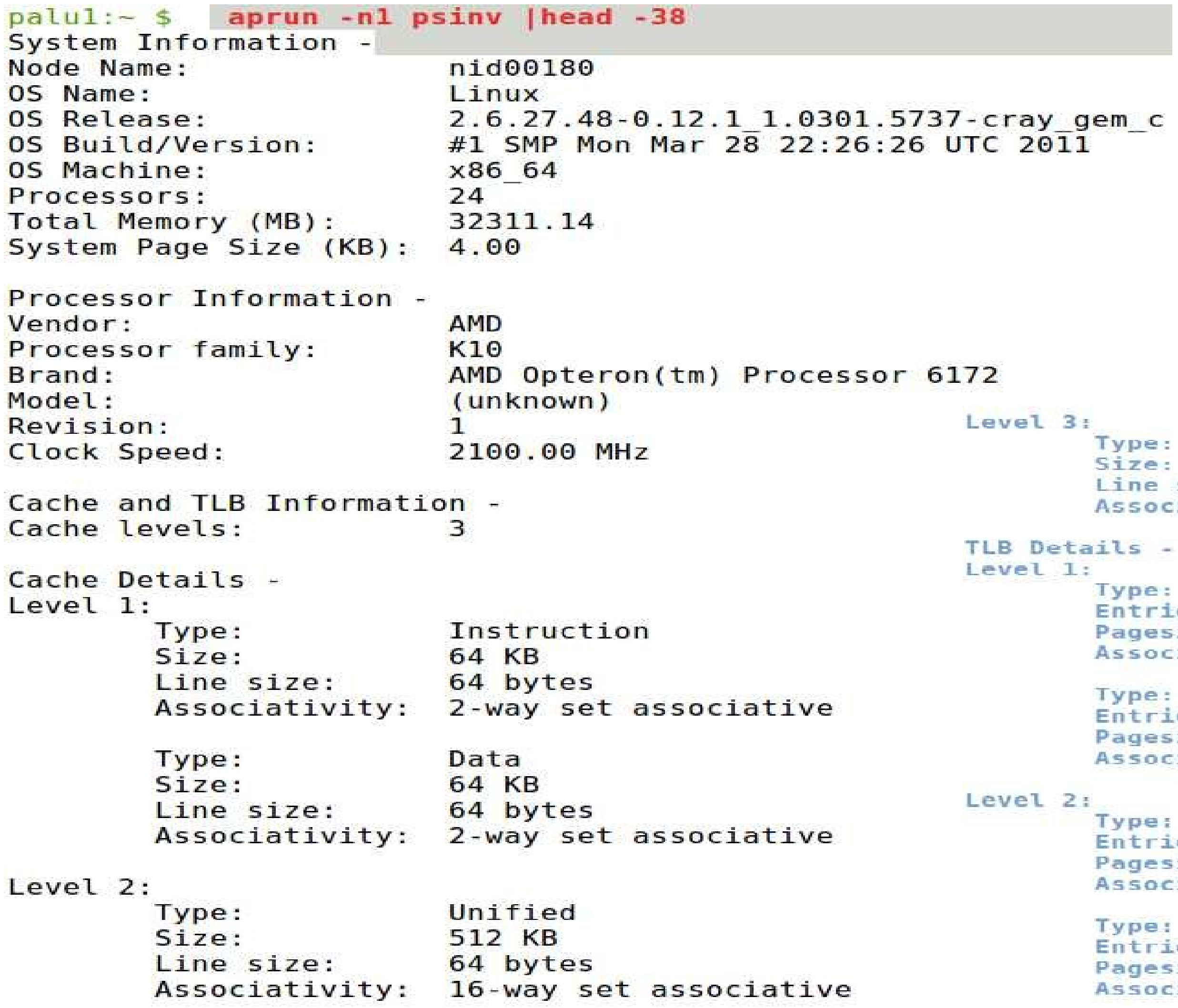


```
n1:/home/piccinali $  /softs/perfsuite/1.0.0/gnu/bin/psinv
System Information -
Node Name:              n1
OS Name:                Linux
OS Release:             2.6.38-11-server
OS Build/Version:       #48-Ubuntu SMP Fri Jul 29 19:20:32 UTC 2011
OS Machine:             x86_64
Processors:             12
Total Memory (MB):      48392.75
System Page Size (KB):  4.00

Processor Information -
Vendor:                 Intel
Processor family:       Pentium Pro (P6)
Brand:                  Intel(R) Xeon(R) CPU          X5660  @ 2.80GHz
Model (Type):           (unknown)
Revision:               2
Clock Speed:            1600.00 MHz
```

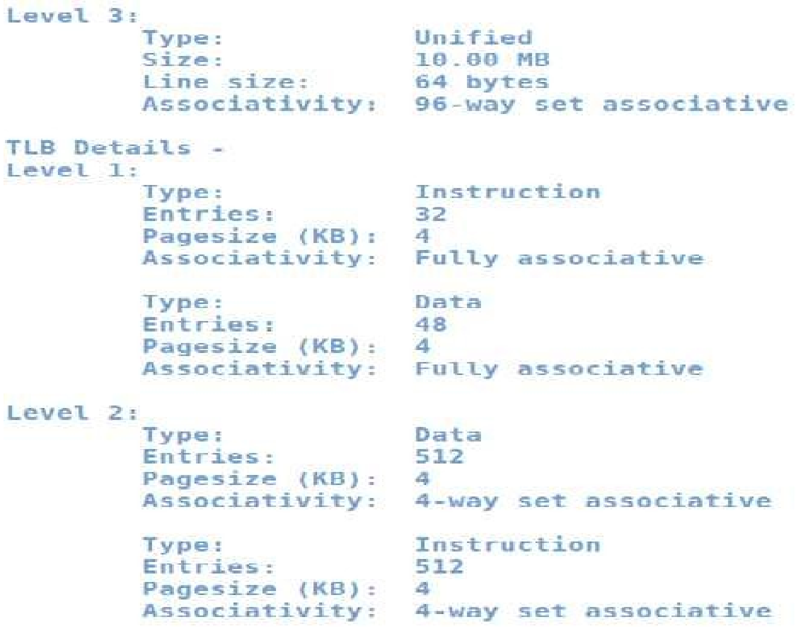


```
Cache and TLB Information -
Cache levels:               3

Cache Details -
Level 1:
        Type:               Instruction
        Size:               32 KB
        Line size:          64 bytes
        Associativity:      4-way set associative

        Type:               Data
        Size:               32 KB
        Line size:          64 bytes
        Associativity:      8-way set associative

Level 2:
        Type:               Unified
        Size:               256 KB
        Line size:          64 bytes
        Associativity:      8-way set associative

Level 3:
        Type:               Unified
        Size:               12.00 MB
        Line size:          64 bytes
        Associativity:      16-way set associative
```

Better know your machine : memory caches (psinv / AMD)

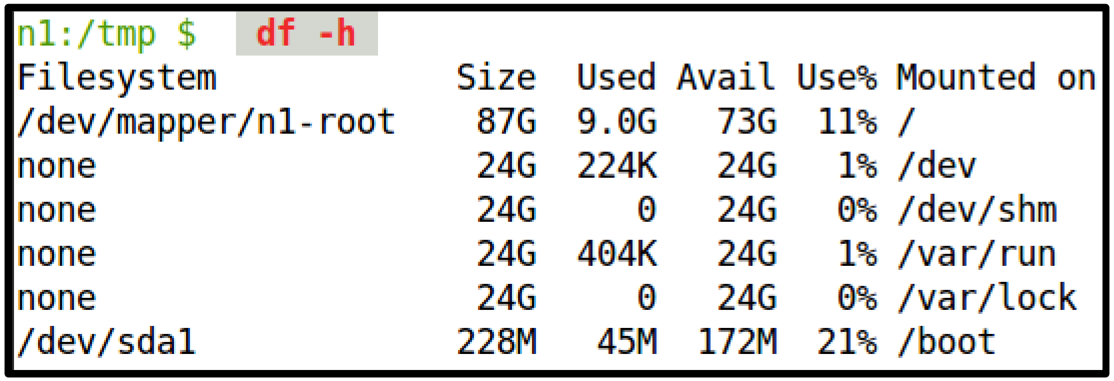


```
palu1:~ $   aprun -n1 psinv |head -38
System Information -
Node Name:                nid00180
OS Name:                  Linux
OS Release:               2.6.27.48-0.12.1_1.0301.5737-cray_gem_c
OS Build/Version:         #1 SMP Mon Mar 28 22:26:26 UTC 2011
OS Machine:               x86_64
Processors:               24
Total Memory (MB):        32311.14
System Page Size (KB):    4.00

Processor Information -
Vendor:                   AMD
Processor family:         K10
Brand:                    AMD Opteron(tm) Processor 6172
Model:                    (unknown)
Revision:                 1
Clock Speed:              2100.00 MHz

Cache and TLB Information -
Cache levels:             3

Cache Details -
Level 1:
        Type:             Instruction
        Size:             64 KB
        Line size:        64 bytes
        Associativity:    2-way set associative

        Type:             Data
        Size:             64 KB
        Line size:        64 bytes
        Associativity:    2-way set associative

Level 2:
        Type:             Unified
        Size:             512 KB
        Line size:        64 bytes
        Associativity:    16-way set associative
```

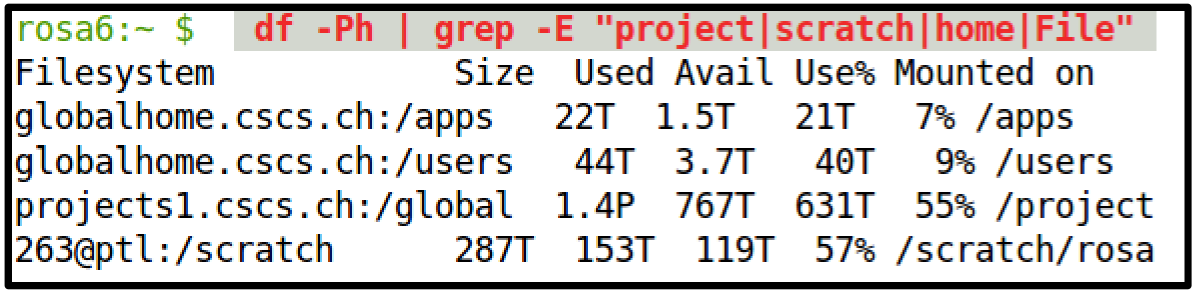


```
Level 3:
        Type:             Unified
        Size:             10.00 MB
        Line size:        64 bytes
        Associativity:    96-way set associative

TLB Details -
Level 1:
        Type:             Instruction
        Entries:          32
        Pagesize (KB):    4
        Associativity:    Fully associative

        Type:             Data
        Entries:          48
        Pagesize (KB):    4
        Associativity:    Fully associative

Level 2:
        Type:             Data
        Entries:          512
        Pagesize (KB):    4
        Associativity:    4-way set associative

        Type:             Instruction
        Entries:          512
        Pagesize (KB):    4
        Associativity:    4-way set associative
```

Better know your machine : i/o



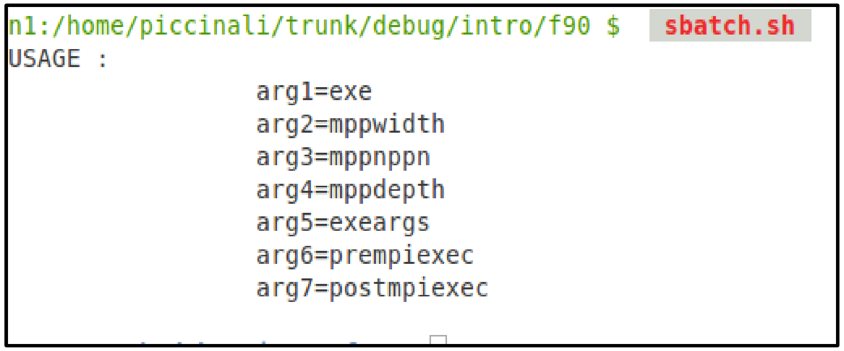
```
n1:/tmp $  df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/mapper/n1-root    87G  9.0G   73G  11% /
none                   24G  224K   24G   1% /dev
none                   24G     0   24G   0% /dev/shm
none                   24G  404K   24G   1% /var/run
none                   24G     0   24G   0% /var/lock
/dev/sda1             228M   45M  172M  21% /boot
```



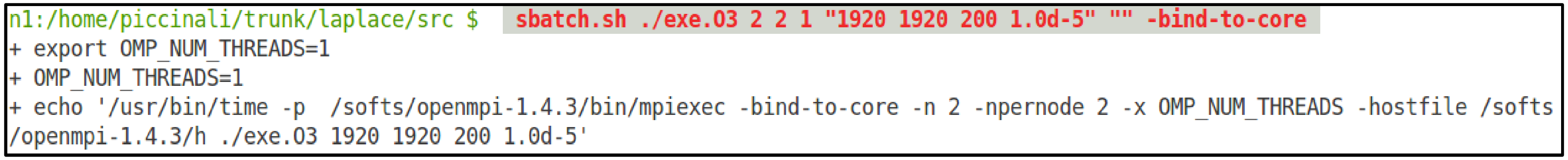
```
rosa6:~ $  df -Ph | grep -E "project|scratch|home|File"
Filesystem            Size  Used Avail Use% Mounted on
globalhome.cscs.ch:/apps   22T  1.5T   21T   7% /apps
globalhome.cscs.ch:/users   44T  3.7T   40T   9% /users
projects1.cscs.ch:/global  1.4P  767T  631T  55% /project
263@ptl:/scratch      287T  153T  119T  57% /scratch/rosa
```

Getting started (1)

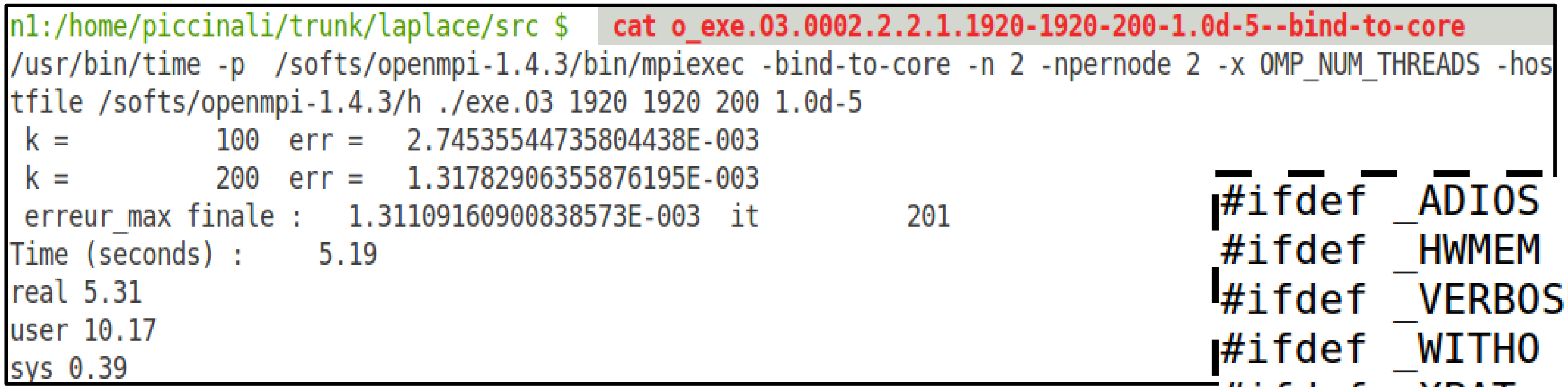
mpif90 Laplace_mpi.F90



```
n1:/home/piccinali/trunk/debug/intro/f90 $  sbatch.sh
USAGE :
                 arg1=exe
                 arg2=mppwidth
                 arg3=mppnppn
                 arg4=mppdepth
                 arg5=exeargs
                 arg6=prempiexec
                 arg7=postmpiexec
```



```
n1:/home/piccinali/trunk/laplace/src $  sbatch.sh ./exe.03 2 2 1 "1920 1920 200 1.0d-5" "" -bind-to-core
+ export OMP_NUM_THREADS=1
+ OMP_NUM_THREADS=1
+ echo '/usr/bin/time -p  /softs/openmpi-1.4.3/bin/mpiexec -bind-to-core -n 2 -npernode 2 -x OMP_NUM_THREADS -hostfile /softs
/openmpi-1.4.3/h ./exe.03 1920 1920 200 1.0d-5'
```



```
n1:/home/piccinali/trunk/laplace/src $  cat o_exe.03.0002.2.2.1.1920-1920-200-1.0d-5--bind-to-core
/usr/bin/time -p  /softs/openmpi-1.4.3/bin/mpiexec -bind-to-core -n 2 -npernode 2 -x OMP_NUM_THREADS -hos
tfile /softs/openmpi-1.4.3/h ./exe.03 1920 1920 200 1.0d-5
 k =          100  err =   2.74535544735804438E-003
 k =          200  err =   1.31782906355876195E-003
 erreur_max finale :   1.31109160900838573E-003  it         201
Time (seconds) :     5.19
real 5.31
user 10.17
sys 0.39
```

```
#ifdef _ADIOS
#ifdef _HWMEM
#ifdef _VERBOSE
#ifdef _WITHO
#ifdef _XPAT
```

Getting started (2)

module permet de modifier l'environnement utilisateur facilement,
en configurant les variables nécessaires à la compilation/exécution.

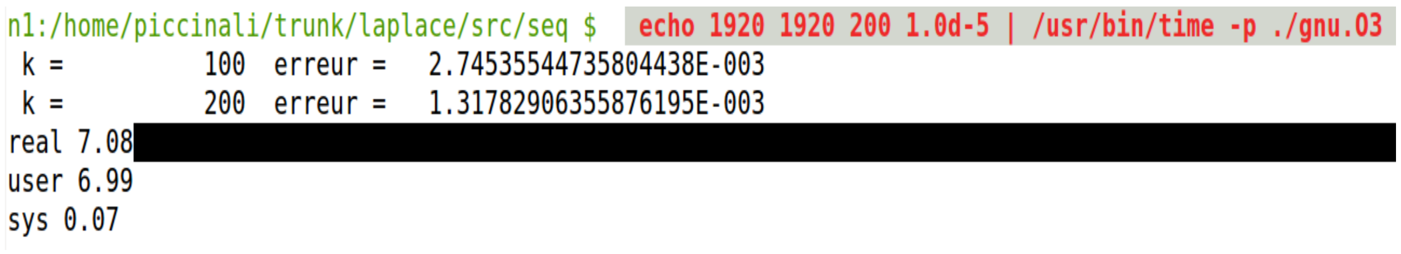
* module avail	: affiche l'environnement disponible
* module list	: affiche l'environnement actuel
* module load <module>	: ajoute application/bibliothèque/compilateur à l'environnement
* module unload <module>	: enlève application/bibliothèque/compilateur de l'environnement
* module show	: affiche le contenu du module
* module swap <module1> <module2>	: remplace version1 avec version2
* module help <module>	: help
* module whatis <module>	: help

http://modules.sf.net

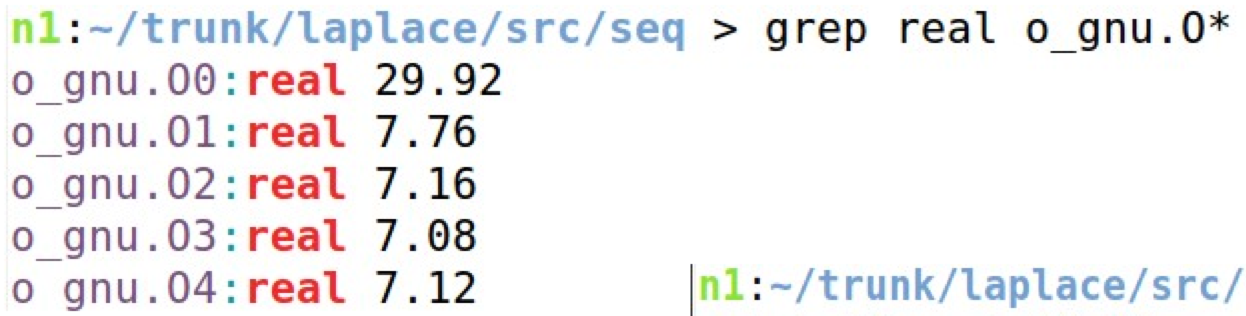
Step0 : Compilation flags

```
gfortran -O3 Laplace_seq.F90 -o gnu.O3
```

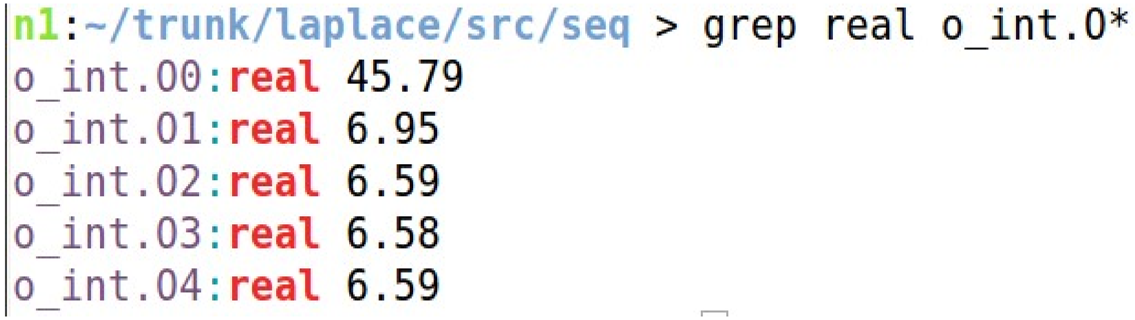
```
ifort -O3 Laplace_seq.F90 -o int.O3
```



```
n1:/home/piccinali/trunk/laplace/src/seq $ echo 1920 1920 200 1.0d-5 | /usr/bin/time -p ./gnu.O3
 k =          100  erreur =   2.74535544735804438E-003
 k =          200  erreur =   1.31782906355876195E-003
real 7.08[redacted]
user 6.99
sys 0.07
```

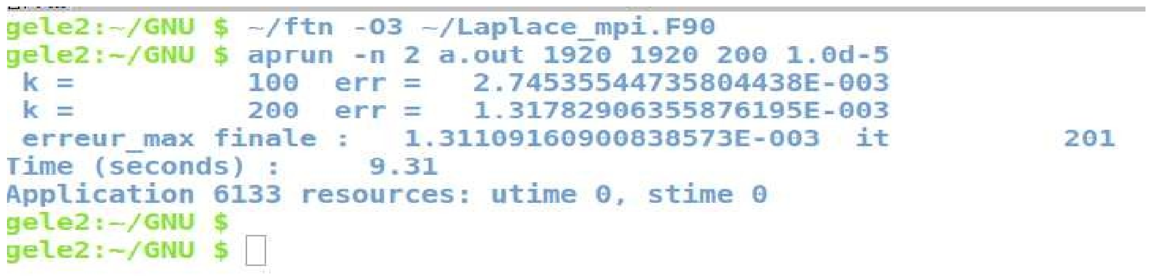


```
n1:~/trunk/laplace/src/seq > grep real o_gnu.O*
o_gnu.O0:real 29.92
o_gnu.O1:real 7.76
o_gnu.O2:real 7.16
o_gnu.O3:real 7.08
o_gnu.O4:real 7.12
```

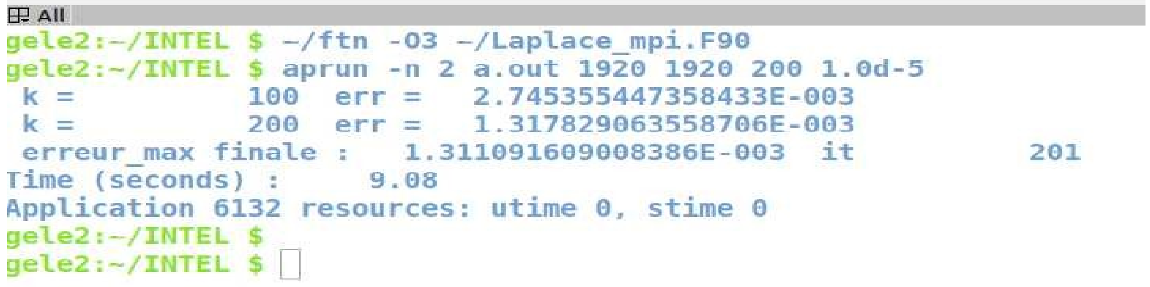


```
n1:~/trunk/laplace/src/seq > grep real o_int.O*
o_int.O0:real 45.79
o_int.O1:real 6.95
o_int.O2:real 6.59
o_int.O3:real 6.58
o_int.O4:real 6.59
```

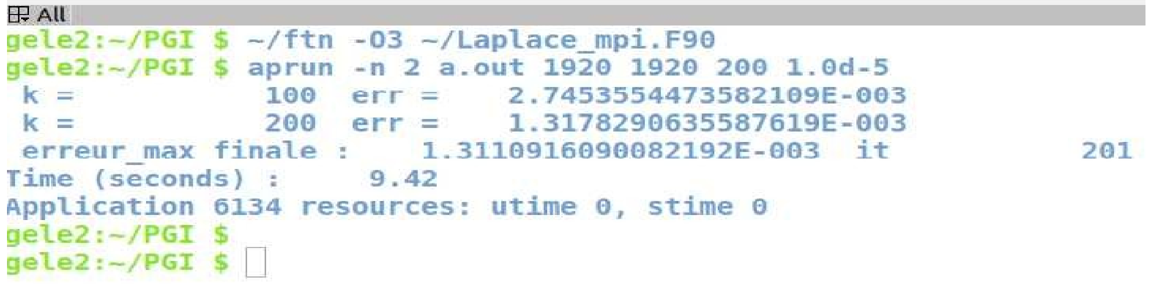
Step0 : Compilers



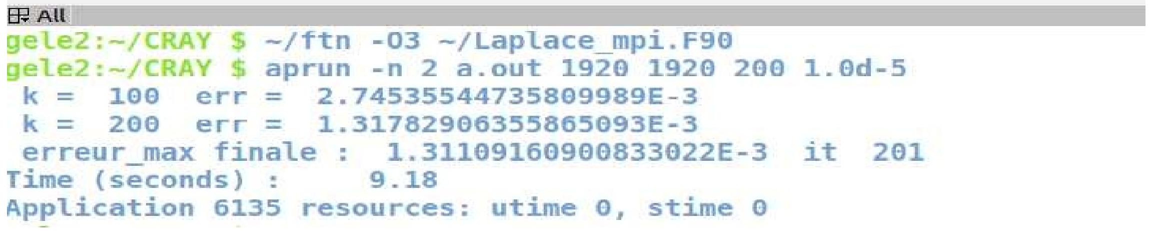
```
gele2:~/GNU $ ~/ftn -O3 ~/Laplace_mpi.F90
gele2:~/GNU $ aprun -n 2 a.out 1920 1920 200 1.0d-5
 k =          100  err =   2.74535544735804438E-003
 k =          200  err =   1.31782906355876195E-003
 erreur_max finale :   1.31109160900838573E-003  it          201
Time (seconds) :     9.31
Application 6133 resources: utime 0, stime 0
gele2:~/GNU $
gele2:~/GNU $
```



```
gele2:~/INTEL $ ~/ftn -O3 ~/Laplace_mpi.F90
gele2:~/INTEL $ aprun -n 2 a.out 1920 1920 200 1.0d-5
 k =          100  err =   2.745355447358433E-003
 k =          200  err =   1.317829063558706E-003
 erreur_max finale :   1.311091609008386E-003  it          201
Time (seconds) :     9.08
Application 6132 resources: utime 0, stime 0
gele2:~/INTEL $
gele2:~/INTEL $
```

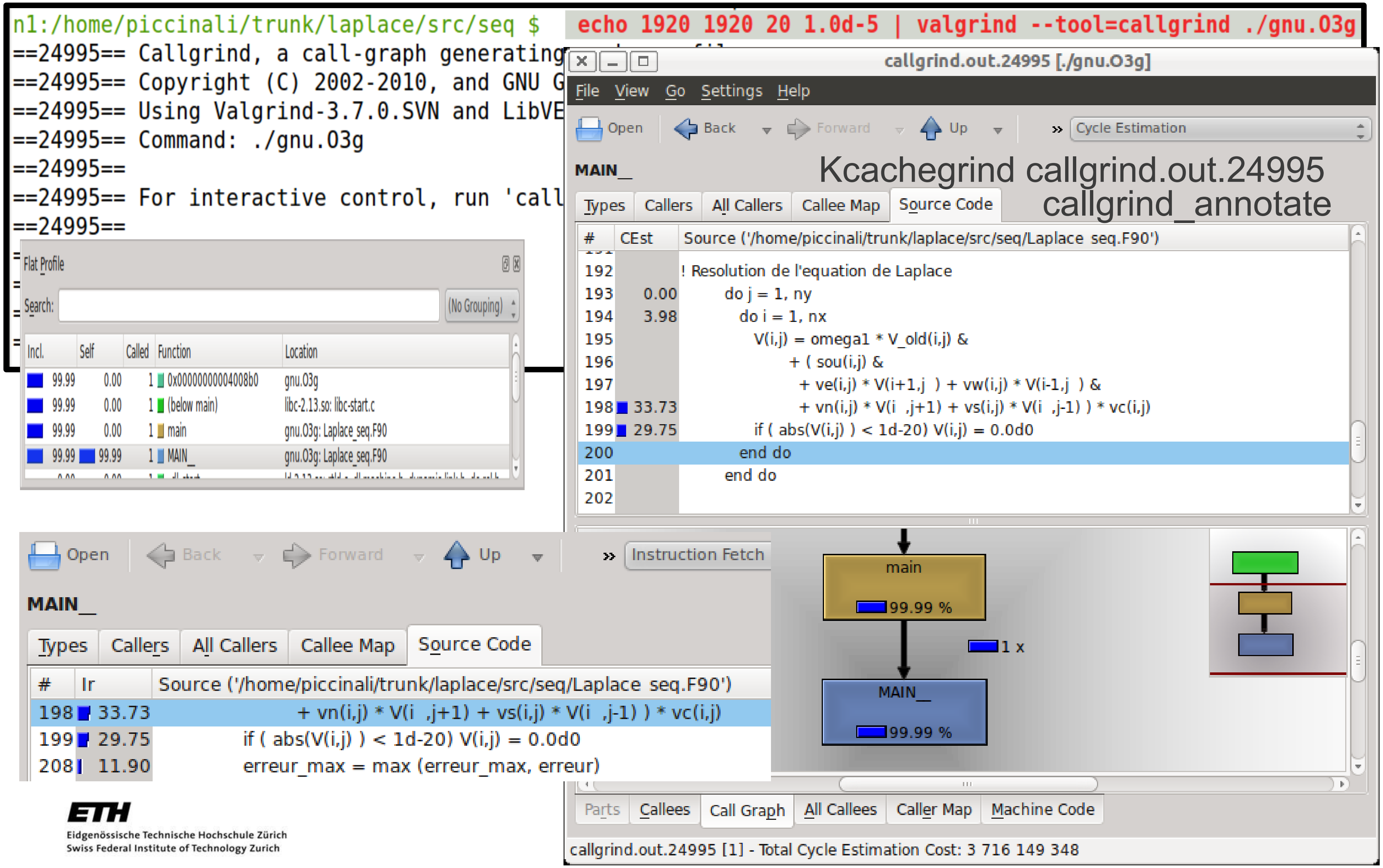


```
gele2:~/PGI $ ~/ftn -O3 ~/Laplace_mpi.F90
gele2:~/PGI $ aprun -n 2 a.out 1920 1920 200 1.0d-5
 k =           100  err =    2.7453554473582109E-003
 k =           200  err =    1.3178290635587619E-003
 erreur_max finale :    1.3110916090082192E-003  it           201
Time (seconds) :     9.42
Application 6134 resources: utime 0, stime 0
gele2:~/PGI $
gele2:~/PGI $
```



```
gele2:~/CRAY $ ~/ftn -O3 ~/Laplace_mpi.F90
gele2:~/CRAY $ aprun -n 2 a.out 1920 1920 200 1.0d-5
 k =  100  err =  2.74535544735809989E-3
 k =  200  err =  1.31782906355865093E-3
 erreur_max finale :  1.31109160900833022E-3  it  201
Time (seconds) :     9.18
Application 6135 resources: utime 0, stime 0
```

Where do I spend time (callgrind) ?



```
n1:/home/piccinali/trunk/laplace/src/seq $ echo 1920 1920 20 1.0d-5 | valgrind --tool=callgrind ./gnu.O3g
==24995== Callgrind, a call-graph generating
==24995== Copyright (C) 2002-2010, and GNU G
==24995== Using Valgrind-3.7.0.SVN and LibVE
==24995== Command: ./gnu.O3g
==24995==
==24995== For interactive control, run 'call
==24995==
```

Flat Profile

Search: (No Grouping)

Incl.	Self	Called	Function	Location
99.99	0.00	1	0x00000000004008b0	gnu.O3g
99.99	0.00	1	(below main)	libc-2.13.so: libc-start.c
99.99	0.00	1	main	gnu.O3g: Laplace_seq.F90
99.99	99.99	1	MAIN_	gnu.O3g: Laplace_seq.F90

callgrind.out.24995 [./gnu.O3g]

Kcachegrind callgrind.out.24995

callgrind_annotate

MAIN_ — Cycle Estimation — Source Code

```
#    CEst    Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
192          ! Resolution de l'equation de Laplace
193   0.00         do j = 1, ny
194   3.98           do i = 1, nx
195                    V(i,j) = omega1 * V_old(i,j) &
196                           + ( sou(i,j) &
197                             + ve(i,j) * V(i+1,j  ) + vw(i,j) * V(i-1,j  ) &
198  33.73                      + vn(i,j) * V(i  ,j+1) + vs(i,j) * V(i  ,j-1) ) * vc(i,j)
199  29.75             if ( abs(V(i,j) ) < 1d-20) V(i,j) = 0.0d0
200                  end do
201                end do
202
```

MAIN_ — Instruction Fetch — Source Code

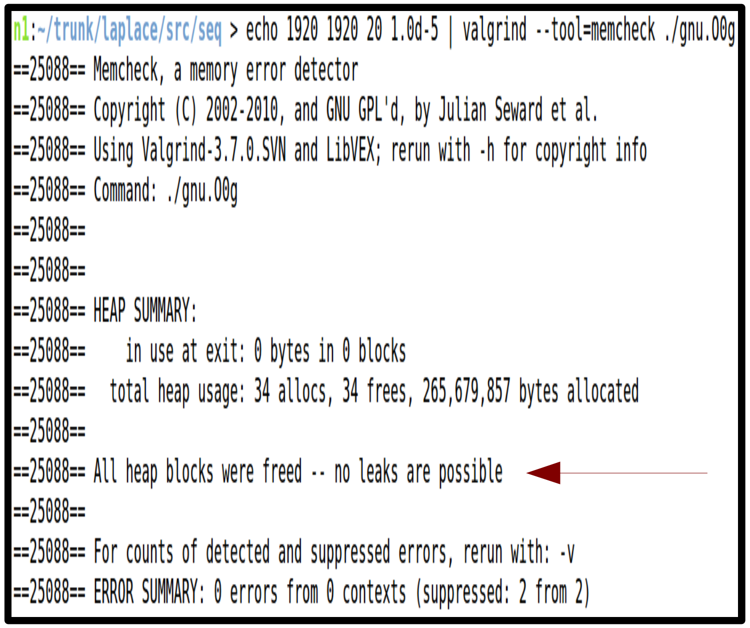
```
#    Ir      Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
198  33.73                      + vn(i,j) * V(i  ,j+1) + vs(i,j) * V(i  ,j-1) ) * vc(i,j)
199  29.75             if ( abs(V(i,j) ) < 1d-20) V(i,j) = 0.0d0
208  11.90             erreur_max = max (erreur_max, erreur)
```

callgrind.out.24995 [1] - Total Cycle Estimation Cost: 3 716 149 348

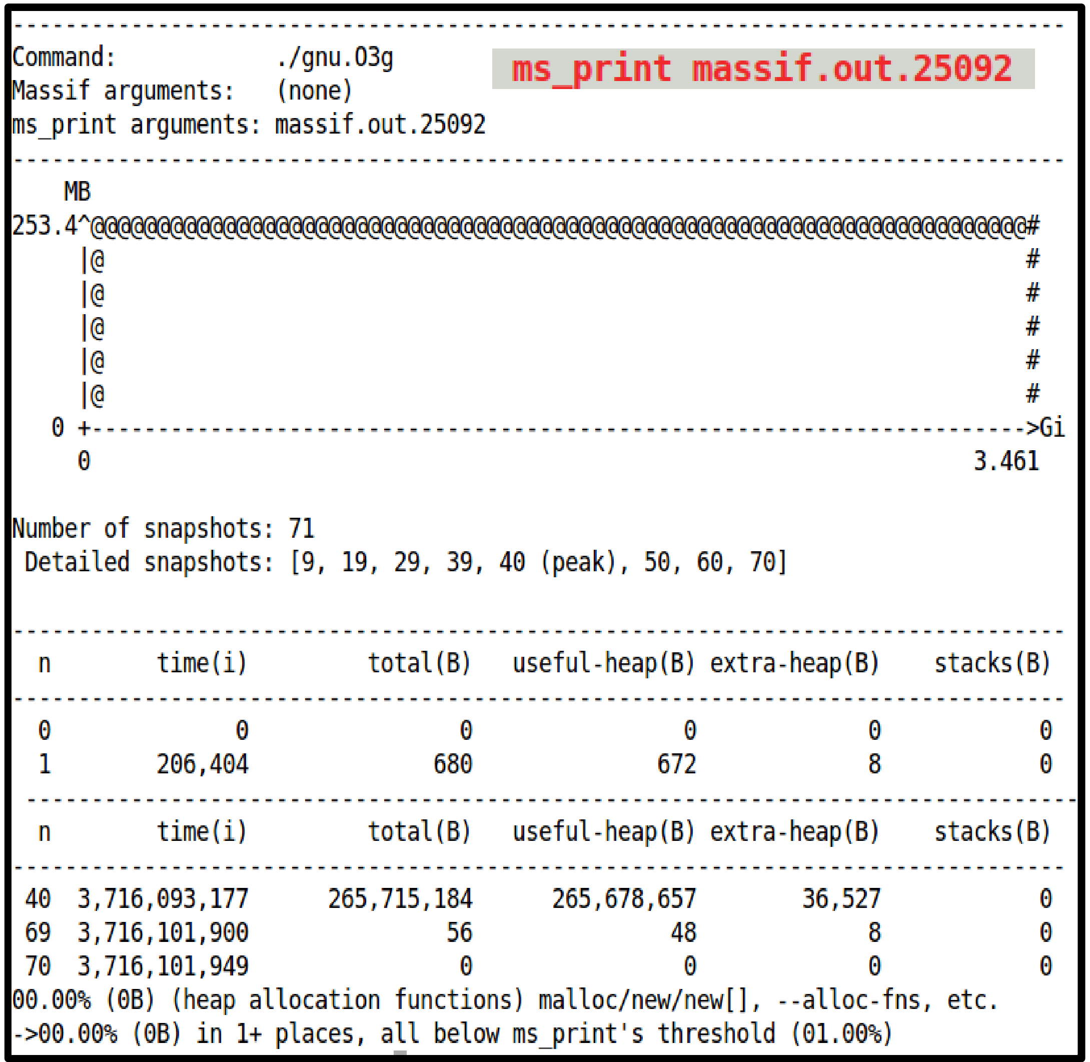
What is my main memory usage (memcheck) ?

```
echo 1920 1920 20 1.0d-5 | valgrind --tool=massif ./gnu.03g
```

```
echo 1920 1920 20 1.0d-5 | valgrind --tool=memcheck ./gnu.00g
```



```
n1:~/trunk/laplace/src/seq > echo 1920 1920 20 1.0d-5 | valgrind --tool=memcheck ./gnu.00g
==25088== Memcheck, a memory error detector
==25088== Copyright (C) 2002-2010, and GNU GPL'd, by Julian Seward et al.
==25088== Using Valgrind-3.7.0.SVN and LibVEX; rerun with -h for copyright info
==25088== Command: ./gnu.00g
==25088==
==25088==
==25088== HEAP SUMMARY:
==25088==     in use at exit: 0 bytes in 0 blocks
==25088==   total heap usage: 34 allocs, 34 frees, 265,679,857 bytes allocated
==25088==
==25088== All heap blocks were freed -- no leaks are possible
==25088==
==25088== For counts of detected and suppressed errors, rerun with: -v
==25088== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 2 from 2)
```



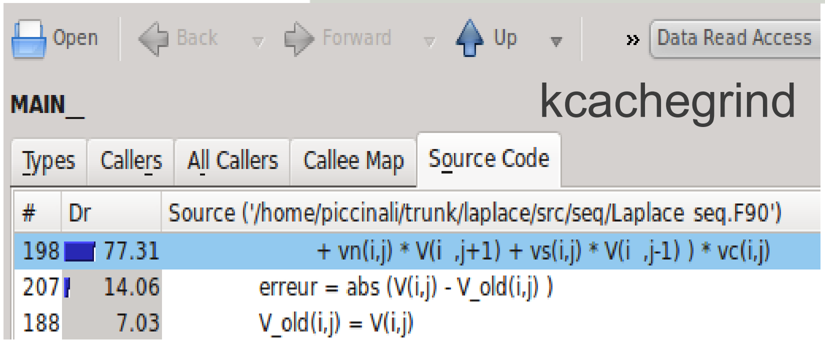
```
ms_print massif.out.25092
```

```
--------------------------------------------------------------------------------
Command:            ./gnu.03g
Massif arguments:   (none)
ms_print arguments: massif.out.25092
--------------------------------------------------------------------------------
    MB
253.4^@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@#
     |@                                                                       #
     |@                                                                       #
     |@                                                                       #
     |@                                                                       #
     |@                                                                       #
   0 +----------------------------------------------------------------------->Gi
     0                                                                   3.461

Number of snapshots: 71
 Detailed snapshots: [9, 19, 29, 39, 40 (peak), 50, 60, 70]

--------------------------------------------------------------------------------
  n        time(i)         total(B)   useful-heap(B) extra-heap(B)    stacks(B)
--------------------------------------------------------------------------------
  0              0                0                0             0            0
  1        206,404              680              672             8            0
--------------------------------------------------------------------------------
  n        time(i)         total(B)   useful-heap(B) extra-heap(B)    stacks(B)
--------------------------------------------------------------------------------
 40  3,716,093,177      265,715,184      265,678,657        36,527            0
 69  3,716,101,900               56               48             8            0
 70  3,716,101,949                0                0             0            0
00.00% (0B) (heap allocation functions) malloc/new/new[], --alloc-fns, etc.
->00.00% (0B) in 1+ places, all below ms_print's threshold (01.00%)
```

What is my L1/L3 cache usage (cachegrind) ?

```
n1:/home/piccinali/trunk/laplace/src/seq $  echo 1920 1920 20 1.0d-5 | valgrind --tool=cachegrind ./gnu.03g
==25248== Cachegrind, a cache and branch-prediction profiler
==25248== Copyright (C) 2002-2010, and GNU GPL'd, by Nicholas Nethercote et al.
==25248== Using Valgrind-3.7.0.SVN and LibVEX; rerun with -h for copyright info
==25248== Command: ./gnu.03g
==25248==
--25248-- warning: L3 cache found, using its data for the LL simulation.
--25248-- warning: pretending that LL cache has associativity 24 instead of actual 16
==25248==
==25248== I   refs:      3,716,149,724
==25248== I1  misses:            1,395
==25248== LLi misses:            1,369
==25248== I1  miss rate:          0.00%
==25248== LLi miss rate:          0.00%
==25248==
==25248== D   refs:      1,269,911,041  (1,048,976,325 rd   + 220,934,716 wr)
==25248== D1  misses:      135,328,344  (  121,463,449 rd   +  13,864,895 wr)
==25248== LLd misses:      115,735,040  (  101,907,212 rd   +  13,827,828 wr)
==25248== D1  miss rate:          10.6% (         11.5%     +         6.2%  )
==25248== LLd miss rate:           9.1% (          9.7%     +         6.2%  )
==25248==
==25248== LL refs:         135,329,739  (  121,464,844 rd   +  13,864,895 wr)
==25248== LL misses:       115,736,409  (  101,908,581 rd   +  13,827,828 wr)
==25248== LL miss rate:            2.3% (          2.1%     +         6.2%  )
```

```
n1:/home/piccinali/trunk/laplace/src/seq $  cg_annotate --auto=yes cachegrind.out.25248
--------------------------------------------------------------------------------
I1 cache:         32768 B, 64 B, 4-way associative
D1 cache:         32768 B, 64 B, 8-way associative
LL cache:         12582912 B, 64 B, 24-way associative
Command:          ./gnu.03g
Data file:        cachegrind.out.25248
Events recorded:  Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Events shown:     Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Event sort order: Ir I1mr ILmr Dr D1mr DLmr Dw D1mw DLmw
Thresholds:       0.1 100 100 100 100 100 100 100 100
Include dirs:
User annotated:
Auto-annotation:  on

--------------------------------------------------------------------------------
           Ir  I1mr  ILmr            Dr        D1mr        DLmr          Dw      D1mw        DLmw
--------------------------------------------------------------------------------
3,716,149,724 1,395 1,369 1,048,976,325 121,463,449 101,907,212 220,934,716 13,864,895 13,827,828  PROGRAM TOTALS
```



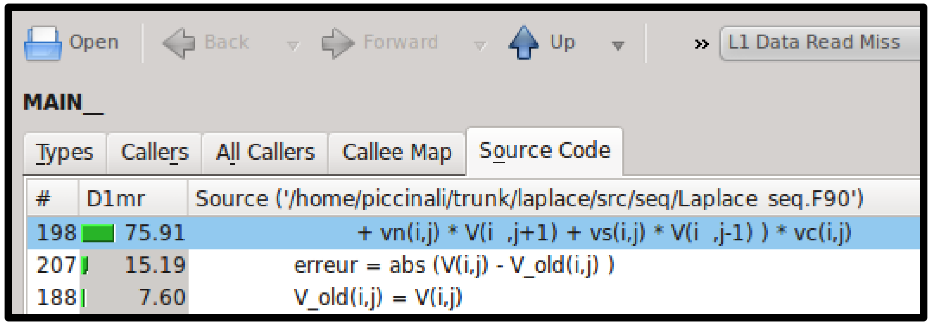
kcachegrind

```
#    Dr      Source ('/home/piccinali/trunk/laplace/src/seq/Laplace seq.F90')
198  77.31        + vn(i,j) * V(i  ,j+1) + vs(i,j) * V(i  ,j-1) ) * vc(i,j)
207  14.06     erreur = abs (V(i,j) - V_old(i,j) )
188   7.03     V_old(i,j) = V(i,j)
```

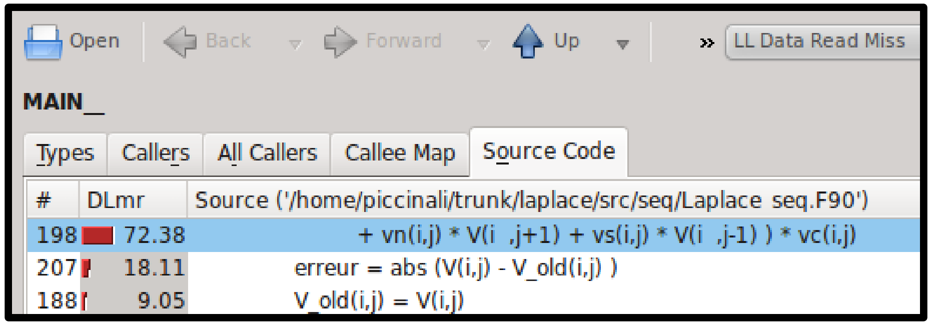
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

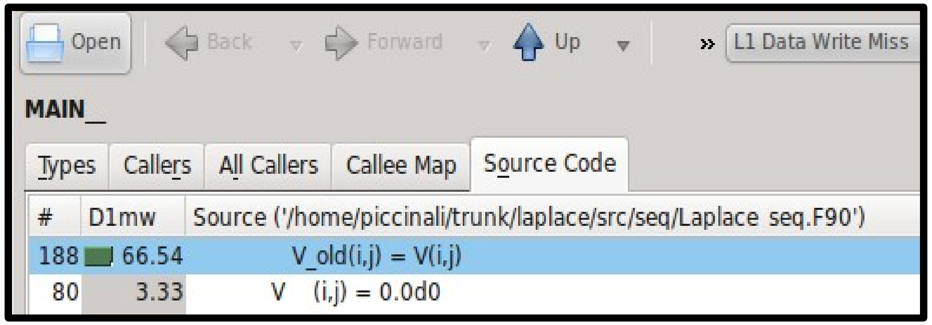
Where is my L1/L3 cache usage (kcachegrind) ?



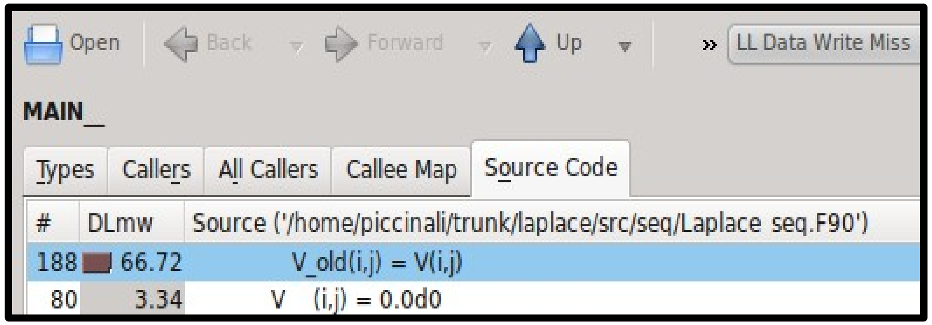
#	D1mr	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
198 🟩	75.91	+ vn(i,j) * V(i ,j+1) + vs(i,j) * V(i ,j-1)) * vc(i,j)
207 🟩	15.19	erreur = abs (V(i,j) - V_old(i,j))
188 🟩	7.60	V_old(i,j) = V(i,j)



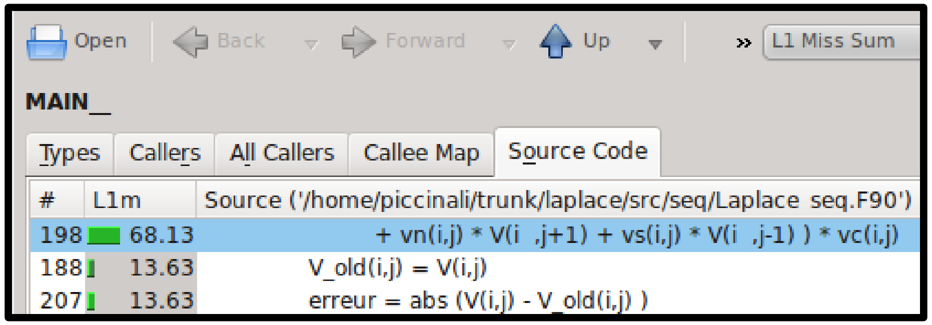
#	DLmr	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
198 🟥	72.38	+ vn(i,j) * V(i ,j+1) + vs(i,j) * V(i ,j-1)) * vc(i,j)
207 🟥	18.11	erreur = abs (V(i,j) - V_old(i,j))
188 🟥	9.05	V_old(i,j) = V(i,j)



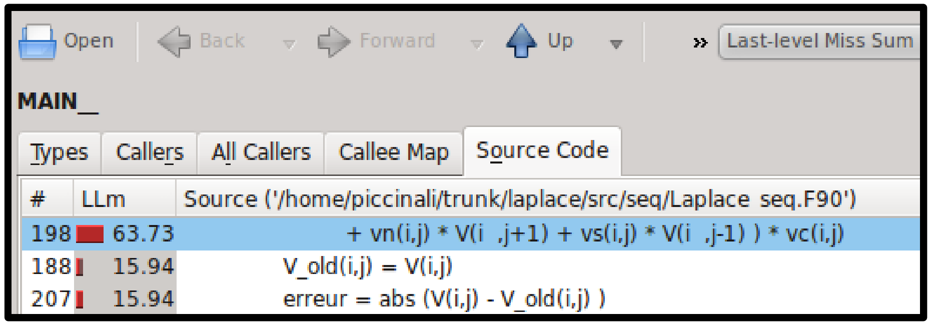
#	D1mw	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
188 🟩	66.54	V_old(i,j) = V(i,j)
80	3.33	V (i,j) = 0.0d0



#	DLmw	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
188 🟥	66.72	V_old(i,j) = V(i,j)
80	3.34	V (i,j) = 0.0d0



#	L1m	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
198 🟩	68.13	+ vn(i,j) * V(i ,j+1) + vs(i,j) * V(i ,j-1)) * vc(i,j)
188 🟩	13.63	V_old(i,j) = V(i,j)
207 🟩	13.63	erreur = abs (V(i,j) - V_old(i,j))



#	LLm	Source ('/home/piccinali/trunk/laplace/src/seq/Laplace_seq.F90')
198 🟥	63.73	+ vn(i,j) * V(i ,j+1) + vs(i,j) * V(i ,j-1)) * vc(i,j)
188 🟥	15.94	V_old(i,j) = V(i,j)
207 🟥	15.94	erreur = abs (V(i,j) - V_old(i,j))

TP : matmul

Objectif :

* Montrer l'importance des indices de boucles sur la performance du code

Instructions :

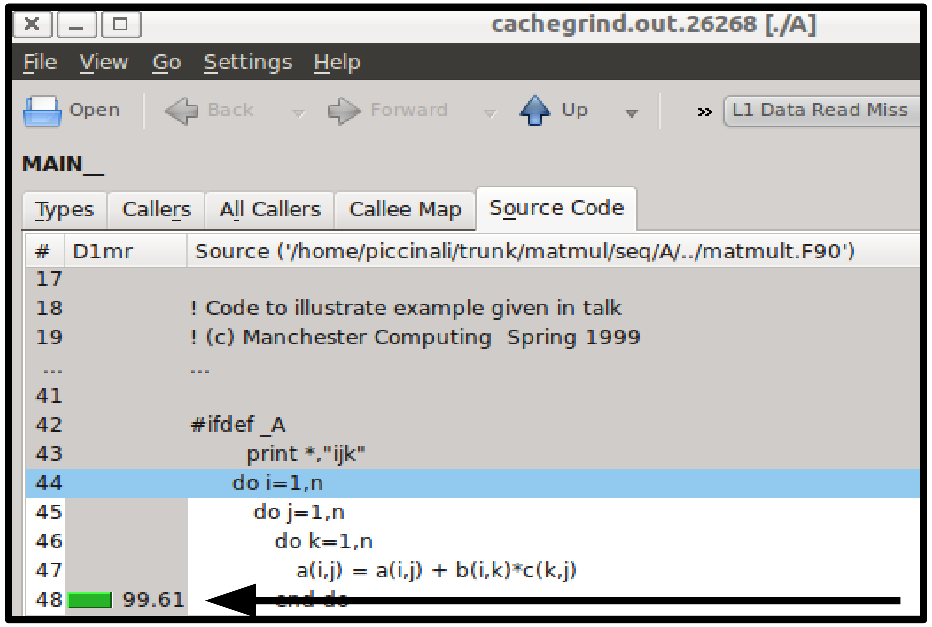
* cp /home/piccinali/trunk/matmul/seq/matmult.F90 $HOME ; cd $HOME
* Compilez le programme : gfortran -D_A -g -O3 matmult.F90 -o A ;
* Executez le programme et notez le temps d'execution.
* Recommencez avec -D_B,-D_C,-D_D,-D_E,-D_F et comparez les resultats.
* Utilisez cachegrind pour trouver une explication

TP : matmul (solution)

```
gfortran -D_A -g -O3 matmult.F90 -o A
```

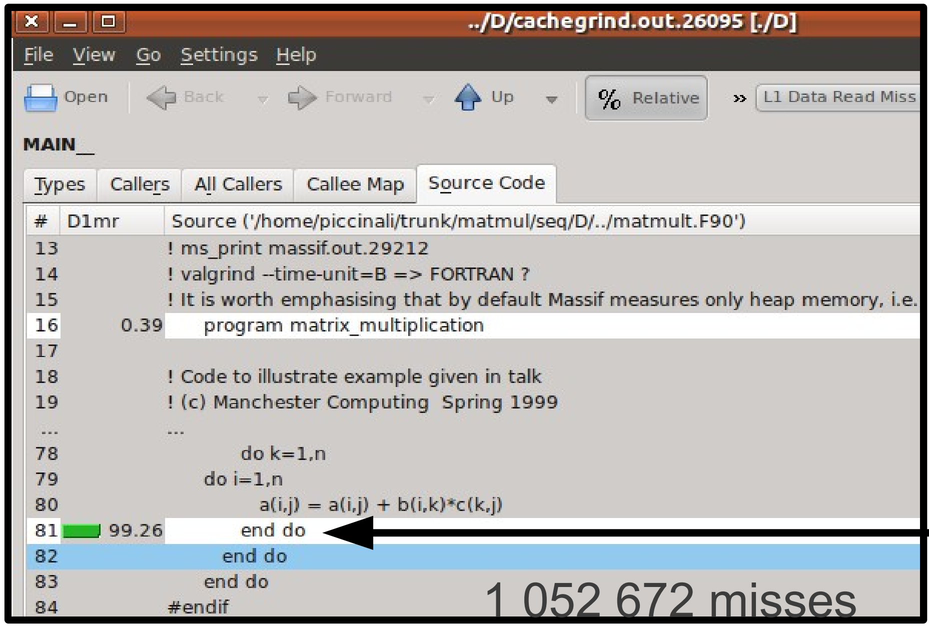
```
/usr/bin/time -p ./A
```

```
A/oo.A:real  80.51
B/oo.B:real  155.29
C/oo.C:real  78.16
D/oo.D:real  3.21
E/oo.E:real  155.73
F/oo.F:real  4.07
```


cachegrind.out.26268 [./A]
File View Go Settings Help
Open Back Forward Up
L1 Data Read Miss
MAIN__
Types Callers All Callers Callee Map Source Code

```
#   D1mr     Source ('/home/piccinali/trunk/matmul/seq/A/../matmult.F90')
17
18           ! Code to illustrate example given in talk
19           ! (c) Manchester Computing  Spring 1999
...          ...
41
42           #ifdef _A
43                   print *,"ijk"
44                 do i=1,n
45                   do j=1,n
46                     do k=1,n
47                       a(i,j) = a(i,j) + b(i,k)*c(k,j)
48  🟩 99.61              end do
```

17 825 792 misses


../D/cachegrind.out.26095 [./D]
File View Go Settings Help
Open Back Forward Up
Relative
L1 Data Read Miss
MAIN__
Types Callers All Callers Callee Map Source Code

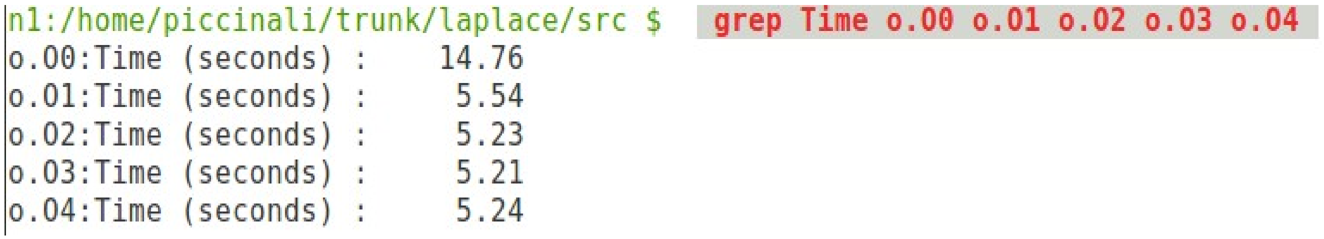
```
#   D1mr      Source ('/home/piccinali/trunk/matmul/seq/D/../matmult.F90')
13            ! ms_print massif.out.29212
14            ! valgrind --time-unit=B => FORTRAN ?
15            ! It is worth emphasising that by default Massif measures only heap memory, i.e.
16     0.39     program matrix_multiplication
17
18            ! Code to illustrate example given in talk
19            ! (c) Manchester Computing  Spring 1999
...           ...
78                   do k=1,n
79                do i=1,n
80                      a(i,j) = a(i,j) + b(i,k)*c(k,j)
81  🟩 99.26          end do
82                  end do
83                end do
84            #endif
```

1 052 672 misses

Step0 : MPI flags

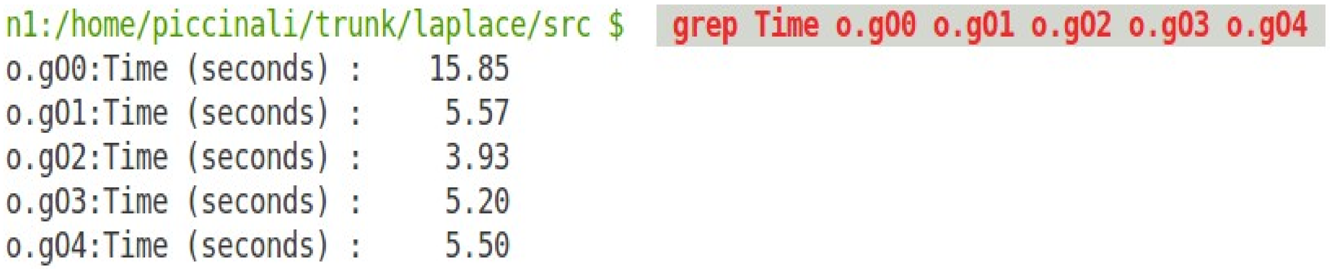
```
/softs/openmpi-1.4.3/bin/mpiexec -n 2 -bind-to-core -npernode 2 -hostfile /softs/openmpi-1.4.3/h ./exe.03 1920 1920 200 1.0d-5
```

-bind-to-core



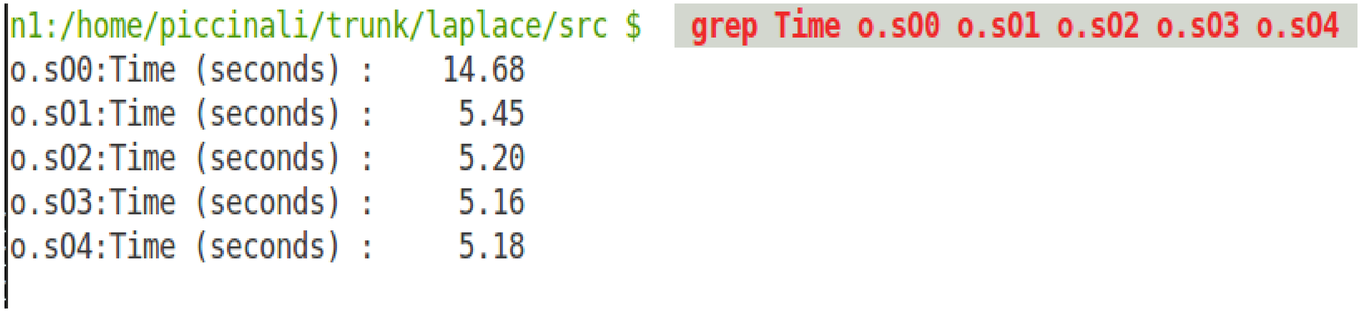
```
n1:/home/piccinali/trunk/laplace/src $ grep Time o.00 o.01 o.02 o.03 o.04
o.00:Time (seconds) :    14.76
o.01:Time (seconds) :     5.54
o.02:Time (seconds) :     5.23
o.03:Time (seconds) :     5.21
o.04:Time (seconds) :     5.24
```

bind-to-none



```
n1:/home/piccinali/trunk/laplace/src $ grep Time o.g00 o.g01 o.g02 o.g03 o.g04
o.g00:Time (seconds) :    15.85
o.g01:Time (seconds) :     5.57
o.g02:Time (seconds) :     3.93
o.g03:Time (seconds) :     5.20
o.g04:Time (seconds) :     5.50
```

bind-to-socket



```
n1:/home/piccinali/trunk/laplace/src $ grep Time o.s00 o.s01 o.s02 o.s03 o.s04
o.s00:Time (seconds) :    14.68
o.s01:Time (seconds) :     5.45
o.s02:Time (seconds) :     5.20
o.s03:Time (seconds) :     5.16
o.s04:Time (seconds) :     5.18
```

Step1 : Scalability study

Speedup = t1 / tn

t1 : temps de l'agorithme sequentiel,

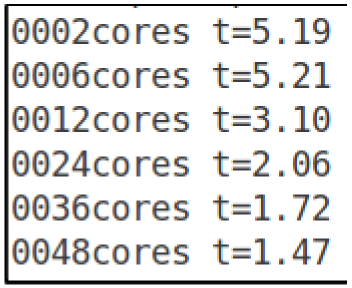
tn : temps de l'agorithme parallele sur n processeurs,

Linear speedup : acceleration lineaire avec le nombre de cpus (rare mais possible)

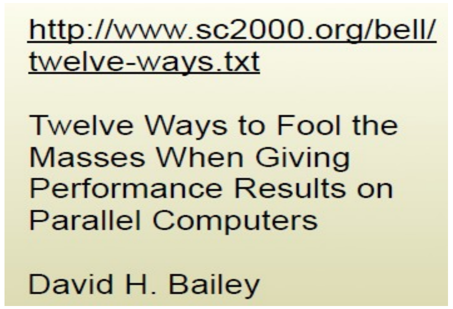
Efficiency : Speedup / n

Weak scaling : la taille du probleme augmente avec le nombre de cpus

Strong scaling : la taille du probleme ne varie pas en fonction du nombre de cpus (harder)



```
0002cores t=5.19
0006cores t=5.21
0012cores t=3.10
0024cores t=2.06
0036cores t=1.72
0048cores t=1.47
```



http://www.sc2000.org/bell/twelve-ways.txt

Twelve Ways to Fool the Masses When Giving Performance Results on Parallel Computers

David H. Bailey

Step1 : Loi Amdahl (1967)

$$\textbf{Speedup} = \frac{1}{(1-P) + \frac{P}{N}}$$

P = fraction // du code ; 1-P = fraction non // du code

N = nombre de cpus

Corollaire 1 :

=> si $N \rightarrow +\infty$

=> alors Speedup <= 1/(1-P)

=> performance majoree par la partie non parallele du code (1-P), independemment du nombre de cpus :-(

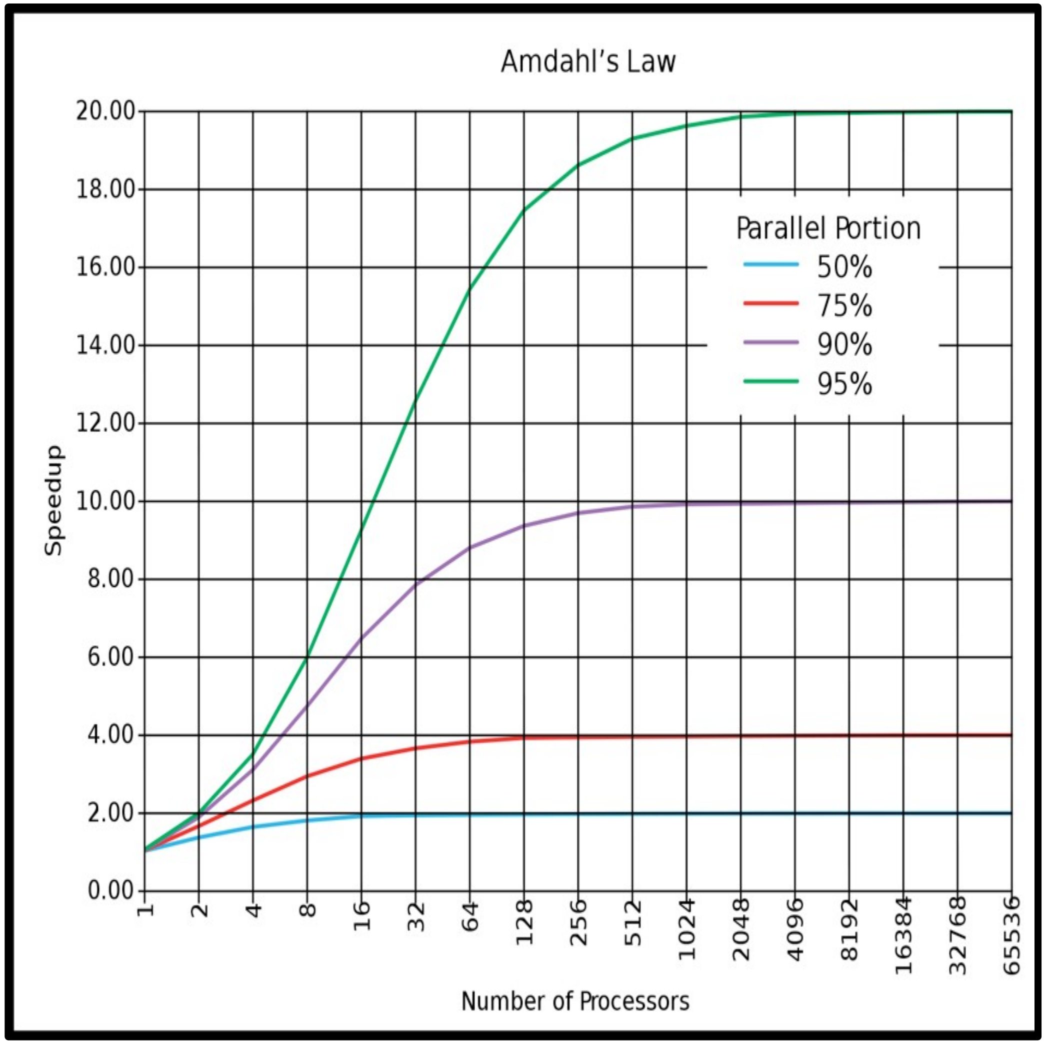
=> ex : si P=90% alors Speedup <= 10

Corollaire 2 :

=> estimation de P a partir du Speedup

=> P ~= (1-Sp)*N / (Sp * (1-N))

(avec N=cores)



How to time MPI programs ?

Use the linux time command :

```
/usr/bin/time -p aprun -n12 exe
```

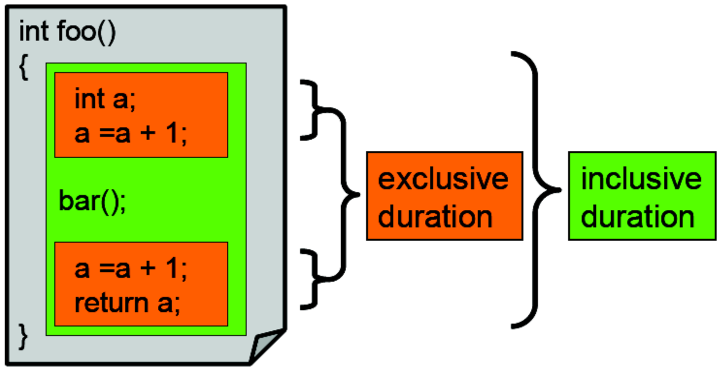
When job finishes, time writes timing statistics :

- User CPU time
  - It’s the time spent executing the timed command,
- System CPU time
  - It’s the time spent executing system calls on behalf of your program,
- Real “wallclock” time = total time
  - It's the total elapsed time taken by the job.

Profiling : Inclusive vs. Exclusive

```
int main( )
{ /* takes 100 secs */
   f1(); /* takes 20 secs */
  /* other work */
   f2(); /* takes 50 secs */
   f1(); /* takes 20 secs */

  /* other work */
}

/* similar for other metrics, such
as hardware performance counters,
etc. */
```

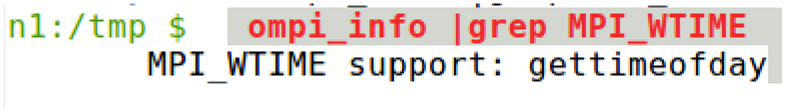
- **Inclusive** time for **main**
  - 100 secs
- **Exclusive** time for **main**
  - 100-20-50-20=10 secs
- Exclusive time sometimes called “self”



How to time MPI regions ?

The elapsed (wall-clock) time between two points in an MPI program can be computed using **MPI_Wtime**:

```
double precision :: t1, t2
t1 = MPI_Wtime()
...
t2 = MPI_Wtime()
print *, 'time is ', t2-t1
```



```
n1:/tmp $  ompi_info |grep MPI_WTIME
          MPI_WTIME support: gettimeofday
```

MPI_Wtick returns the resolution of MPI_Wtime in seconds, i.e the number of seconds between successive hardware clockticks.

Performance tools allow you to measure time without modifying your src code...

Step2 : Profiling/Sampling vs Tracing

Level of performance information depends on 2 types of experiments :

- Sampling experiments :
  - captures values at specified time intervals or when a specified counter overflows,
  - provides a summary of performance events and timings for the execution as a whole,

- Tracing experiments :
  - count some event such as the number of times a specific system call is executed,
  - records the chronology of events,
  - amount of data increases with the runtime, and can lead to rather large tracing files,

A profile is sufficient to pinpoint load imbalance due to problem decomposition and/or identify the origin of excessive communication time.

A trace is useful for detailed examination of timing issues occurring within a code.

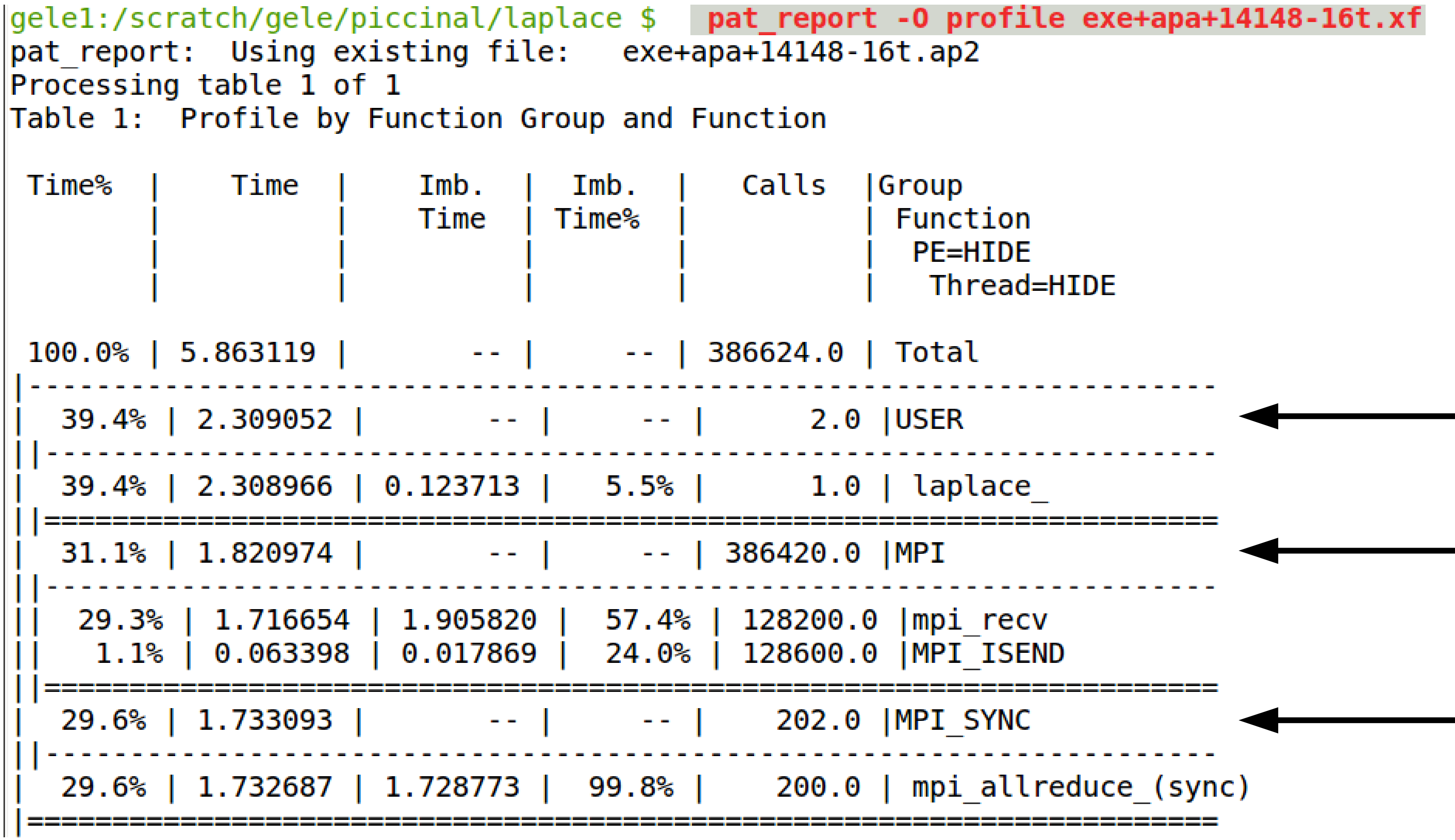
Overview of some tools

* opensrc :
  > scalasca
  > tau
* vampir
* vendors :
  > cray, ibm, sgi, etc...

XT-CRAYPAT : Features

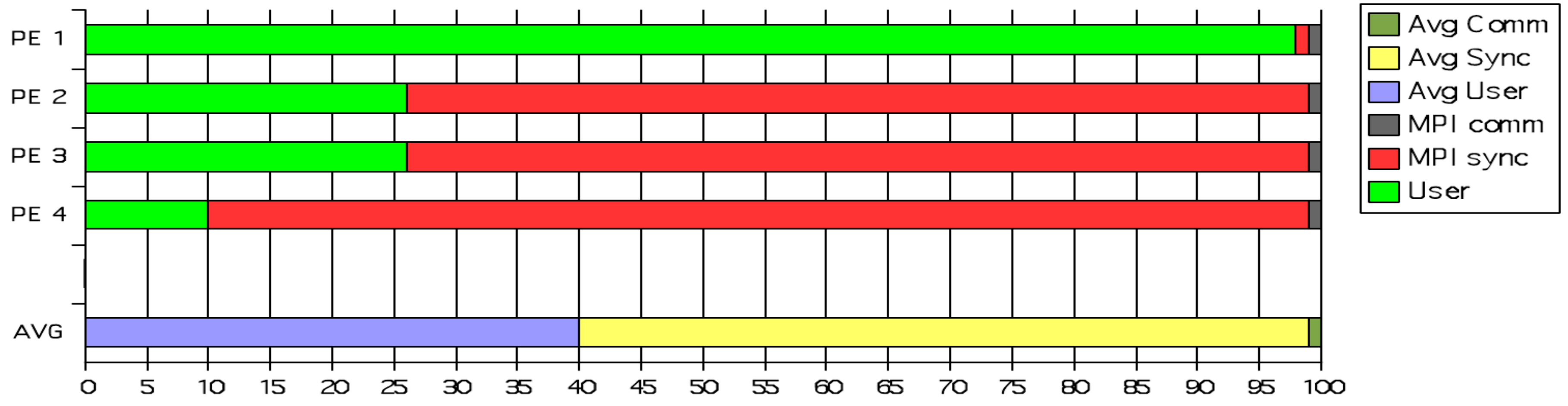
- Craypat basics
  - Developped by CRAY (Luiz DeRose)
  - Multiple functionalities
    - Most consuming routines
    - Load balance across computing resources
    - Communication overhead and Cache utilization
    - FLOPS and HW counters
    - SSE instructions (Vectorization)
    - Ratio of computation vs communication
  - Integrated and easy to use
    - pat_build : utility to instrument the application
    - pat_report : utility to create performance report
    - pat_help : provides craypat infos
    - apprentice2 : graphical performance analysis tool

Step2 : pat_report (imbalance)



```
gele1:/scratch/gele/piccinal/laplace $  pat_report -O profile exe+apa+14148-16t.xf
pat_report:  Using existing file:   exe+apa+14148-16t.ap2
Processing table 1 of 1
Table 1:  Profile by Function Group and Function

 Time%   |     Time   |     Imb.  |  Imb.   |    Calls   |Group
         |            |     Time  | Time%   |            | Function
         |            |           |         |            |  PE=HIDE
         |            |           |         |            |   Thread=HIDE

 100.0% | 5.863119 |        -- |      -- | 386624.0 | Total
|-----------------------------------------------------------------------
|   39.4% | 2.309052 |        -- |      -- |      2.0 |USER
||----------------------------------------------------------------------
|   39.4% | 2.308966 | 0.123713 |    5.5% |      1.0 | laplace_
||======================================================================
|   31.1% | 1.820974 |        -- |      -- | 386420.0 |MPI
||----------------------------------------------------------------------
||   29.3% | 1.716654 | 1.905820 |   57.4% | 128200.0 |mpi_recv
||    1.1% | 0.063398 | 0.017869 |   24.0% | 128600.0 |MPI_ISEND
||======================================================================
|   29.6% | 1.733093 |        -- |      -- |    202.0 |MPI_SYNC
||----------------------------------------------------------------------
|   29.6% | 1.732687 | 1.728773 |   99.8% |    200.0 | mpi_allreduce_(sync)
|=======================================================================
```

Detecting load imbalance



- **Motivation**
  - Increase of systems complexity
  - Increase applications scaling
- **Imbalance time**
  - Metric based on execution time
  - User functions : $t_{\text{Maximum}} - t_{\text{Average}}$
  - MPI sync time : $t_{\text{Average}} - t_{\text{Minimum}}$

- Identifies computational code regions and synchronization calls that could benefit most from load balance optimization
- Estimates how much overall program time could be saved if corresponding section of code had a perfect balance
- Represents upper bound on “potential savings”
- Assumes other processes are waiting, not doing useful work while slowest member finishes
- Minimize computing resources 'waste'

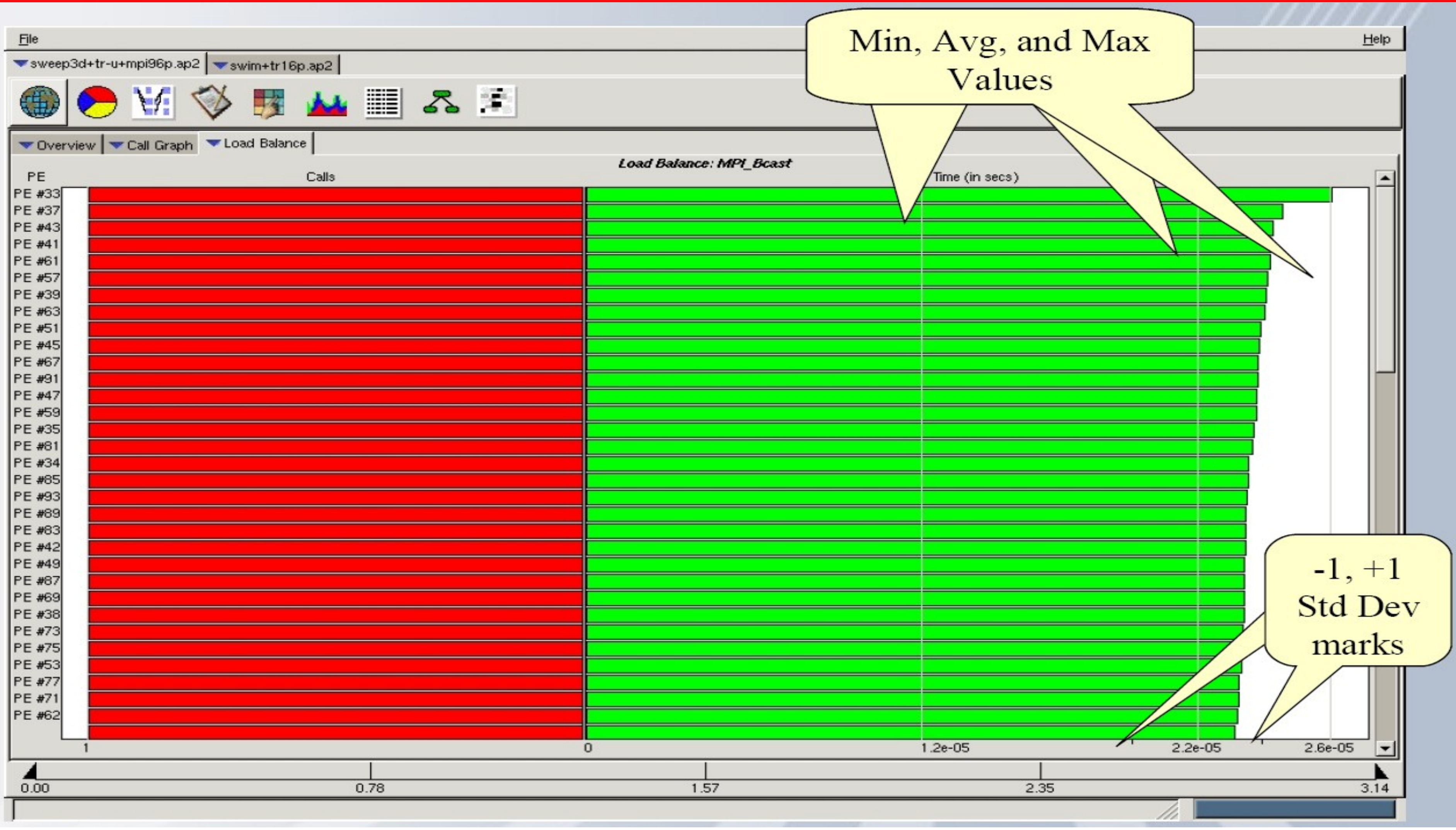
```
gele1:/scratch/gele/piccinal/laplace $  pat_report -O load_balance_m exe+apa+14148-16t.xf
pat_report:  Using existing file:   exe+apa+14148-16t.ap2
Processing table 1 of 1
Table 1:  Load Balance with MPI Message Stats

 Time%   |    Time   | MPI Msg   |  MPI Msg   |    Avg   |Group
         |           |   Count   |    Bytes   |    MPI   | PE=[mmm]
         |           |           |            |    Msg   |
         |           |           |            |   Size   |

 100.0% | 6.171748 |  96618.7 | 2561620.0 |   26.51 | Total
|------------------------------------------------------------
|  37.4% | 2.309054 |      0.0 |       0.0 |      -- |USER
||-----------------------------------------------------------
||  39.4% | 2.432767 |      0.0 |       0.0 |      -- |pe.10
||  37.4% | 2.310209 |      0.0 |       0.0 |      -- |pe.6
||  33.1% | 2.039806 |      0.0 |       0.0 |      -- |pe.5
||===========================================================
|  34.5% | 2.129440 |  96618.7 | 2561620.0 |   26.51 |MPI
||-----------------------------------------------------------
||  65.7% | 4.055387 | 128602.0 | 2817620.0 |   21.91 |pe.5
||  34.1% | 2.104354 |    802.0 | 2561620.0 | 3194.04 |pe.10
||   6.4% | 0.392414 | 128402.0 | 1793620.0 |   13.97 |pe.0
||===========================================================
|  28.1% | 1.733254 |      0.0 |       0.0 |      -- |MPI_SYNC
||-----------------------------------------------------------
||  56.3% | 3.477124 |      0.0 |       0.0 |      -- |pe.0
||  26.7% | 1.649781 |      0.0 |       0.0 |      -- |pe.7
||   0.1% | 0.004096 |      0.0 |       0.0 |      -- |pe.11
|============================================================
```

max, average, min

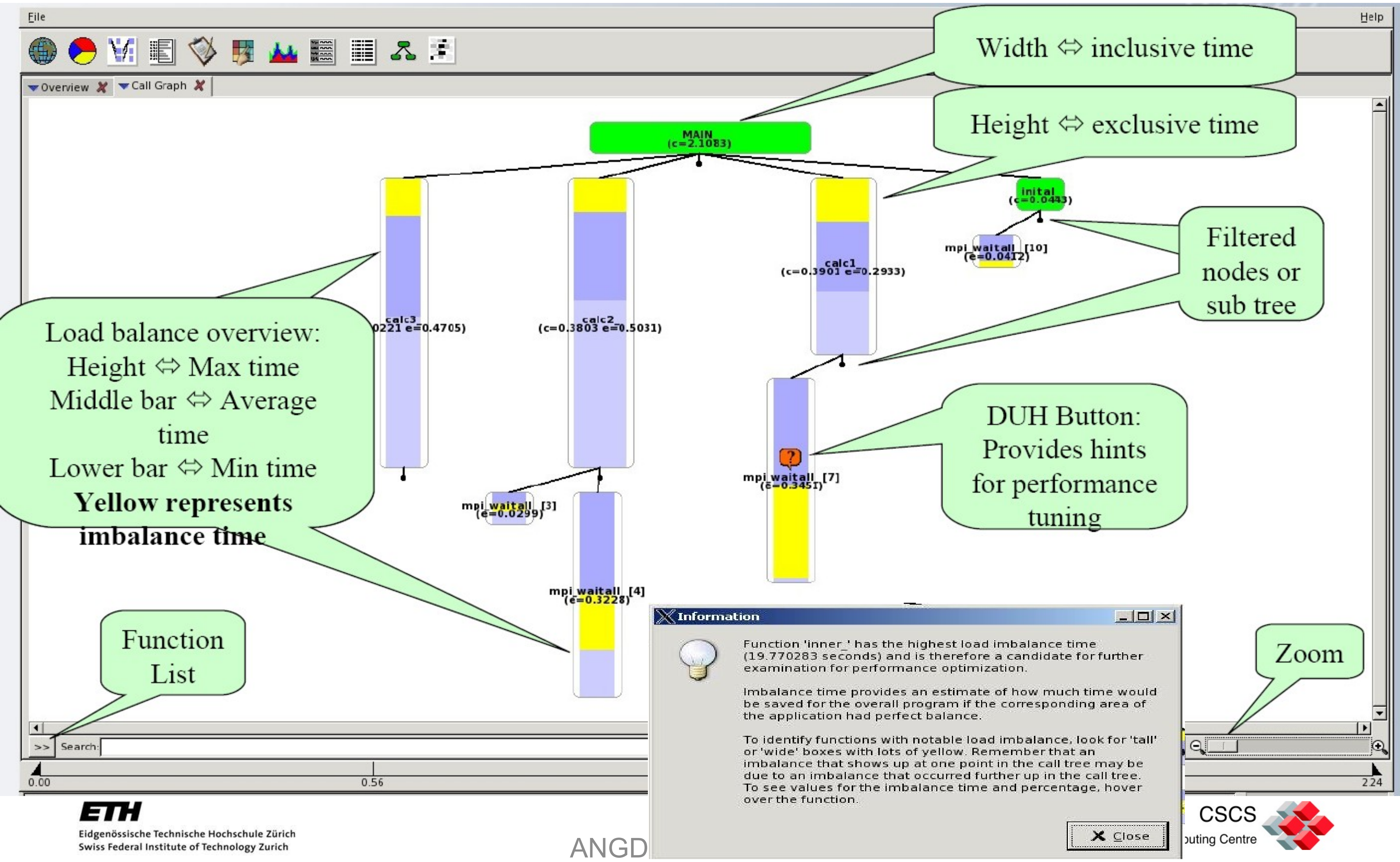
max, average, min

max, average, min

Apprentice2 : Load imbalance view



Apprentice2 : Call tree view



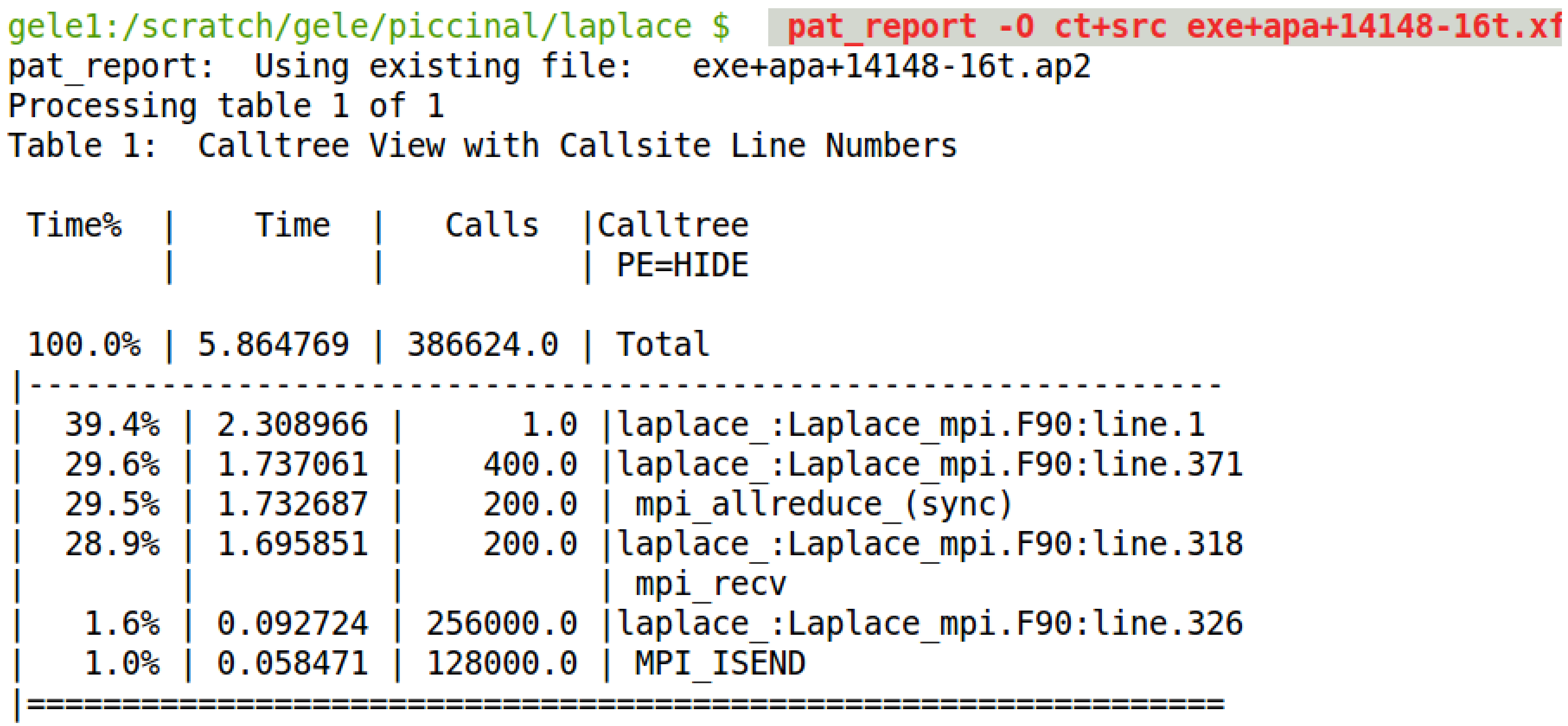
File
Help
Overview
Call Graph
MAIN (c=2.1083)
calc3 0221 e=0.4705)
calc2 (c=0.3803 e=0.5031)
calc1 (c=0.3901 e=0.2933)
inital (c=0.0443)
mpi_waitall [10] (e=0.0412)
mpi_waitall [3] (e=0.0299)
mpi_waitall [4] (e=0.3228)
mpi_waitall [7] (e=0.3451)
Width ⇔ inclusive time
Height ⇔ exclusive time
Filtered nodes or sub tree
Load balance overview: Height ⇔ Max time Middle bar ⇔ Average time Lower bar ⇔ Min time **Yellow represents imbalance time**
DUH Button: Provides hints for performance tuning
Function List
Zoom
Search:
0.00
0.56
224
Information
Close

Function 'inner_' has the highest load imbalance time (19.770283 seconds) and is therefore a candidate for further examination for performance optimization.

Imbalance time provides an estimate of how much time would be saved for the overall program if the corresponding area of the application had perfect balance.

To identify functions with notable load imbalance, look for 'tall' or 'wide' boxes with lots of yellow. Remember that an imbalance that shows up at one point in the call tree may be due to an imbalance that occurred further up in the call tree. To see values for the imbalance time and percentage, hover over the function.

Step2 : pat_report (src line numbers)



```
gele1:/scratch/gele/piccinal/laplace $  pat_report -O ct+src exe+apa+14148-16t.xf
pat_report:  Using existing file:   exe+apa+14148-16t.ap2
Processing table 1 of 1
Table 1:  Calltree View with Callsite Line Numbers

 Time%   |    Time   |   Calls   |Calltree
         |           |           | PE=HIDE

 100.0% | 5.864769 | 386624.0 | Total
|---------------------------------------------------------------
|  39.4% | 2.308966 |       1.0 |laplace_:Laplace_mpi.F90:line.1
|  29.6% | 1.737061 |     400.0 |laplace_:Laplace_mpi.F90:line.371
|  29.5% | 1.732687 |     200.0 | mpi_allreduce_(sync)
|  28.9% | 1.695851 |     200.0 |laplace_:Laplace_mpi.F90:line.318
|        |          |           | mpi_recv
|   1.6% | 0.092724 | 256000.0 |laplace_:Laplace_mpi.F90:line.326
|   1.0% | 0.058471 | 128000.0 | MPI_ISEND
|===============================================================
gele1:/scratch/gele/piccinal/laplace $  pat_report -O mpi_callers exe+apa+14148-16t.xf
pat_report:  Using existing file:   exe+apa+14148-16t.ap2
Processing table 1 of 1
Table 1:  MPI Message Stats by Caller

  MPI Msg   | MPI Msg   |    MsgSz  | 256B<=   | 4KB<=   |Function
    Bytes   |   Count   |     <16B  |  MsgSz   | MsgSz   | Caller
            |           |    Count  |   <4KB   | <64KB   |  PE=[mmm]
            |           |           |  Count   | Count   |

 2561620.0 |  96618.7 |  96202.0 |   266.7 |  150.0 | Total
|------------------------------------------------------------------
| 2560000.0 |  96416.7 |  96000.0 |   266.7 |  150.0 |MPI_ISEND
| 2560000.0 |  96416.7 |  96000.0 |   266.7 |  150.0 | laplace_
|||----------------------------------------------------------------
3|| 3584000.0 | 128600.0 | 128000.0 |   400.0 |  200.0 |pe.4
3|| 2560000.0 | 128400.0 | 128000.0 |   400.0 |    0.0 |pe.1
3|| 1792000.0 |    400.0 |      0.0 |   200.0 |  200.0 |pe.11
|||================================================================
|    1600.0 |    200.0 |    200.0 |     0.0 |    0.0 |MPI_ALLREDUCE
|    1600.0 |    200.0 |    200.0 |     0.0 |    0.0 | laplace_
|||----------------------------------------------------------------
3||    1600.0 |    200.0 |    200.0 |     0.0 |    0.0 |pe.0
3||    1600.0 |    200.0 |    200.0 |     0.0 |    0.0 |pe.6
3||    1600.0 |    200.0 |    200.0 |     0.0 |    0.0 |pe.11
|||================================================================
```

Average of 12

```
gele1:/scratch/gele/piccinal/laplace $  pat_report -O mpi_dest_bytes exe+apa+14148-16t.x
pat_report:  Using existing file:   exe+apa+14148-16t.ap2
Processing table 1 of 1
Table 1:  MPI Sent Message Stats by Distance

   Sent   |    Sent   |   Sent   |    MsgSz  | 256B<=   | 4KB<=   |Sent Distance
    Msg   |     Msg   |    Msg   |     <16B  |  MsgSz   | MsgSz   |
  Total   |   Total   |  Count   |    Count  |   <4KB   | <64KB   |
 Bytes%   |   Bytes   |          |           |  Count   | Count   |

  100.0% | 30720000 | 1157000 | 1152000.0 |   3200.0 | 1800.0 | Total
|---------------------------------------------------------------------------
|   60.0% | 18432000 | 1153800 | 1152000.0 |      0.0 | 1800.0 |3
|   40.0% | 12288000 |    3200 |       0.0 |   3200.0 |    0.0 |1
|===========================================================================
```

Sum of 12

Step3 : Memory usage (INTEL)

```
make clean ; make FFLAGS="-O3 -w -D_HWMEM"
```

```
t=5.3(sec) 140(MB) o_exe.mem.0002.2.2.1.1920-1920-200-1.0d-5--bind-to-core
t=6.3(sec) 53(MB) o_exe.mem.0006.6.6.1.1920-1920-200-1.0d-5--bind-to-core
t=4.2(sec) 32(MB) o_exe.mem.0012.12.12.1.1920-1920-200-1.0d-5--bind-to-core
t=4.8(sec) 51(MB) o_exe.mem.0012.12.6.1.1920-1920-200-1.0d-5--bind-to-core
t=3.6(sec) 41(MB) o_exe.mem.0024.24.12.1.1920-1920-200-1.0d-5--bind-to-core
```

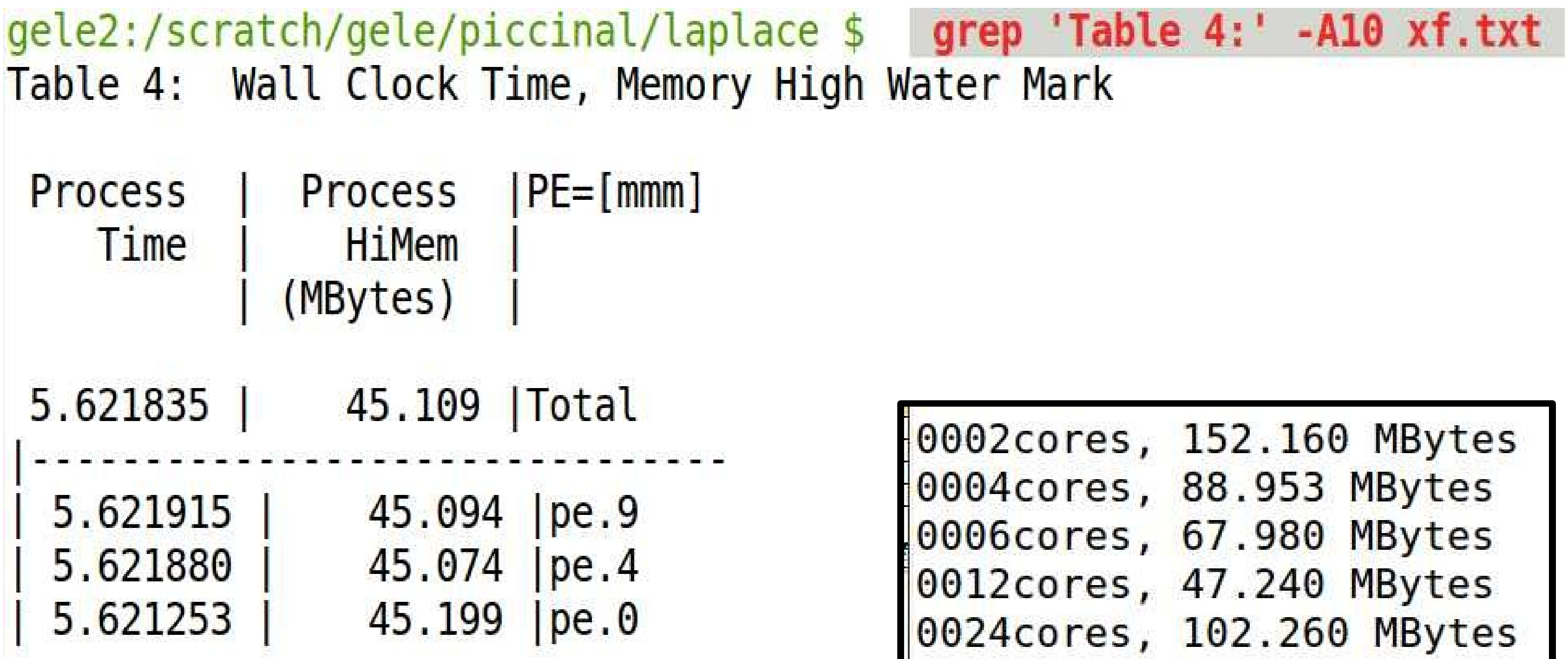
Domain decomposition =>
Less memory / process

```
o_exe.mem.0004.4.1.1.1920-1920-200-1.0d-5--bind-to-core t=4.87sec m=91MB
o_exe.mem.0004.4.2.1.1920-1920-200-1.0d-5--bind-to-core t=5.50sec m=90MB
o_exe.mem.0004.4.3.1.1920-1920-200-1.0d-5--bind-to-core t=7.30sec m=73MB
o_exe.mem.0004.4.4.1.1920-1920-200-1.0d-5--bind-to-core t=6.69sec m=73MB
```

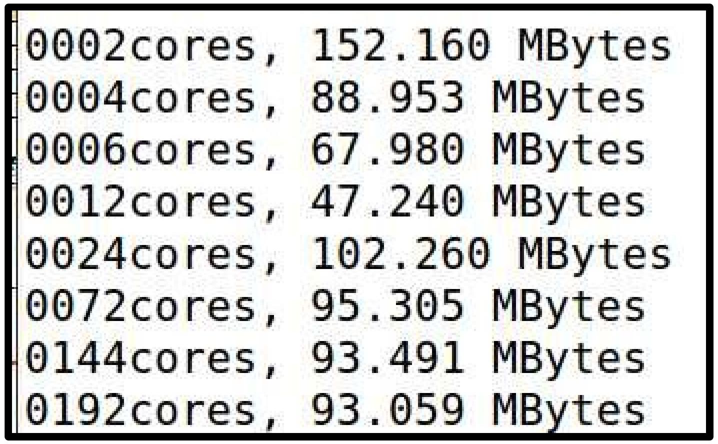
<=============== FAST

<== SLOW but only 1 cnode
and less memory / process

Step3 : Memory usage (CRAY)



```
gele2:/scratch/gele/piccinal/laplace $  grep 'Table 4:' -A10 xf.txt
Table 4:  Wall Clock Time, Memory High Water Mark

 Process   |  Process   |PE=[mmm]
    Time   |    HiMem   |
           | (MBytes)   |

 5.621835 |    45.109 |Total
|---------------------------------
| 5.621915 |    45.094 |pe.9
| 5.621880 |    45.074 |pe.4
| 5.621253 |    45.199 |pe.0
```



```
0002cores, 152.160 MBytes
0004cores, 88.953 MBytes
0006cores, 67.980 MBytes
0012cores, 47.240 MBytes
0024cores, 102.260 MBytes
0072cores, 95.305 MBytes
0144cores, 93.491 MBytes
0192cores, 93.059 MBytes
```

```
make clean ; make FFLAGS="-O3 -w -D_HWMEM"
```

How to print memory usage ?

Every Linux based system supports a /proc/self/status file :

- 1 file for each active process id (mpi rank)

VmPeak:	Peak virtual memory usage	
VmSize:	Current virtual memory usage	
VmLck:	Current mlocked memory	
VmHWM:	Peak resident set size	### high water mark ###
VmRSS:	Resident set size	### current memory usage ###
VmData:	Size of "data" segment	
VmStk:	Size of stack	
VmExe:	Size of "text" segment	
VmLib:	Shared library usage	
VmPTE:	Pagetable entries size	
VmSwap:	Swap space used	

PAPI Features and usage

PAPI characteristics

- University of Tennessee and vendors

  http://icl.cs.utk.edu/papi

- PAPI (Performance Application Programming Interface) is a portable interface to hardware performance counters
- Collecting low level performance metrics (e.g. clock cycles and instruction counts, memory cache misses, functional units, etc)

PAPI features

- 2 types of performance events :
- Platform dependent native events
- Platform independent preset events
- Preset events are derived from multiple native events
- PAPI support on most HPC platforms
  - IBM, CRAY, INTEL, etc...

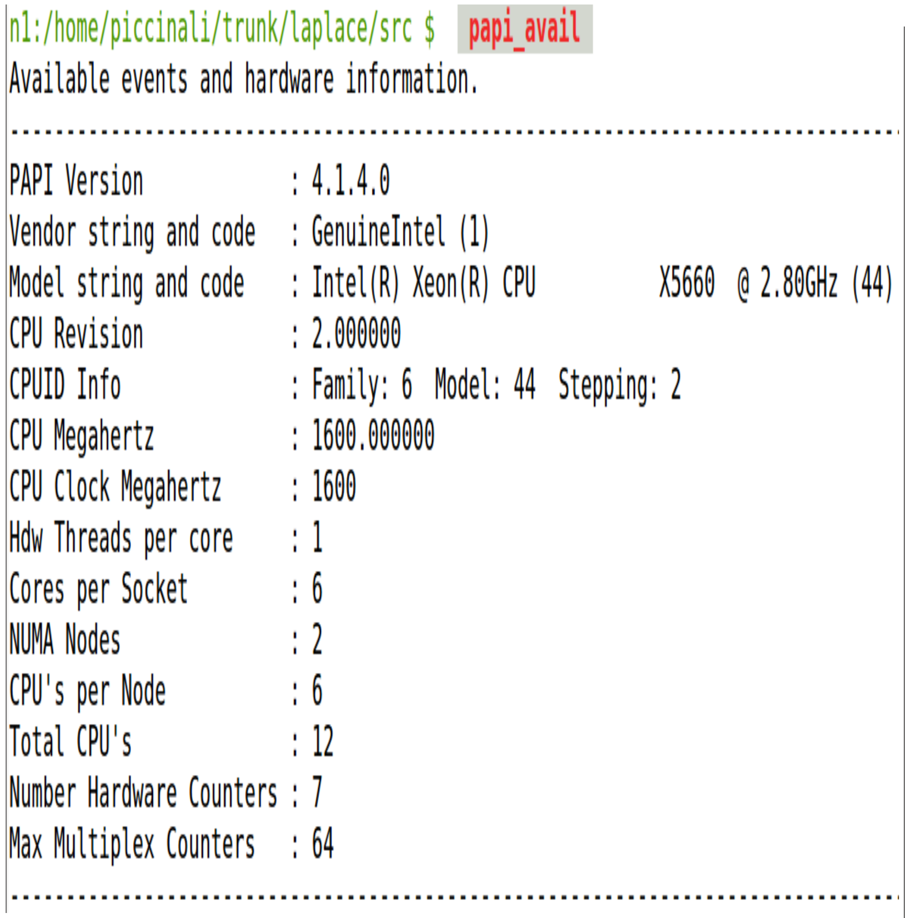
Information on PAPI and AMD hardware counters

- Load the xt-papi (and xt-craypat) modulefile(s)
  - module load xt-papi ; module load xt-craypat
  - man intro_papi ; man papi_counters ; man hwpc
  - pat_help counters

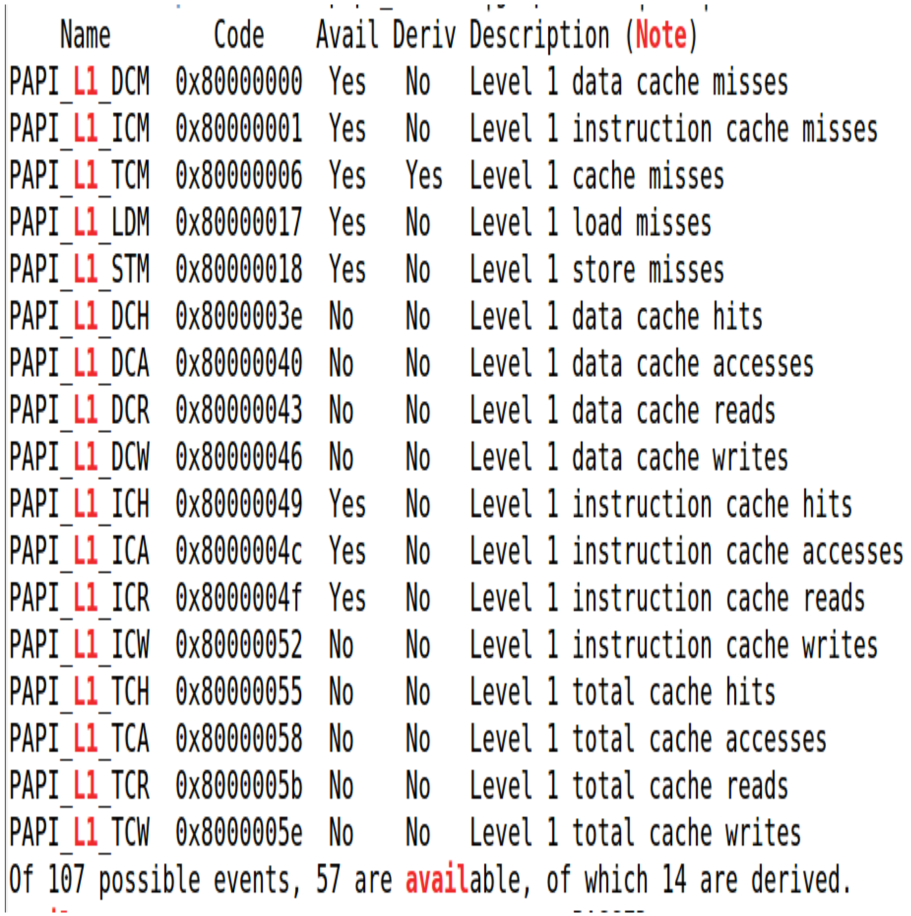
PAPI Utilities (to be executed on compute node)

- papi_avail shows which predefined events are available on the system
- papi_native_avail lists all AMD native events available on the system
- derived metrics are made from native counter event names
- papi_event_chooser reports information about the current PAPI installation and supported preset events
- papi_command_line adds named events from the command line to a PAPI EventSet and does some work with that EventSet. This serves as a handy way to see if events can be counted together, and if they give reasonable results for known work.
- papi_mem_info provides information on the memory architecture of the processor

Better know your machine : papi_avail



```
n1:/home/piccinali/trunk/laplace/src $  papi_avail
Available events and hardware information.
--------------------------------------------------------------------------------
PAPI Version             : 4.1.4.0
Vendor string and code   : GenuineIntel (1)
Model string and code    : Intel(R) Xeon(R) CPU         X5660  @ 2.80GHz (44)
CPU Revision             : 2.000000
CPUID Info               : Family: 6  Model: 44  Stepping: 2
CPU Megahertz            : 1600.000000
CPU Clock Megahertz      : 1600
Hdw Threads per core     : 1
Cores per Socket         : 6
NUMA Nodes               : 2
CPU's per Node           : 6
Total CPU's              : 12
Number Hardware Counters : 7
Max Multiplex Counters   : 64
--------------------------------------------------------------------------------
```



```
    Name        Code    Avail Deriv Description (Note)
PAPI_L1_DCM  0x80000000  Yes   No   Level 1 data cache misses
PAPI_L1_ICM  0x80000001  Yes   No   Level 1 instruction cache misses
PAPI_L1_TCM  0x80000006  Yes   Yes  Level 1 cache misses
PAPI_L1_LDM  0x80000017  Yes   No   Level 1 load misses
PAPI_L1_STM  0x80000018  Yes   No   Level 1 store misses
PAPI_L1_DCH  0x8000003e  No    No   Level 1 data cache hits
PAPI_L1_DCA  0x80000040  No    No   Level 1 data cache accesses
PAPI_L1_DCR  0x80000043  No    No   Level 1 data cache reads
PAPI_L1_DCW  0x80000046  No    No   Level 1 data cache writes
PAPI_L1_ICH  0x80000049  Yes   No   Level 1 instruction cache hits
PAPI_L1_ICA  0x8000004c  Yes   No   Level 1 instruction cache accesses
PAPI_L1_ICR  0x8000004f  Yes   No   Level 1 instruction cache reads
PAPI_L1_ICW  0x80000052  No    No   Level 1 instruction cache writes
PAPI_L1_TCH  0x80000055  No    No   Level 1 total cache hits
PAPI_L1_TCA  0x80000058  No    No   Level 1 total cache accesses
PAPI_L1_TCR  0x8000005b  No    No   Level 1 total cache reads
PAPI_L1_TCW  0x8000005e  No    No   Level 1 total cache writes
Of 107 possible events, 57 are available, of which 14 are derived.
```

How to use PAPI API ?

```
#include fpapi.h
integer :: num_events, event(2), values(2)
call PAPIf_num_counters( num_events )
```

- Initialise PAPI, also shows how many hardware events are supported (num_events),

```
call PAPIf_query_event(PAPI_FP_INS, retval)
event(1) = PAPI_FP_INS    # Total floating point operations,
event(2) = PAPI_TOT_CYC   # Time used
```

- check if the PAPI Preset event can be counted on the architecture,
- If the event CAN be counted, the function returns PAPI_OK, if the event CANNOT be counted, the function returns an error code,

```
call PAPIf_start_counters( event, num_events, retval)
call PAPIf_read_counters(values, num_events,retval)
```

- start counting the events named in the "event" array,
- and reset the counters before entering the region of code to be measured,
- size of the event array should be no longer than the value returned by PAPIf_num_counters,

```
call PAPIf_stop_counters(values,num_events,retval)
```

- stop the counters and copy the counts into the values array after the src code region to be measured.

TP : papi

Objectif : Calcul de la performance crete

Instructions :

* Calculez le nombre de Flops theorique (rpeak) pour 1, 2, 12, 24, 36, 48 cores.

Indices :
  * Ghz = Nombre de Cycles / 1 seconde
  * Flop / Cycle = 24 (intel xeon X5660)
  * Flops = floating point operations / 1 seconde

* Compilez et executez ~piccinali/trunk/matmul/seq/papi/matmult.F90 :
  * `gfortran -O3 -I/softs/papi/4.1.4/gnu/include -D_A matmul.F90 -L/softs/papi/4.1.4/gnu/lib -lpapi -o A ; ./A`
  * `gfortran -O3 -I/softs/papi/4.1.4/gnu/include -D_D matmul.F90 -L/softs/papi/4.1.4/gnu/lib -lpapi -o D ; ./D`
  * Quel est le nombre de flop ? Quel est la performance (Flops) par rapport a la crete ?

* Changez la taille de la matrice (PARAMETER ligne 12) : y a-t-il une difference ?

TP : papi (solution)

flops = [cycles / sec] * [flop / cycle]
1 core => 2.8 10^9 * 24 * 1 = 67.3 GFLOPS
2 cores => 2.8 10^9 * 24 * 2 = 134 GFLOPS
12 cores => 2.8 10^9 * 24 * 12 = 806 GFLOPS
24 cores => 2.8 10^9 * 24 * 24 = 1.6 TFLOPS
36 cores => 2.8 10^9 * 24 * 36 = 2.4 GFLOPS
48 cores => 2.8 10^9 * 24 * 48 = 3.225 TFLOPS

jki (D)
FP Instructions: 67110331
Cycles : 107688026
=> D = 1.75 GFLOPS
2.6% peak (2.8*67110331/107688026)/67.3*100)

ijk (A)
FP Instructions: 279047242
Cycles : 898272636
=> A = 0.86 GFLOPS
1.3% peak (2.8*279047242/898272636)/67.3*100)

Step5 : I/O

```
n1:~/trunk/laplace/src/io
 make clean
 Comparez les temps d'exe de :
 * make FFLAGS="-D_WITHO -O3 -w"
 * make FFLAGS="-O3 -w"

 /usr/bin/time -p /softs/openmpi-1.4.3/bin/mpiexec
-machinefile /softs/openmpi/h -n 4 ./exe 1920 1920 25 1.0d-5
Recommencez en variant le nombre de taches mpi...
```

```
  CORES / TOTALTIME / CPU / IO
  2cores t=13 : 10.7 +  2.8 seconds
  4cores t=11 :  9.9 +  1.8 seconds
  6cores t=10 :  8.7 +  1.5 seconds
 12cores t= 6 :  5.5 +  1.4 seconds
 24cores t= 7 :  5.0 +  3.0 seconds
 72cores t=12 :  3.7 +  8.7 seconds
144cores t=19 :  4.2 + 15.7 seconds
192cores t=21 :  4.0 + 17.3 seconds
```

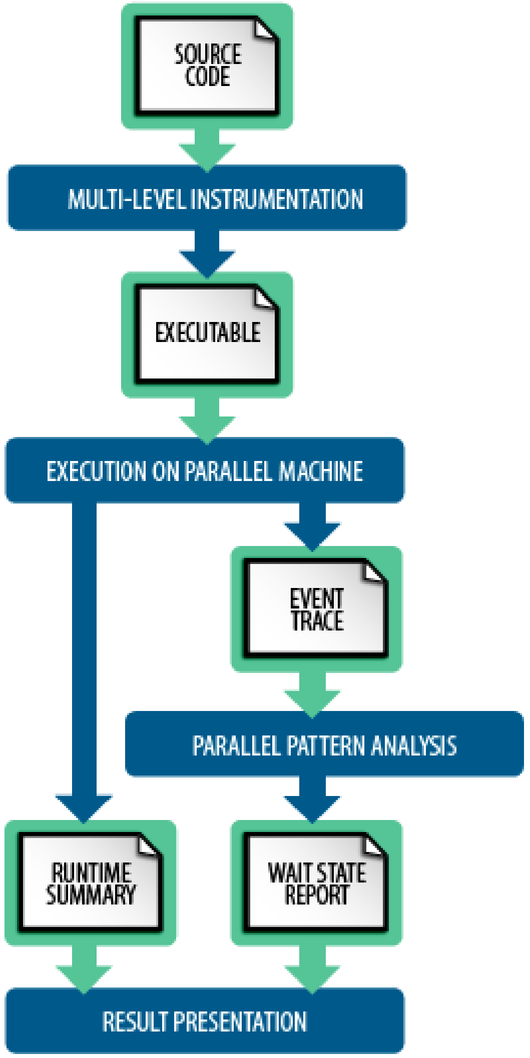
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

CSCS
Swiss National Supercomputing Centre

```
  The Total value for each data item is the sum of the File Name values.
  The File Name value for Write B/Call is the avg of the PE values.
  The File Name value for each of Write MB, Writes, Write Time, Write Rate
    MB/sec is the sum of the PE values.
    (To specify different aggregations, see:  pat_help report options s1)

  This table shows only lines with Writes > 0.

Table 6:  File Output Stats by Filename

    Write |  Write MB |Write Rate |        Writes |  Write |File Name
     Time |           |    MB/sec |               | B/Call | PE='HIDE'

 0.515503 | 43.457216 | 84.300636 | 2467865.000000 |  18.46 |Total
|------------------------------------------------------------------
| 0.044181 |  3.613281 | 81.783283 |  204800.000000 |   18.50 |Potentiel2D_008.dat
| 0.043875 |  3.613281 | 82.354246 |  204800.000000 |   18.50 |Potentiel2D_006.dat
| 0.043224 |  3.613281 | 83.593739 |  204800.000000 |   18.50 |Potentiel2D_001.dat
| 0.043046 |  3.613281 | 83.939147 |  204800.000000 |   18.50 |Potentiel2D_011.dat
| 0.042989 |  3.613281 | 84.051665 |  204800.000000 |   18.50 |Potentiel2D_010.dat
| 0.042737 |  3.613281 | 84.547593 |  204800.000000 |   18.50 |Potentiel2D_000.dat
| 0.042590 |  3.613281 | 84.838995 |  204800.000000 |   18.50 |Potentiel2D_007.dat
| 0.042543 |  3.613281 | 84.932644 |  204800.000000 |   18.50 |Potentiel2D_005.dat
| 0.042341 |  3.613281 | 85.337258 |  204800.000000 |   18.50 |Potentiel2D_004.dat
| 0.042326 |  3.613281 | 85.368827 |  204800.000000 |   18.50 |Potentiel2D_009.dat
| 0.042148 |  3.613281 | 85.729068 |  204800.000000 |   18.50 |Potentiel2D_003.dat
| 0.041697 |  3.613281 | 86.656569 |  204800.000000 |   18.50 |Potentiel2D_002.dat
| 0.000423 |  0.024413 | 57.667058 |    2559.000000 |   10.00 |Potentiel1D_004.dat
| 0.000423 |  0.024413 | 57.669839 |    2559.000000 |   10.00 |Potentiel1D_007.dat
| 0.000422 |  0.024413 | 57.903052 |    2559.000000 |   10.00 |Potentiel1D_001.dat
| 0.000417 |  0.024413 | 58.489748 |    2559.000000 |   10.00 |Potentiel1D_010.dat
| 0.000121 |  0.000189 |  1.558470 |      29.000000 |    6.83 |stdout
|==================================================================
```

Scalasca : Features

- Scalasca is an open-source toolset that can be used to analyze the performance behavior of parallel applications :
  - Developed by the Jülich Supercomputing Centre, Germany
  - Designed for use on large-scale systems (Cray XT, IBM) and also for medium scale,
  - Supports message passing and threads (MPI, OpenMP, SHMEM, CAF)
  - Provides a low-overhead performance summary of CPU, parallel and memory usage.

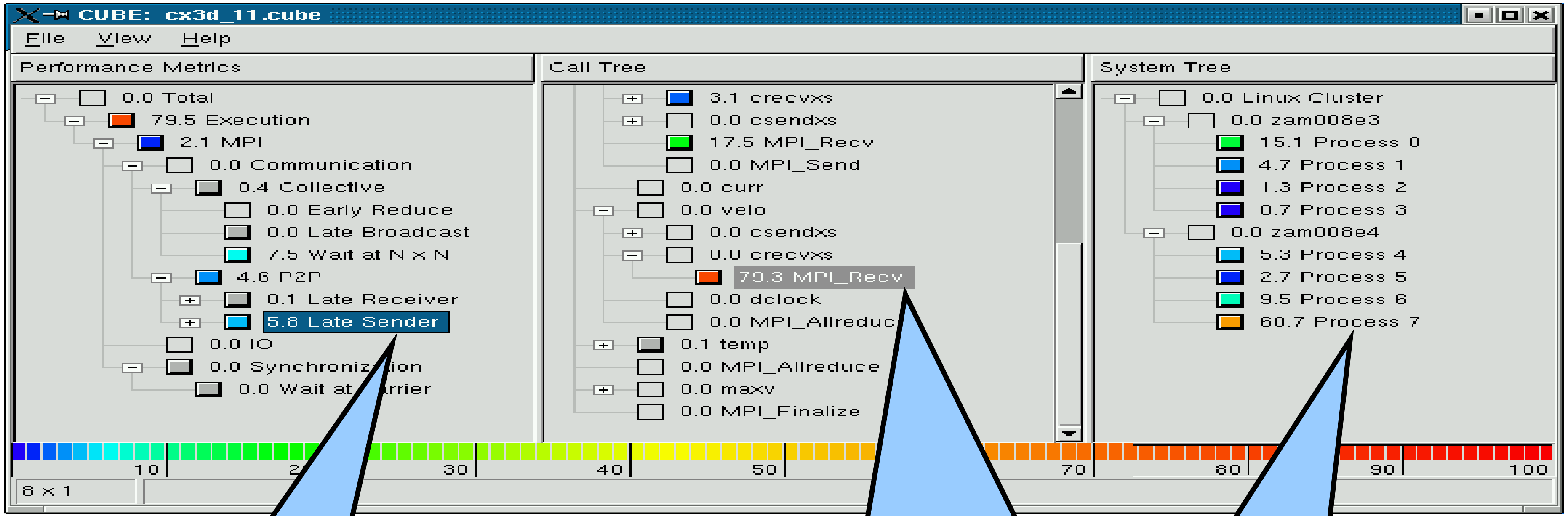


Using SCALASCA

- Profiling your code with SCALASCA involves the following steps :
  - Load the SCALASCA module
    - ➔ module load *scalasca*
    - – Using the CRAY wrappers (ftn, cc, CC) will automatically link your MPI code with the ipm library : no need to modify your compilation line, the wrapper will do it !
  - Instrument your code (cannot handle compiling and linking in one step)
    - ➔ scalasca -instrument *mpif90 -c test.f90*
    - – scalasca -instrument *mpif90 -o test.exe test.o*
  - Execute the resultant executable by submitting your PBS job
    - ➔ scalasca -analyse *mpirun -np 128 test.exe*
  - Visualize the results
    - ➔ scalasca -examine epik_a
    - ➔ cube3 epik_a/epitome.cube (java GUI)
      - can be done on your workstation !

Using CUBE

- CUBE is the tool to interactively examine the parallel application execution analysis reports
  - Easy to use and portable
  - Uses a relatively simple XML input file structure, simple operations can be performed om multiple inputs (diff, merge)
  - Displays tree-based views of collected infos (calls, walltime, etc...),
  - Is compatible with IPM/TAU log files



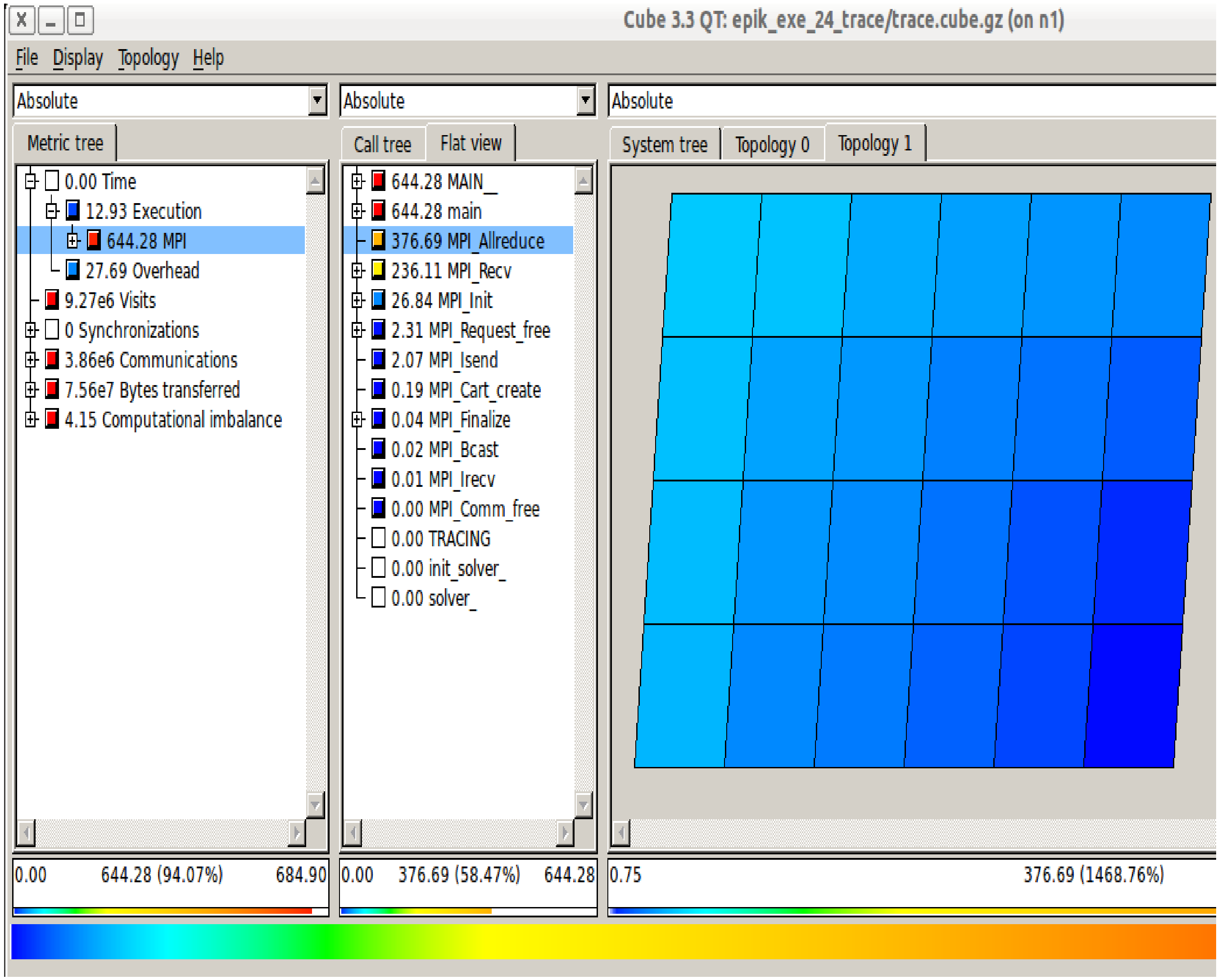
```
Performance Metrics            Call Tree                        System Tree
0.0 Total                      3.1 crecvxs                      0.0 Linux Cluster
 79.5 Execution                0.0 csendxs                       0.0 zam008e3
  2.1 MPI                      17.5 MPI_Recv                      15.1 Process 0
   0.0 Communication           0.0 MPI_Send                       4.7 Process 1
    0.4 Collective             0.0 curr                           1.3 Process 2
     0.0 Early Reduce          0.0 velo                           0.7 Process 3
     0.0 Late Broadcast        0.0 csendxs                       0.0 zam008e4
     7.5 Wait at N x N         0.0 crecvxs                        5.3 Process 4
    4.6 P2P                    79.3 MPI_Recv                      2.7 Process 5
     0.1 Late Receiver         0.0 dclock                         9.5 Process 6
     5.8 Late Sender           0.0 MPI_Allreduc                   60.7 Process 7
   0.0 IO                      0.1 temp
   0.0 Synchroniz ion          0.0 MPI_Allreduce
    0.0 Wait at   rrier        0.0 maxv
                               0.0 MPI_Finalize
```

Which type of problem?

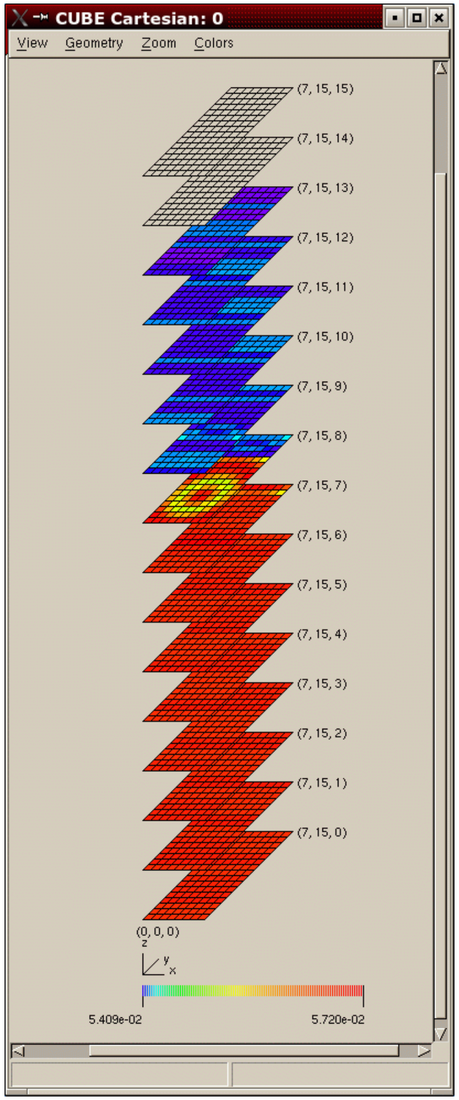
Where in the source code? Which call path?

Which process / thread ?

CUBE example output : Laplace (scalasca)



```
Metric tree                      Call tree
0.00 Time                        644.28 MAIN_
  12.93 Execution                644.28 main
    644.28 MPI                   376.69 MPI_Allreduce
    27.69 Overhead               236.11 MPI_Recv
9.27e6 Visits                    26.84 MPI_Init
0 Synchronizations               2.31 MPI_Request_free
3.86e6 Communications            2.07 MPI_Isend
7.56e7 Bytes transferred         0.19 MPI_Cart_create
4.15 Computational imbalance     0.04 MPI_Finalize
                                 0.02 MPI_Bcast
                                 0.01 MPI_Irecv
                                 0.00 MPI_Comm_free
                                 0.00 TRACING
                                 0.00 init_solver_
                                 0.00 solver_

0.00   644.28 (94.07%)   684.90
0.00   376.69 (58.47%)   644.28
0.75   376.69 (1468.76%)
```



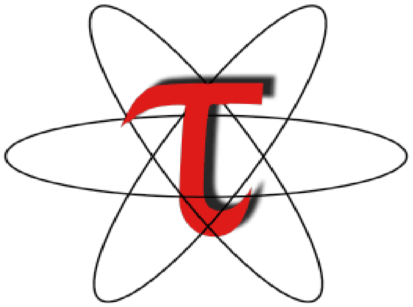
- <span style="color:red">Topology display :</span>
- Shows distribution of pattern over HW topology

References

- SCALASCA website (M. Geimer, B. Wylie, F. Wolf)
  - http://www.scalasca.org
- CSCS website
  - **User Entry Point >**
    - **Software and Programming Environment >**
      - **Debugging and Performance Analysis > Performance > Scalasca**

TAU Features and usage

- TAU characteristics
  - ➔ Flat MPI and callgraph profiling
  - ➔ Hardware counter data collection
  - ➔ OpenMP & pthread profiling
  - ➔ MPI tracing
  - ➔ Memory profiling
- TAU features
  - ➔ Auto-instrumentation utility (PDT)
  - ➔ Custom configurations
  - ➔ Interoperability with other tools



Profiling your code with TAU involves the following steps

- Load the TAU modulefile
  - ➔ `module load tau ; module help tau`
- Recompile your code using the TAU compiler scripts
  - ➔ `tau_f90.sh -c file.f90`
  - ➔ `tau_f90.sh -o exe file.o`
  - ➔ A simple relink will skip the autoinstrumentation step
- Execute the resultant executable by submitting your PBS job
  - ➔ `mpirun -n 12 exe`
- Visualize the results
  - ➔ `pprof profile.*`
  - ➔ `paraprof &`
  - ➔ can be done on your workstation !

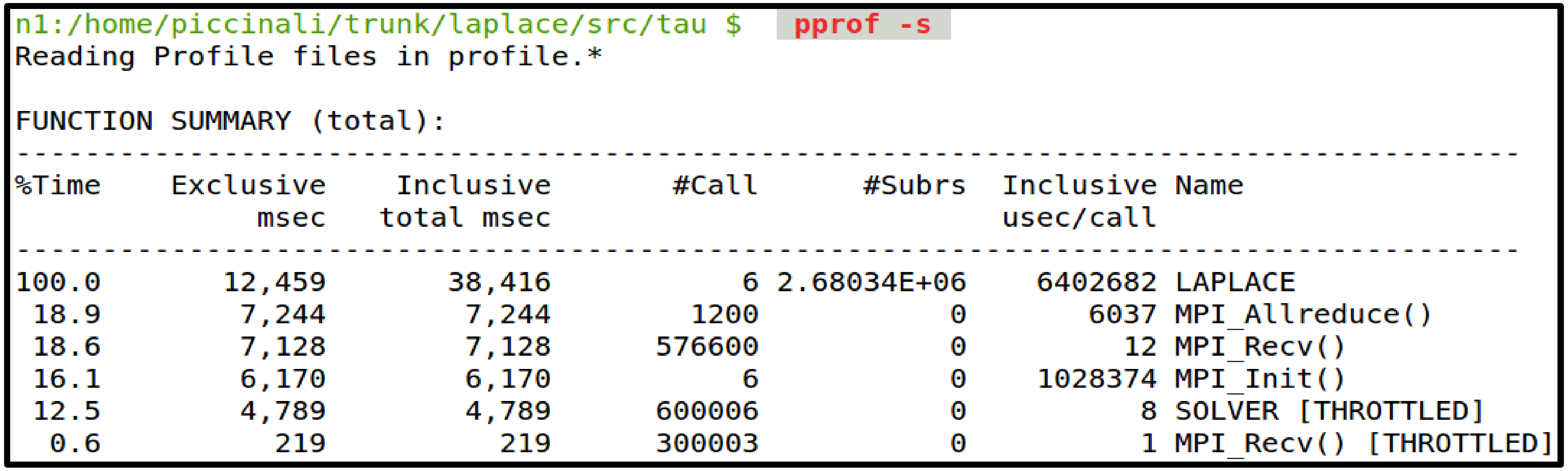
TAU

```
n1:/home/piccinali/trunk/laplace/src $  make clean ; make FCOMPILER=tau_f90.sh FFLAGS="-O3 -w" PROGRAM=exe.tau
```

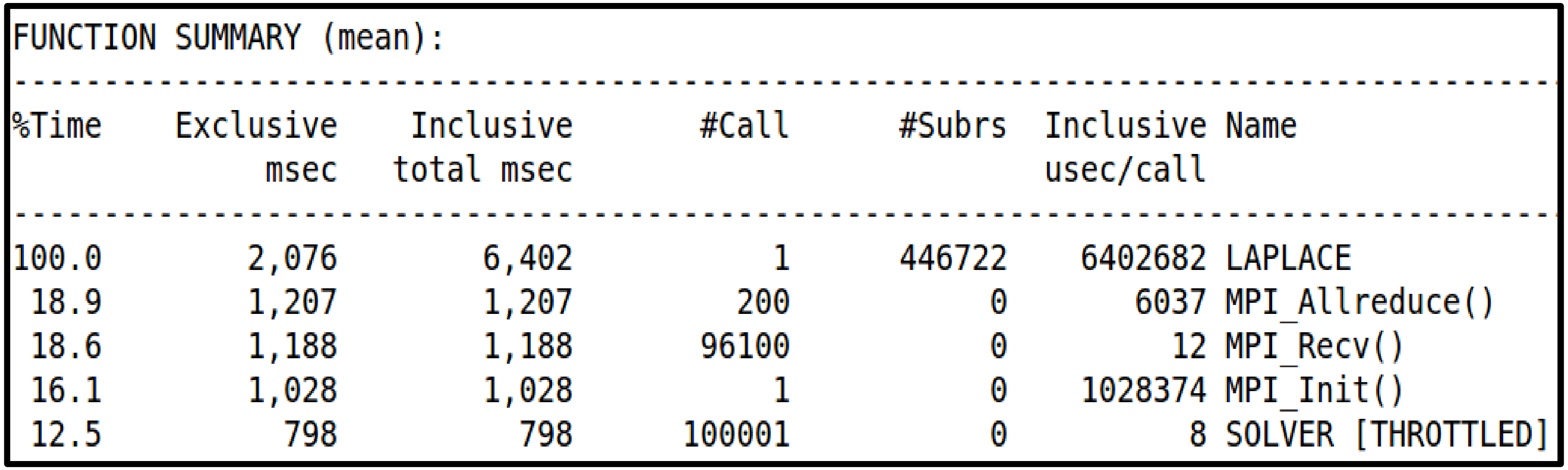
```
module help tau

----------- Module Specific Help for 'tau/2.20.3' -----------------


                modulefile : tau/2.20.3 help
                tau Version 2.20.3 :
                CSCS Users Documentation : http://www.cscs.ch
                http://www.cs.uoregon.edu/research/tau

                This version was compiled with gnu
                See : /softs/tau/2.20.3/gnu

                Usage :
                  module load tau/2.20.3  will set $TAU_MAKEFILE to Makefile.tau-papi-mpi-pdt,
                 currently  $TAU_MAKEFILE = Makefile.tau-papi-mpi-pdt
                 You can also choose to set TAU_MAKEFILE to other Makefile configurations
                 available in /softs/tau/2.20.3/gnu/x86_64/lib/ :
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-mpi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-phase-papi-mpi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-param-mpi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-depthlimit-mpi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-papi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-papi-mpi-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-papi-pthread-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-pthread-pdt
                 export TAU_MAKEFILE=/softs/tau/2.20.3/gnu/x86_64/lib/Makefile.tau-pdt

                 Use tau_cc.sh or tau_cxx.sh or  tau_f90.sh to compile your code,
                 Run your batch job as usual,
                 Use  pprof profile.0.0.0 OR  paraprof (= GUI) to visualize your results.
```

TAU

```
sbatch.sh ../exe.tau 6 6 1 "1920 1920 200 1.0d-5" "" "-bind-to-core"
```

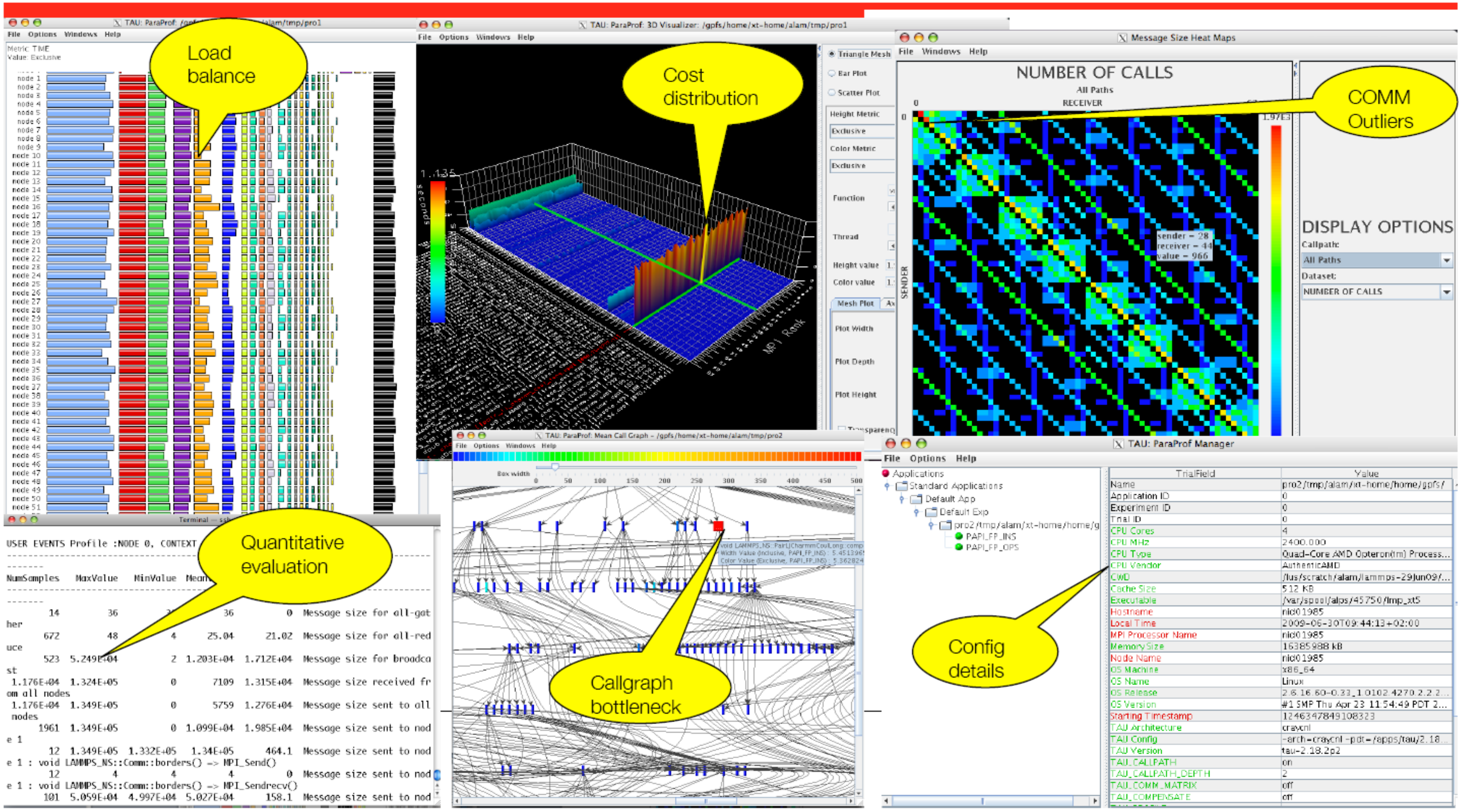


```
n1:/home/piccinali/trunk/laplace/src/tau $  pprof -s
Reading Profile files in profile.*

FUNCTION SUMMARY (total):
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0       12,459       38,416           6 2.68034E+06    6402682 LAPLACE
 18.9        7,244        7,244        1200           0       6037 MPI_Allreduce()
 18.6        7,128        7,128      576600           0         12 MPI_Recv()
 16.1        6,170        6,170           6           0    1028374 MPI_Init()
 12.5        4,789        4,789      600006           0          8 SOLVER [THROTTLED]
  0.6          219          219      300003           0          1 MPI_Recv() [THROTTLED]
```



```
FUNCTION SUMMARY (mean):
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0        2,076        6,402           1      446722    6402682 LAPLACE
 18.9        1,207        1,207         200           0       6037 MPI_Allreduce()
 18.6        1,188        1,188       96100           0         12 MPI_Recv()
 16.1        1,028        1,028           1           0    1028374 MPI_Init()
 12.5          798          798      100001           0          8 SOLVER [THROTTLED]
```

Outputs (TAU)



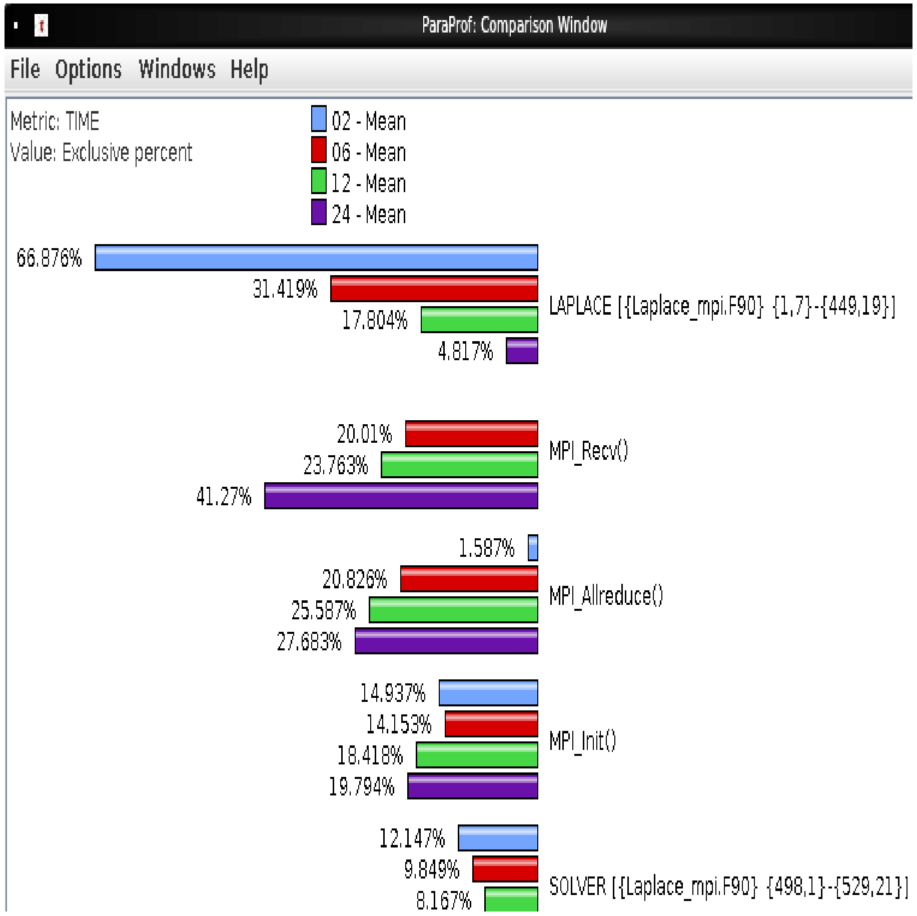
```
USER EVENTS Profile :NODE 0, CONTEXT
---------------------------------------------------------------------------
-------
NumSamples   MaxValue   MinValue  Mean
---------------------------------------------------------------------------
-------
        14         36                   36          0  Message size for all-gat
her
       672         48          4     25.04      21.02  Message size for all-red
uce
       523  5.249E+04          2 1.203E+04  1.712E+04  Message size for broadca
st
 1.176E+04  1.324E+05          0      7109  1.315E+04  Message size received fr
om all nodes
 1.176E+04  1.349E+05          0      5759  1.276E+04  Message size sent to all
 nodes
      1961  1.349E+05          0 1.099E+04  1.985E+04  Message size sent to nod
e 1
        12  1.349E+05  1.332E+05   1.34E+05      464.1  Message size sent to nod
e 1 : void LAMMPS_NS::Comm::borders() => MPI_Send()
        12          4          4          4          0  Message size sent to nod
e 1 : void LAMMPS_NS::Comm::borders() => MPI_Sendrecv()
       101  5.059E+04  4.997E+04  5.027E+04      158.1  Message size sent to nod
```

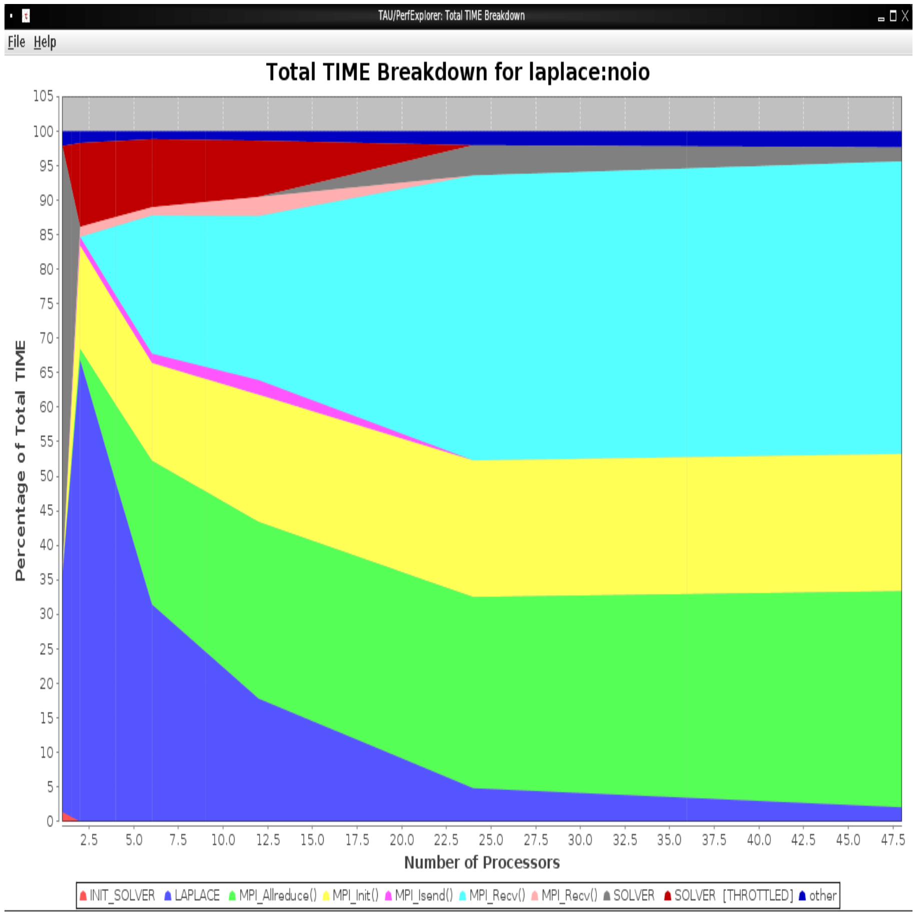
TrialField	Value
Name	pro2/tmp/alam/xt-home/home/gpfs/
Application ID	0
Experiment ID	0
Trial ID	0
CPU Cores	4
CPU MHz	2400.000
CPU Type	Quad-Core AMD Opteron(tm) Process...
CPU Vendor	AuthenticAMD
CWD	/lus/scratch/alam/lammps-29Jun09/...
Cache Size	512 KB
Executable	/var/spool/alps/45750/lmp_xt5
Hostname	nid01985
Local Time	2009-06-30T09:44:13+02:00
MPI Processor Name	nid01985
Memory Size	16385988 kB
Node Name	nid01985
OS Machine	x86_64
OS Name	Linux
OS Release	2.6.16.60-0.33_1.0102.4270.2.2.2...
OS Version	#1 SMP Thu Apr 23 11:54:49 PDT 2...
Starting Timestamp	1246347849108323
TAU Architecture	craycnl
TAU Config	-arch=craycnl -pdt=/apps/tau/2.18...
TAU Version	tau-2.18.2p2
TAU_CALLPATH	on
TAU_CALLPATH_DEPTH	2
TAU_COMM_MATRIX	off
TAU_COMPENSATE	off

Functions profile (tau)

What routines account for the most time ?

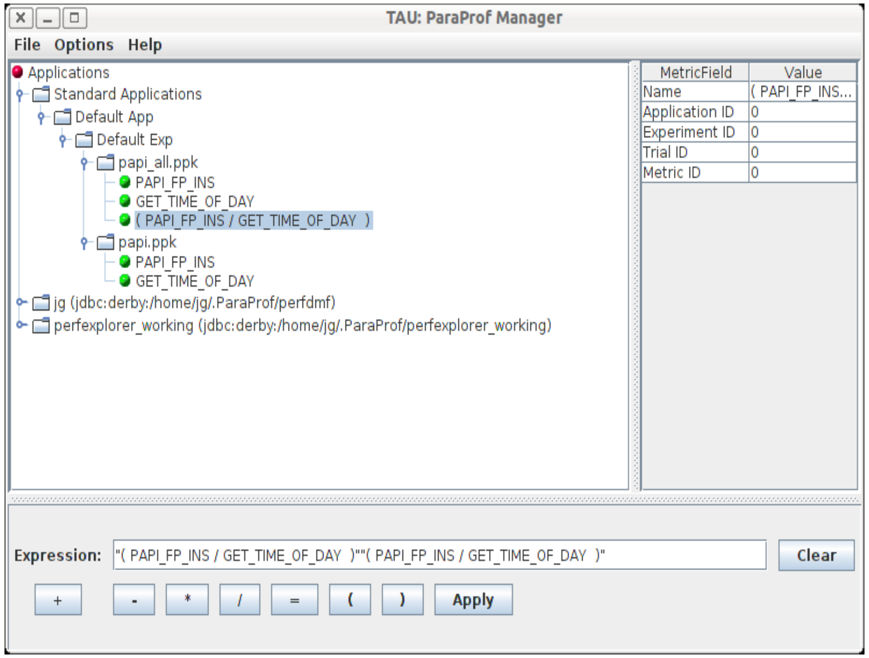
```
n1:/home/piccinali/trunk/laplace/src $  make clean ; make FCOMPILER=tau_f90.sh FFLAGS="-O3 -w"
```





PAPI (tau)

```
export TAU_OPTIONS=
export TAU_METRICS="GET_TIME_OF_DAY:PAPI_FP_INS"
paraprof --pack papi.ppk
paraprof papi.ppk
```

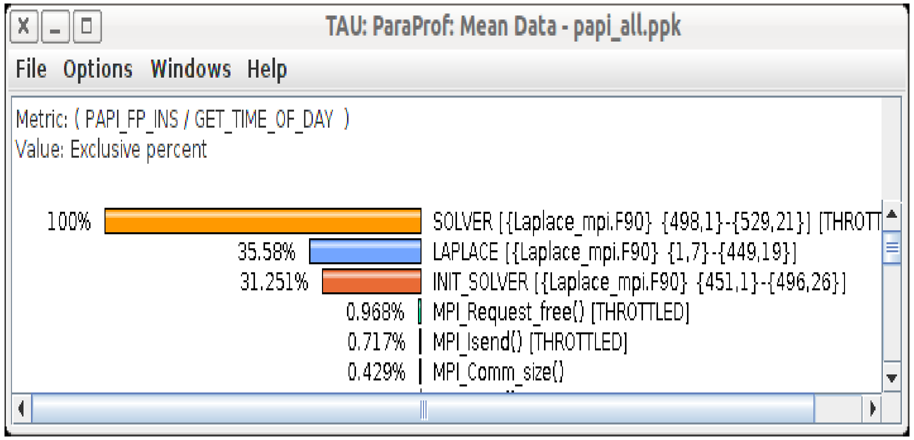

TAU: ParaProf Manager

File Options Help

- Applications
  - Standard Applications
    - Default App
      - Default Exp
        - papi_all.ppk
          - PAPI_FP_INS
          - GET_TIME_OF_DAY
          - (PAPI_FP_INS / GET_TIME_OF_DAY)
        - papi.ppk
          - PAPI_FP_INS
          - GET_TIME_OF_DAY
  - jg (jdbc:derby:/home/jg/.ParaProf/perfdmf)
  - perfexplorer_working (jdbc:derby:/home/jg/.ParaProf/perfexplorer_working)

MetricField	Value
Name	(PAPI_FP_INS...
Application ID	0
Experiment ID	0
Trial ID	0
Metric ID	0

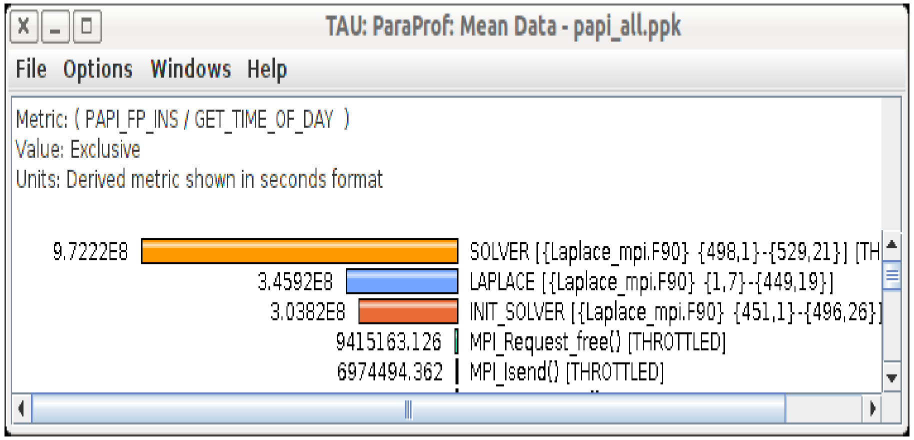
Expression: "(PAPI_FP_INS / GET_TIME_OF_DAY)""(PAPI_FP_INS / GET_TIME_OF_DAY)"


TAU: ParaProf: Mean Data - papi_all.ppk

File Options Windows Help

Metric: (PAPI_FP_INS / GET_TIME_OF_DAY)
Value: Exclusive percent

```
    100%  SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [THROTT
  35.58%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}]
 31.251%  INIT_SOLVER [{Laplace_mpi.F90} {451,1}-{496,26}]
  0.968%  MPI_Request_free() [THROTTLED]
  0.717%  MPI_Isend() [THROTTLED]
  0.429%  MPI_Comm_size()
```


TAU: ParaProf: Mean Data - papi_all.ppk

File Options Windows Help

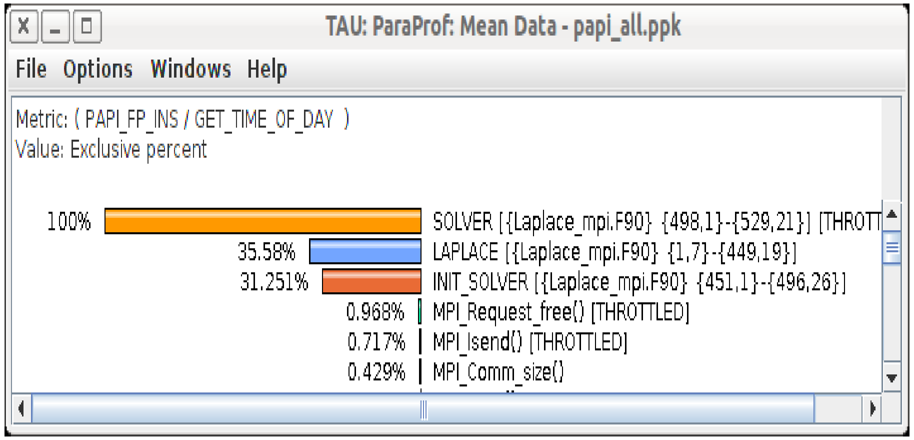
Metric: (PAPI_FP_INS / GET_TIME_OF_DAY)
Value: Exclusive
Units: Derived metric shown in seconds format

```
     9.7222E8  SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [TH
     3.4592E8  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}]
     3.0382E8  INIT_SOLVER [{Laplace_mpi.F90} {451,1}-{496,26}]
  9415163.126  MPI_Request_free() [THROTTLED]
  6974494.362  MPI_Isend() [THROTTLED]
```

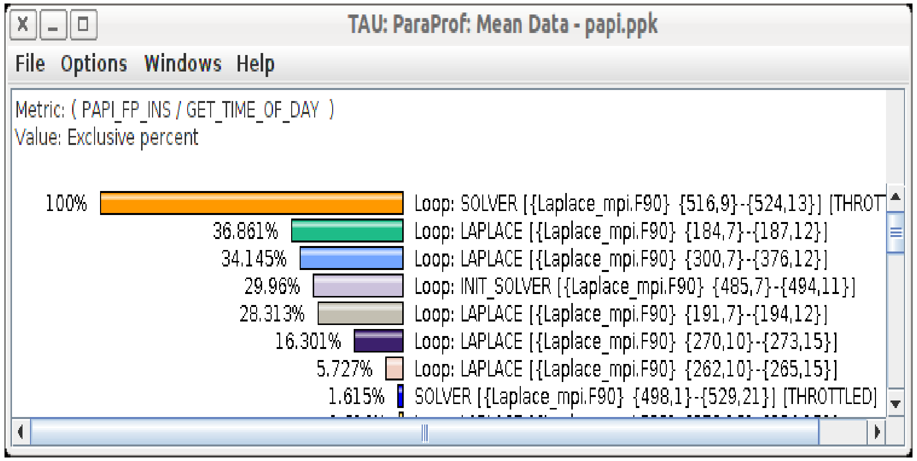
Loops profiling (tau)

```
export TAU_OPTIONS="-optTauSelectFile=select.tau"
export TAU_METRICS="GET_TIME_OF_DAY:PAPI_FP_INS"
paraprof --pack papi.ppk
paraprof papi.ppk
```

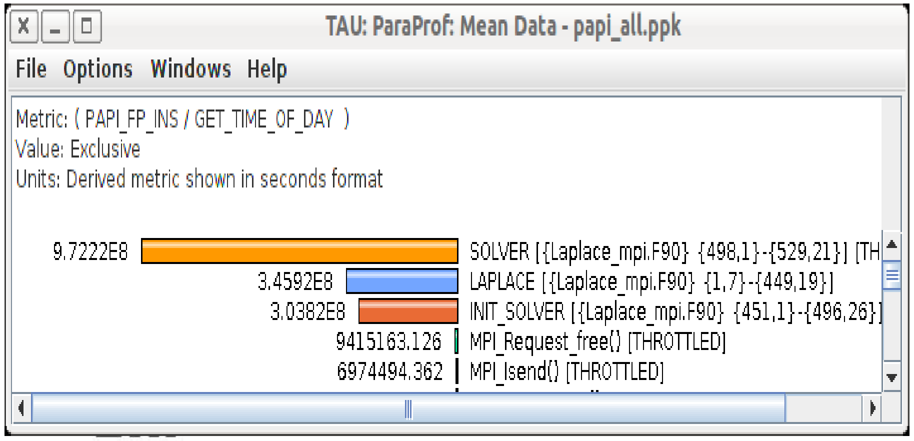
```
BEGIN_INSTRUMENT_SECTION
loops routine="#"
END_INSTRUMENT_SECTION
```


TAU: ParaProf: Mean Data - papi_all.ppk
File Options Windows Help

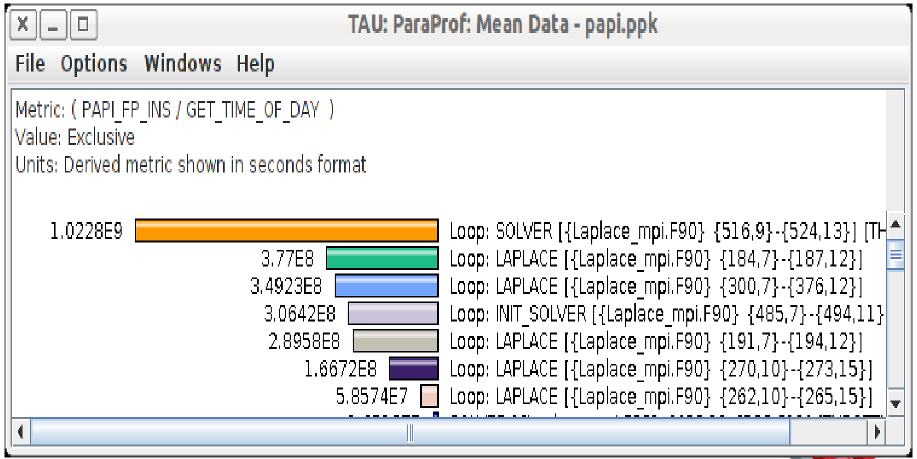
```
Metric: ( PAPI_FP_INS / GET_TIME_OF_DAY )
Value: Exclusive percent

   100%  SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [THROTT
 35.58%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}]
31.251%  INIT_SOLVER [{Laplace_mpi.F90} {451,1}-{496,26}]
 0.968%  MPI_Request_free() [THROTTLED]
 0.717%  MPI_Isend() [THROTTLED]
 0.429%  MPI_Comm_size()
```


TAU: ParaProf: Mean Data - papi.ppk
File Options Windows Help

```
Metric: ( PAPI_FP_INS / GET_TIME_OF_DAY )
Value: Exclusive percent

   100%  Loop: SOLVER [{Laplace_mpi.F90} {516,9}-{524,13}] [THROT
36.861%  Loop: LAPLACE [{Laplace_mpi.F90} {184,7}-{187,12}]
34.145%  Loop: LAPLACE [{Laplace_mpi.F90} {300,7}-{376,12}]
 29.96%  Loop: INIT_SOLVER [{Laplace_mpi.F90} {485,7}-{494,11}]
28.313%  Loop: LAPLACE [{Laplace_mpi.F90} {191,7}-{194,12}]
16.301%  Loop: LAPLACE [{Laplace_mpi.F90} {270,10}-{273,15}]
 5.727%  Loop: LAPLACE [{Laplace_mpi.F90} {262,10}-{265,15}]
 1.615%  SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [THROTTLED]
```


TAU: ParaProf: Mean Data - papi_all.ppk
File Options Windows Help

```
Metric: ( PAPI_FP_INS / GET_TIME_OF_DAY )
Value: Exclusive
Units: Derived metric shown in seconds format

   9.7222E8  SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [TH
   3.4592E8  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}]
   3.0382E8  INIT_SOLVER [{Laplace_mpi.F90} {451,1}-{496,26}]
9415163.126  MPI_Request_free() [THROTTLED]
6974494.362  MPI_Isend() [THROTTLED]
```

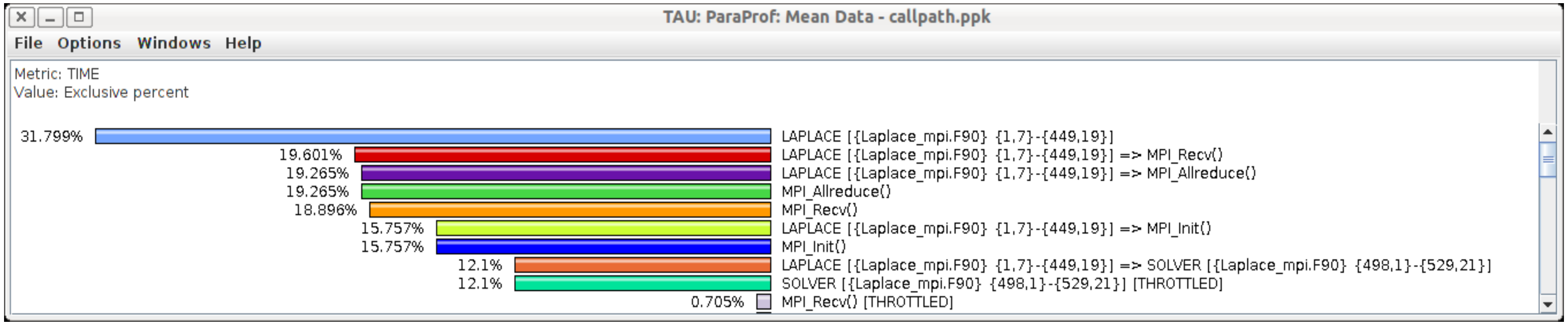

TAU: ParaProf: Mean Data - papi.ppk
File Options Windows Help

```
Metric: ( PAPI_FP_INS / GET_TIME_OF_DAY )
Value: Exclusive
Units: Derived metric shown in seconds format

1.0228E9  Loop: SOLVER [{Laplace_mpi.F90} {516,9}-{524,13}] [TH
  3.77E8  Loop: LAPLACE [{Laplace_mpi.F90} {184,7}-{187,12}]
3.4923E8  Loop: LAPLACE [{Laplace_mpi.F90} {300,7}-{376,12}]
3.0642E8  Loop: INIT_SOLVER [{Laplace_mpi.F90} {485,7}-{494,11}]
2.8958E8  Loop: LAPLACE [{Laplace_mpi.F90} {191,7}-{194,12}]
1.6672E8  Loop: LAPLACE [{Laplace_mpi.F90} {270,10}-{273,15}]
5.8574E7  Loop: LAPLACE [{Laplace_mpi.F90} {262,10}-{265,15}]
```

CALLPATH (tau)

```
export TAU_CALLPATH=1
```

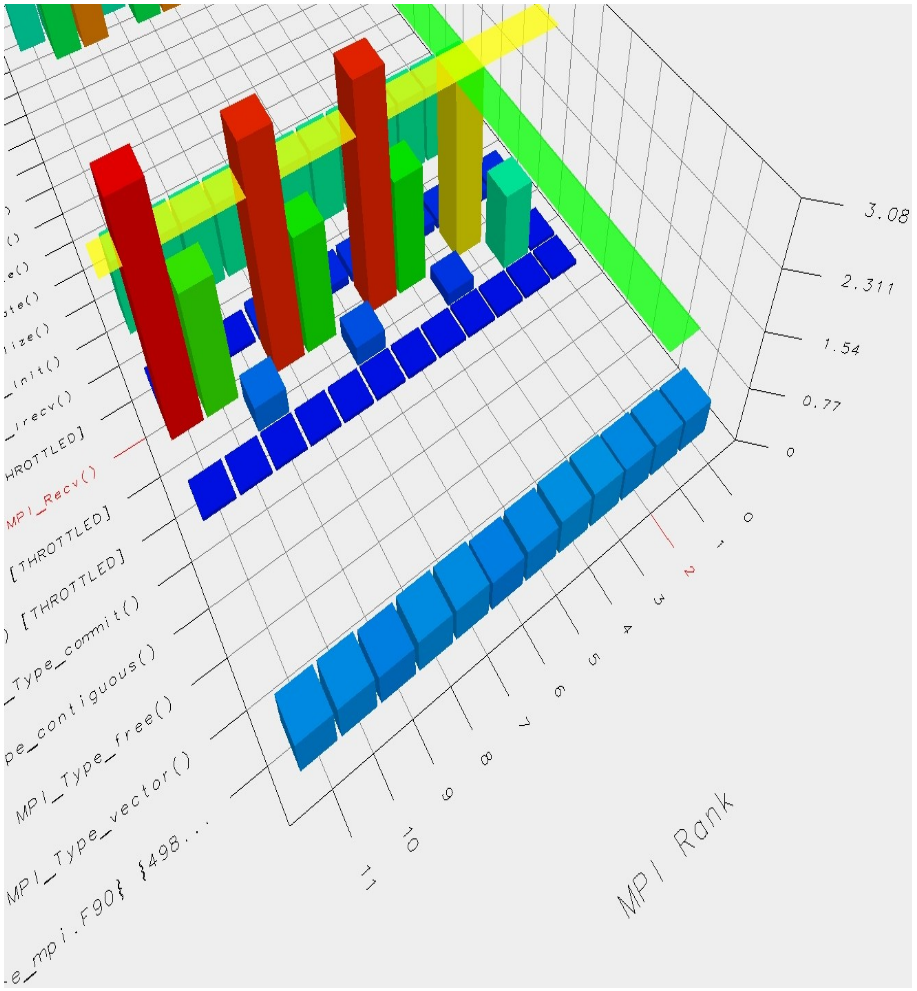
```
paraprof --pack callpath.ppk
```

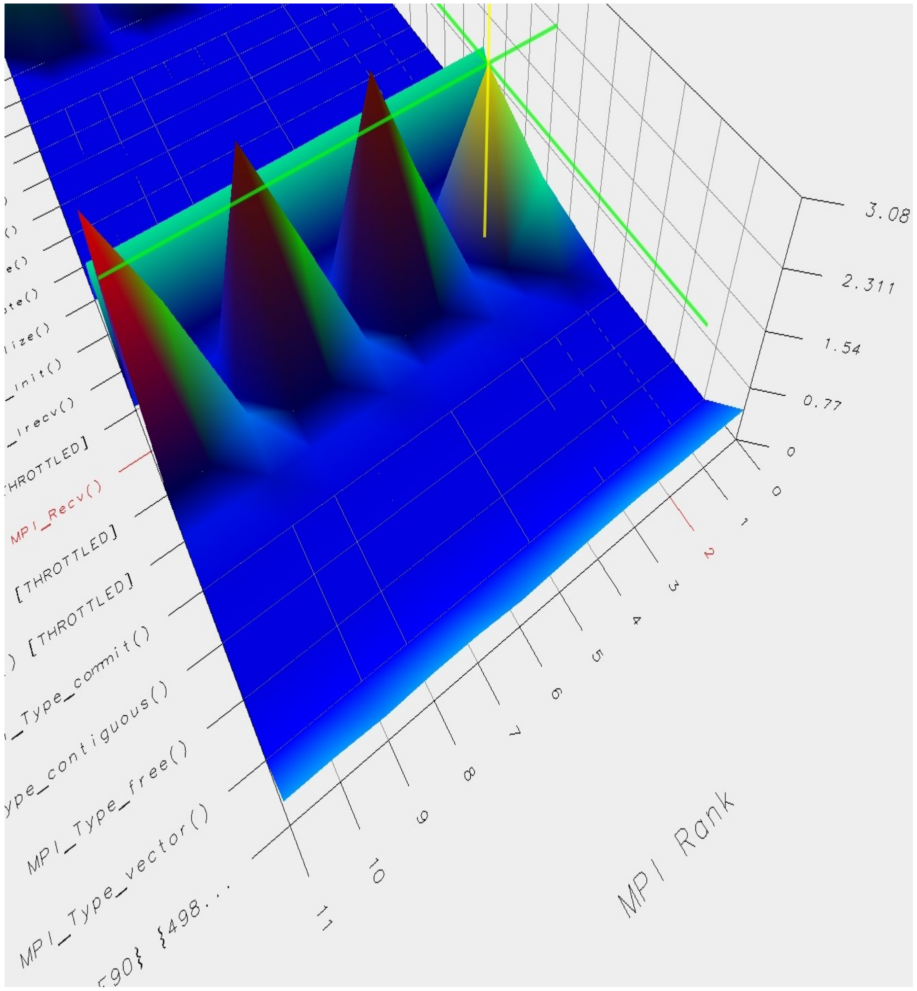


```
Metric: TIME
Value: Exclusive percent

31.799%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}]
19.601%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}] => MPI_Recv()
19.265%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}] => MPI_Allreduce()
19.265%  MPI_Allreduce()
18.896%  MPI_Recv()
15.757%  LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}] => MPI_Init()
15.757%  MPI_Init()
12.1%    LAPLACE [{Laplace_mpi.F90} {1,7}-{449,19}] => SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}]
12.1%    SOLVER [{Laplace_mpi.F90} {498,1}-{529,21}] [THROTTLED]
0.705%   MPI_Recv() [THROTTLED]
```

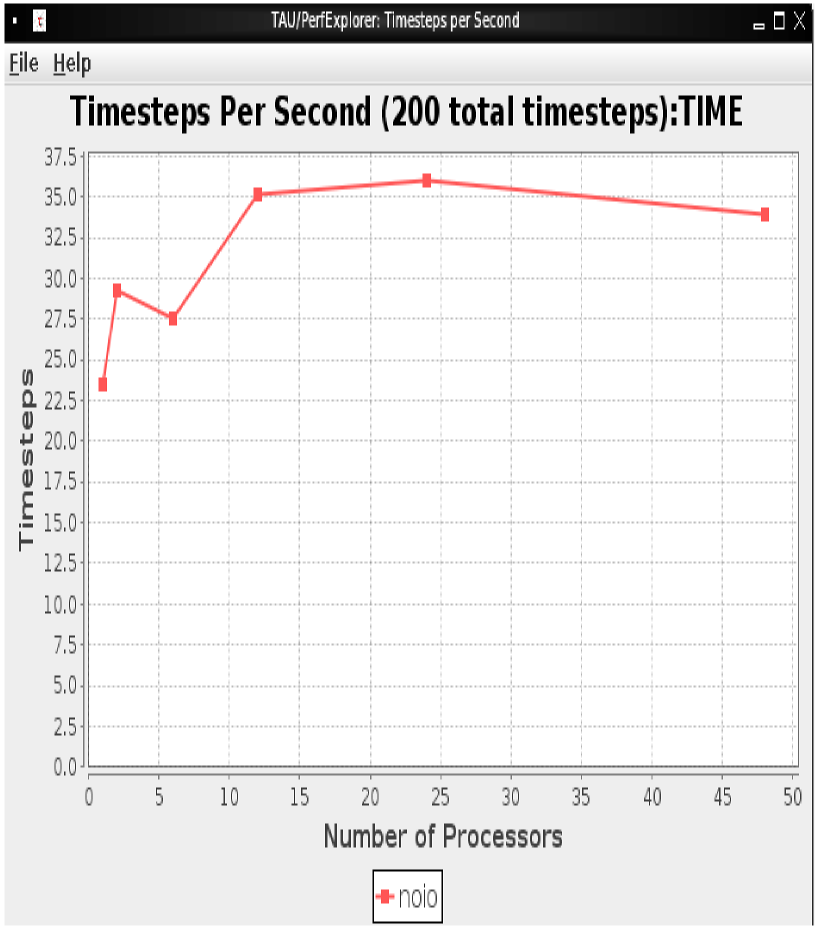
```
n1:~/trunk/laplace/src/tau > pprof -s
Reading Profile files in profile.*

FUNCTION SUMMARY (total):
---------------------------------------------------------------------------------------
%Time    Exclusive     Inclusive        #Call        #Subrs  Inclusive Name
              msec    total msec                             usec/call
---------------------------------------------------------------------------------------
100.0       12,451        39,157            6   2.68034E+06    6526308 LAPLACE
 19.6        7,675         7,675       876603             0          9 LAPLACE => MPI_Recv()
 19.3        7,543         7,543         1200             0       6287 LAPLACE => MPI_Allreduce()
 19.3        7,543         7,543         1200             0       6287 MPI_Allreduce()
 18.9        7,399         7,399       576600             0         13 MPI_Recv()
 15.8        6,170         6,170            6             0    1028347 LAPLACE => MPI_Init()
 15.8        6,170         6,170            6             0    1028347 MPI_Init()
 12.1        4,738         4,738       600006             0          8 LAPLACE => SOLVER
 12.1        4,738         4,738       600006             0          8 SOLVER [THROTTLED]
  0.7          276           276       300003             0          1 MPI_Recv() [THROTTLED]
```

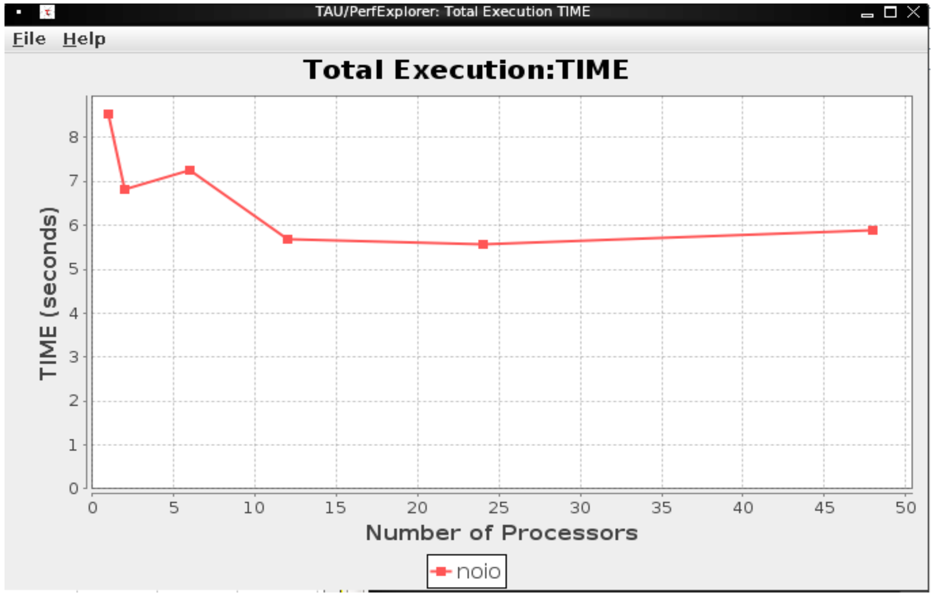
PARAPROF 3D (tau)

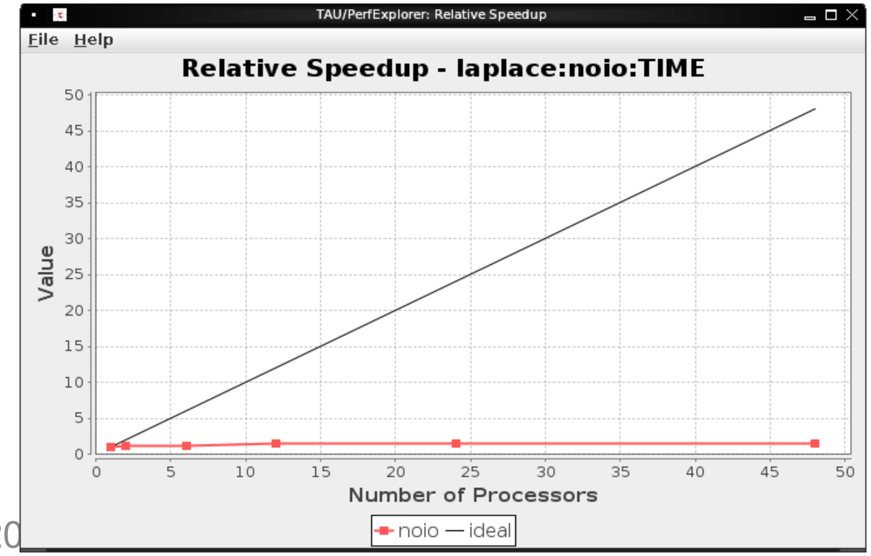




PerfExplorer (tau)







Selective instrumentation/profiling (tau)

```
export TAU_OPTIONS="-optTauSelectFile=select.tau"
```

```
BEGIN_INSTRUMENT_SECTION
loops routine="#"
END_INSTRUMENT_SECTION
```

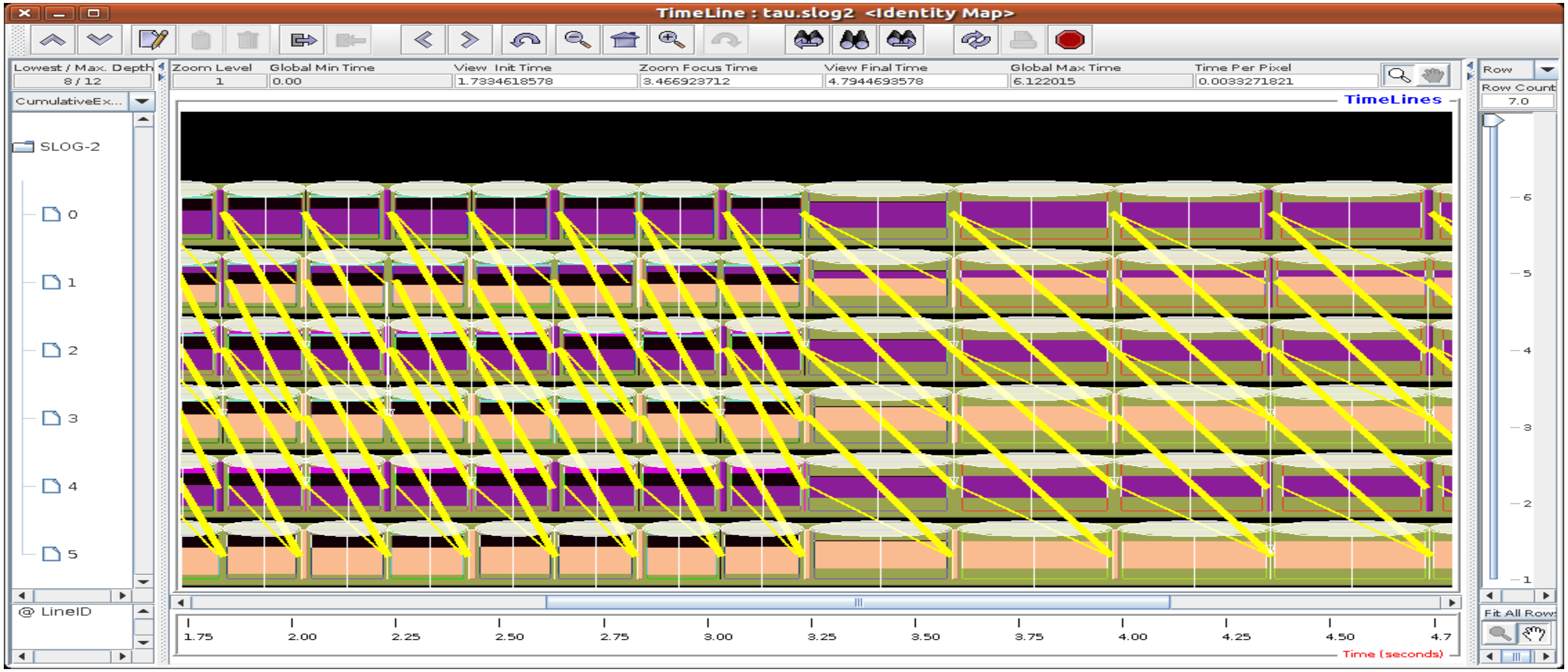
```
n1:/home/piccinali/trunk/laplace/src $  make clean ; make FCOMPILER=tau_f90.sh FFLAGS="-O3 -w"
```

```
n1:/home/piccinali/trunk/laplace/src/tau $  pprof -s
Reading Profile files in profile.*

FUNCTION SUMMARY (total):
---------------------------------------------------------------------------------------
%Time    Exclusive    Inclusive       #Call      #Subrs  Inclusive Name
              msec   total msec                          usec/call
---------------------------------------------------------------------------------------
100.0           25       38,646           6         156    6441096 LAPLACE
 83.4       12,414       32,213           6 2.68022E+06    5368984 Loop: LAPLACE [{Laplace_mpi.F90} {300,7}-{376,12}]
 18.9        7,304        7,304        1200           0       6087 MPI_Allreduce()
 18.5        7,165        7,165      576600           0         12 MPI_Recv()
 16.0        6,172        6,172           6           0    1028826 MPI_Init()
 12.6           88        4,872      600006      600006          8 SOLVER [THROTTLED]
 12.4        4,784        4,784      600006           0          8 Loop: SOLVER [{Laplace_mpi.F90} {516,9}-{524,13}]
```

Tracing (tau)

```
export TAU_TRACE=1
```

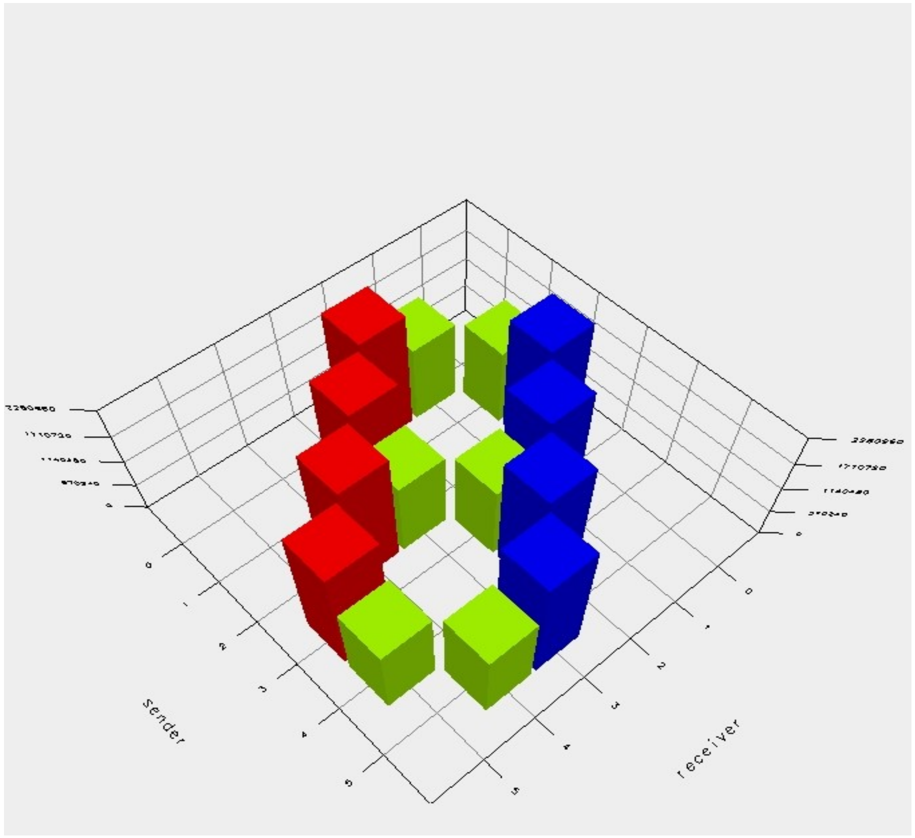
```
sbatch.sh ../exe.tau 6 6 1 "1920 1920 200 1.0d-5" "" "-bind-to-core"
tau2slog2 tau.trc tau.edf -o tau.slog2
jumpshot tau.slog2
```

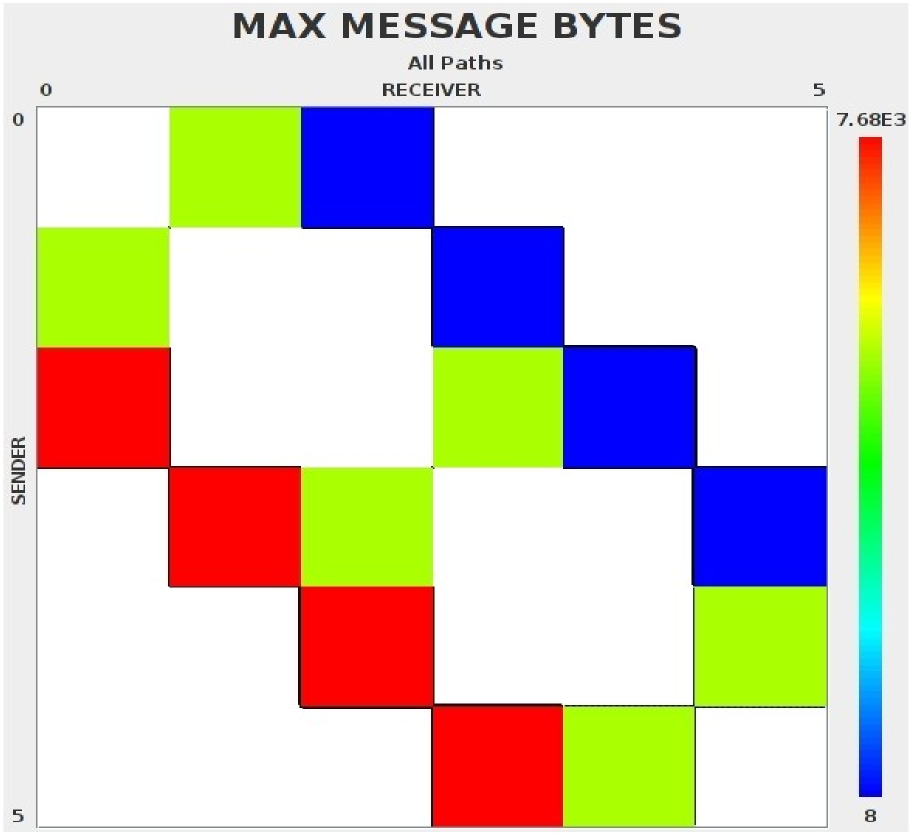


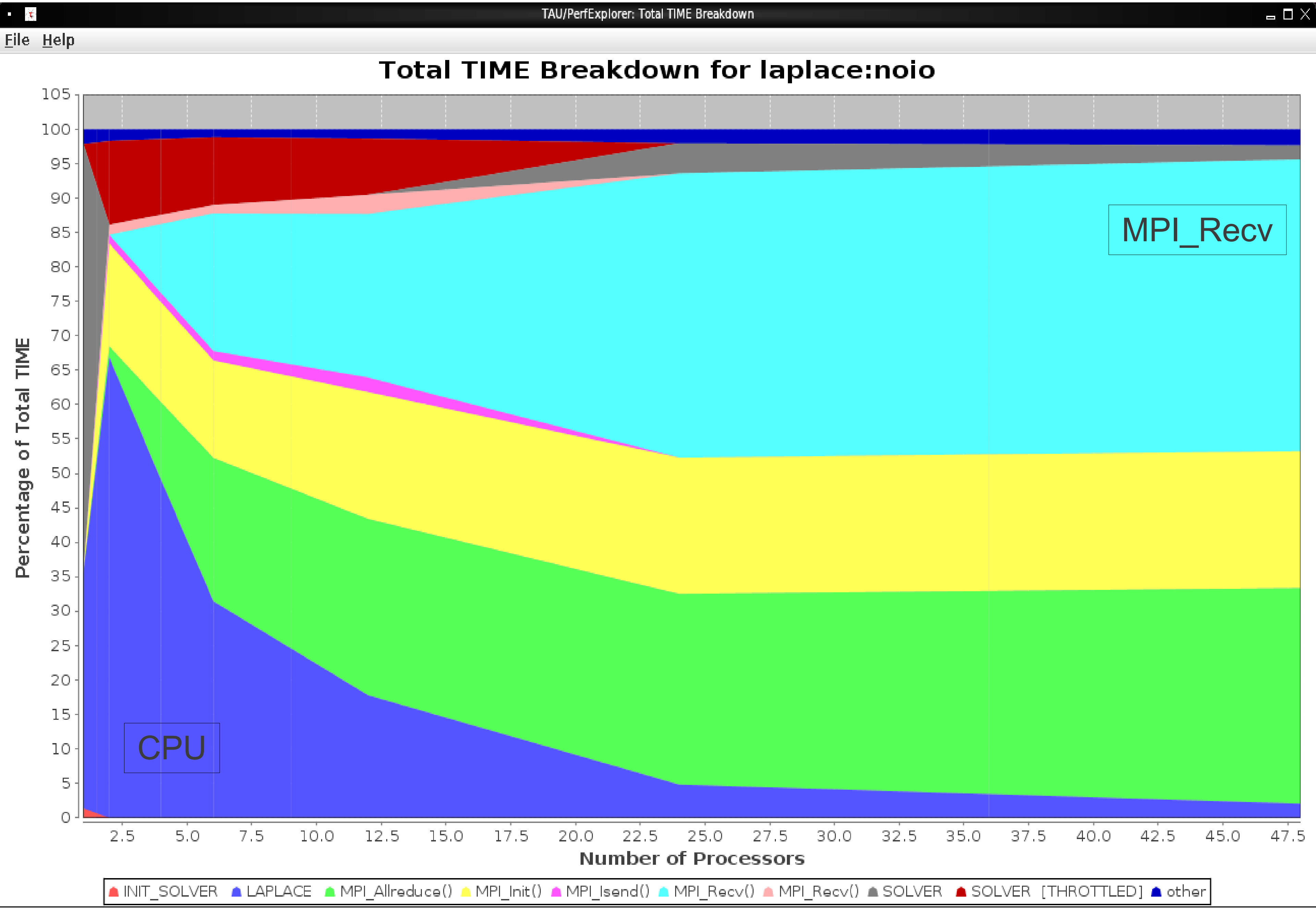
Communication matrix (tau)

```
export TAU_TRACE=0 ; export TAU_COMM_MATRIX=1
```

```
paraprof
```





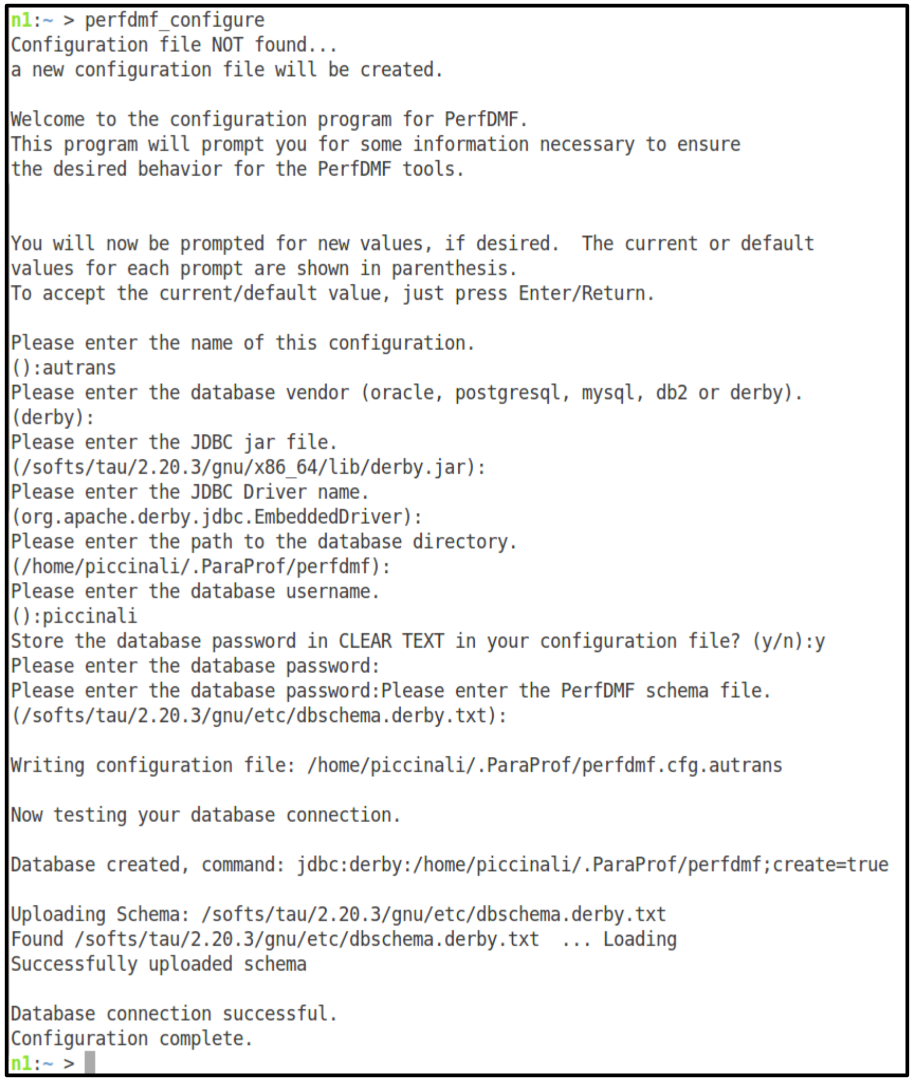

TAU/PerfExplorer: Total TIME Breakdown
File Help
Total TIME Breakdown for laplace:noio
Percentage of Total TIME
Number of Processors
0
5
10
15
20
25
30
35
40
45
50
55
60
65
70
75
80
85
90
95
100
105
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
22.5
25.0
27.5
30.0
32.5
35.0
37.5
40.0
42.5
45.0
47.5
MPI_Recv
CPU
INIT_SOLVER
LAPLACE
MPI_Allreduce()
MPI_Init()
MPI_Isend()
MPI_Recv()
MPI_Recv()
SOLVER
SOLVER [THROTTLED]
other

References

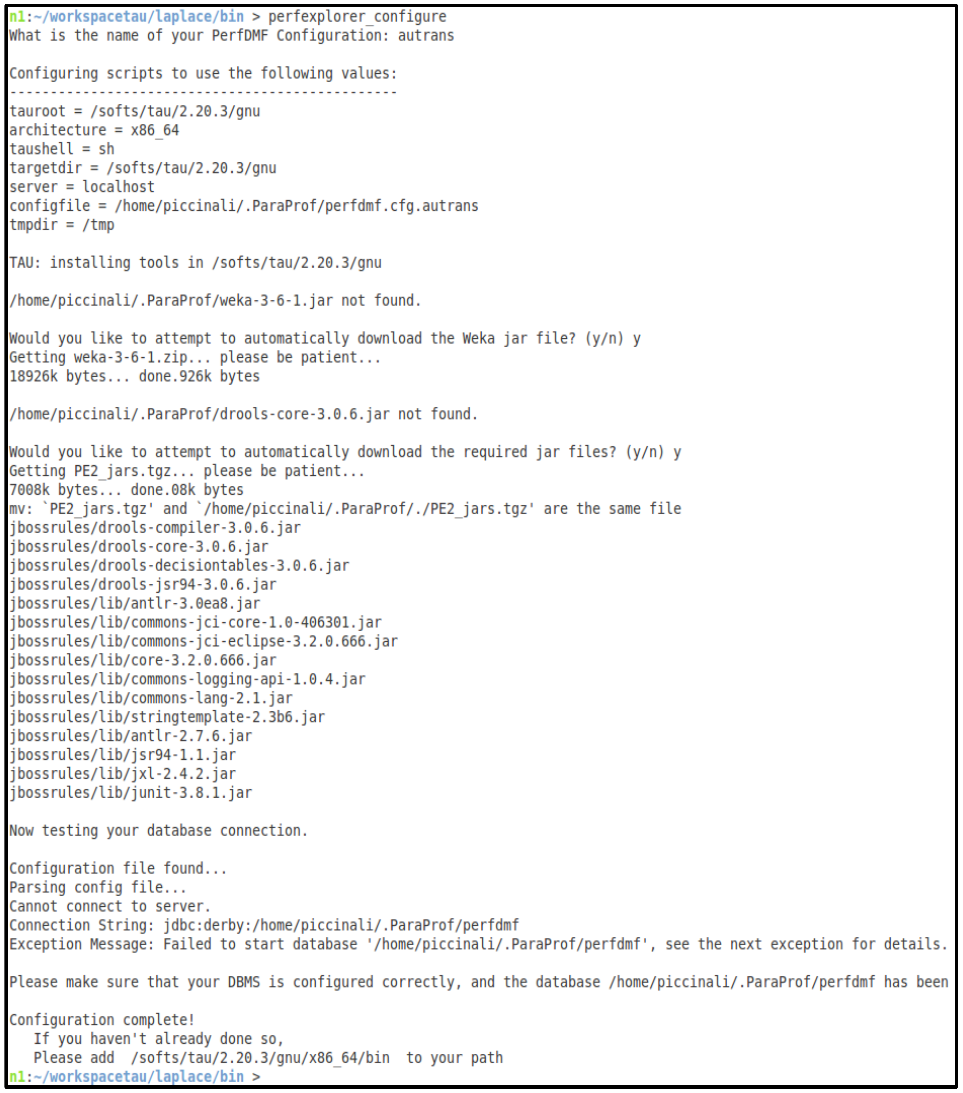
- TAU website
  - http://www.cs.uoregon.edu/research/tau/home.php
  - Documentation
  - Tutorials
  - Downloads (offline analysis utilities)

- CSCS Rosa website
  - http://www.cscs.ch > User Entry Point
    - > Software and Programming Environment
      - > Debugging and Performance Analysis
        - > Performance
          - > TAU
  - Module setup
  - Usage information

TAU : Perfdmf / Perfexplorer

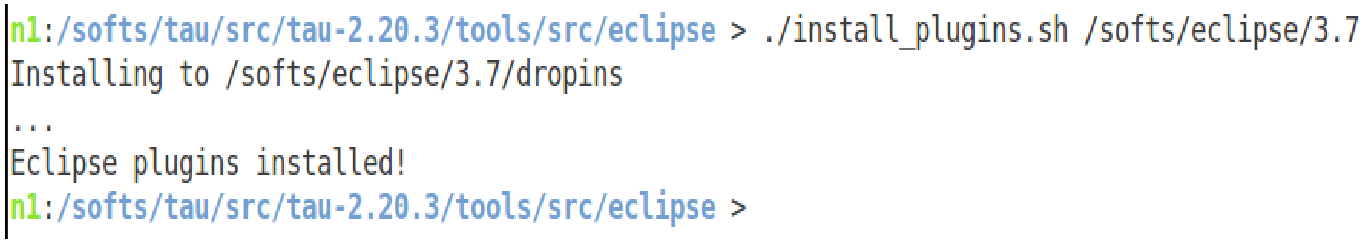


```
n1:~ > perfdmf_configure
Configuration file NOT found...
a new configuration file will be created.

Welcome to the configuration program for PerfDMF.
This program will prompt you for some information necessary to ensure
the desired behavior for the PerfDMF tools.


You will now be prompted for new values, if desired.  The current or default
values for each prompt are shown in parenthesis.
To accept the current/default value, just press Enter/Return.

Please enter the name of this configuration.
():autrans
Please enter the database vendor (oracle, postgresql, mysql, db2 or derby).
(derby):
Please enter the JDBC jar file.
(/softs/tau/2.20.3/gnu/x86_64/lib/derby.jar):
Please enter the JDBC Driver name.
(org.apache.derby.jdbc.EmbeddedDriver):
Please enter the path to the database directory.
(/home/piccinali/.ParaProf/perfdmf):
Please enter the database username.
():piccinali
Store the database password in CLEAR TEXT in your configuration file? (y/n):y
Please enter the database password:
Please enter the database password:Please enter the PerfDMF schema file.
(/softs/tau/2.20.3/gnu/etc/dbschema.derby.txt):

Writing configuration file: /home/piccinali/.ParaProf/perfdmf.cfg.autrans

Now testing your database connection.

Database created, command: jdbc:derby:/home/piccinali/.ParaProf/perfdmf;create=true

Uploading Schema: /softs/tau/2.20.3/gnu/etc/dbschema.derby.txt
Found /softs/tau/2.20.3/gnu/etc/dbschema.derby.txt  ... Loading
Successfully uploaded schema

Database connection successful.
Configuration complete.
n1:~ >
```



```
n1:~/workspacetau/laplace/bin > perfexplorer_configure
What is the name of your PerfDMF Configuration: autrans

Configuring scripts to use the following values:
------------------------------------------------
tauroot = /softs/tau/2.20.3/gnu
architecture = x86_64
taushell = sh
targetdir = /softs/tau/2.20.3/gnu
server = localhost
configfile = /home/piccinali/.ParaProf/perfdmf.cfg.autrans
tmpdir = /tmp

TAU: installing tools in /softs/tau/2.20.3/gnu

/home/piccinali/.ParaProf/weka-3-6-1.jar not found.

Would you like to attempt to automatically download the Weka jar file? (y/n) y
Getting weka-3-6-1.zip... please be patient...
18926k bytes... done.926k bytes

/home/piccinali/.ParaProf/drools-core-3.0.6.jar not found.

Would you like to attempt to automatically download the required jar files? (y/n) y
Getting PE2_jars.tgz... please be patient...
7008k bytes... done.08k bytes
mv: `PE2_jars.tgz' and `/home/piccinali/.ParaProf/./PE2_jars.tgz' are the same file
jbossrules/drools-compiler-3.0.6.jar
jbossrules/drools-core-3.0.6.jar
jbossrules/drools-decisiontables-3.0.6.jar
jbossrules/drools-jsr94-3.0.6.jar
jbossrules/lib/antlr-3.0ea8.jar
jbossrules/lib/commons-jci-core-1.0-406301.jar
jbossrules/lib/commons-jci-eclipse-3.2.0.666.jar
jbossrules/lib/core-3.2.0.666.jar
jbossrules/lib/commons-logging-api-1.0.4.jar
jbossrules/lib/commons-lang-2.1.jar
jbossrules/lib/stringtemplate-2.3b6.jar
jbossrules/lib/antlr-2.7.6.jar
jbossrules/lib/jsr94-1.1.jar
jbossrules/lib/jxl-2.4.2.jar
jbossrules/lib/junit-3.8.1.jar

Now testing your database connection.

Configuration file found...
Parsing config file...
Cannot connect to server.
Connection String: jdbc:derby:/home/piccinali/.ParaProf/perfdmf
Exception Message: Failed to start database '/home/piccinali/.ParaProf/perfdmf', see the next exception for details.

Please make sure that your DBMS is configured correctly, and the database /home/piccinali/.ParaProf/perfdmf has been

Configuration complete!
   If you haven't already done so,
   Please add  /softs/tau/2.20.3/gnu/x86_64/bin  to your path
n1:~/workspacetau/laplace/bin >
```

TAU → ECLIPSE



```
n1:/softs/tau/src/tau-2.20.3/tools/src/eclipse > ./install_plugins.sh /softs/eclipse/3.7
Installing to /softs/eclipse/3.7/dropins
...
Eclipse plugins installed!
n1:/softs/tau/src/tau-2.20.3/tools/src/eclipse >
```

VAMPIR (http://www.vampir.eu)

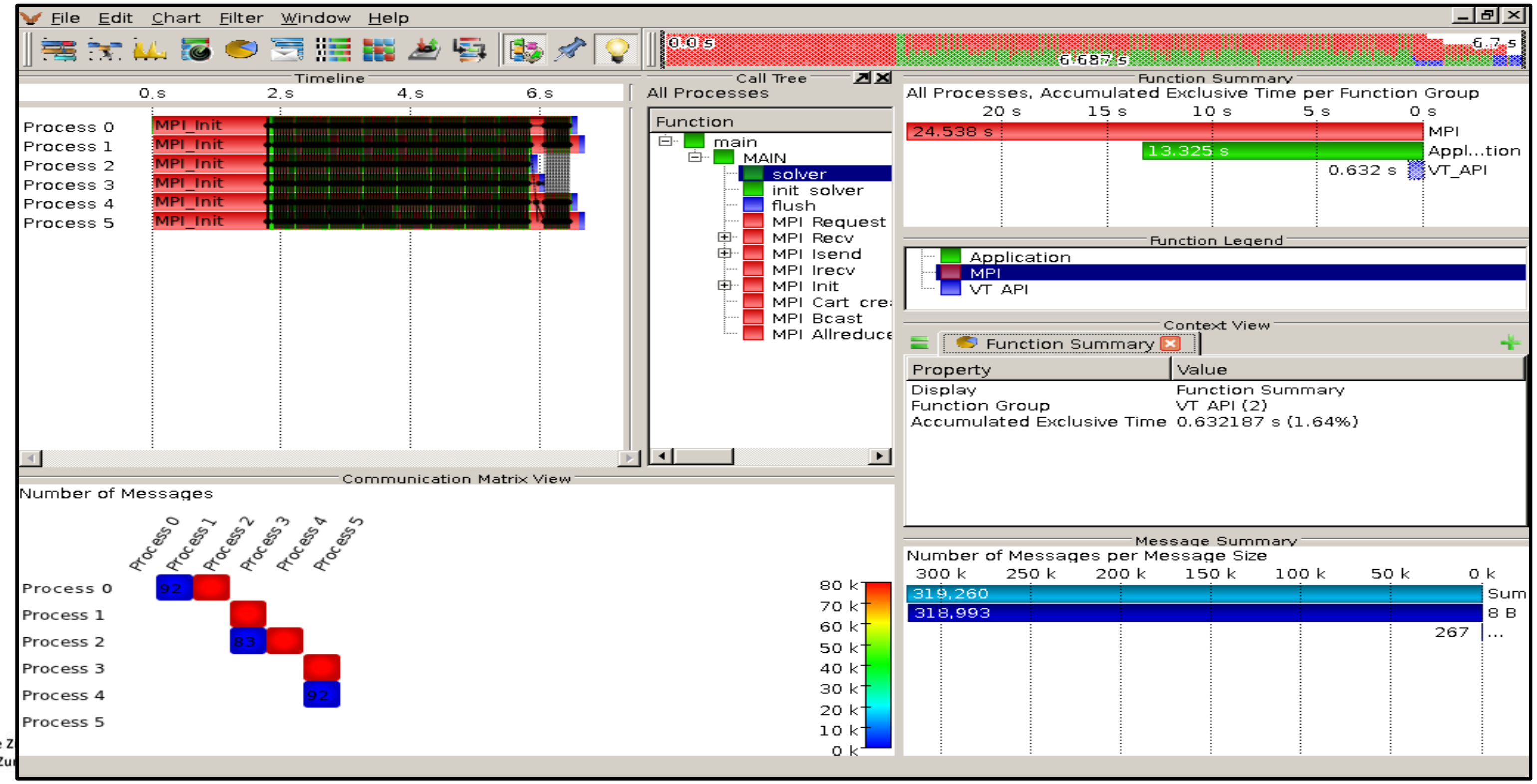
```
n1:/home/piccinali/trunk/laplace/src/vampir $ which mpif90-vt
/softs/openmpi-1.4.3/bin/mpif90-vt
```

```
n1:/home/piccinali/trunk/laplace/src $ make clean ; make FCOMPILER=mpif90-vt FFLAGS="-O3 -w" PROGRAM=exe.vampir
rm -f exe mem_ntk.o Laplace_mpi.o
mpicc -O3 -w -c mem_ntk.c
mpif90-vt -O3 -w -c Laplace_mpi.F90
mpif90-vt -O3 -w -o exe.vampir mem_ntk.o Laplace_mpi.o
```

```
sbatch.sh ../exe.vampir 6 6 1 "1920 1920 200 1.0d-5" "" "-bind-to-core"
```

```
a.0.def
a.1.events
a.2.events
a.3.events
a.4.events
a.5.events
a.otf
a.6.events
```

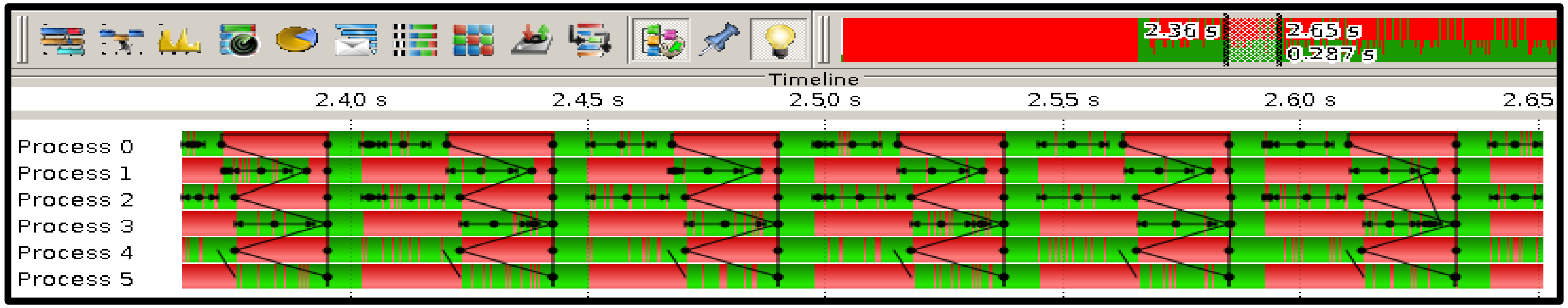
```
module load vampir
```

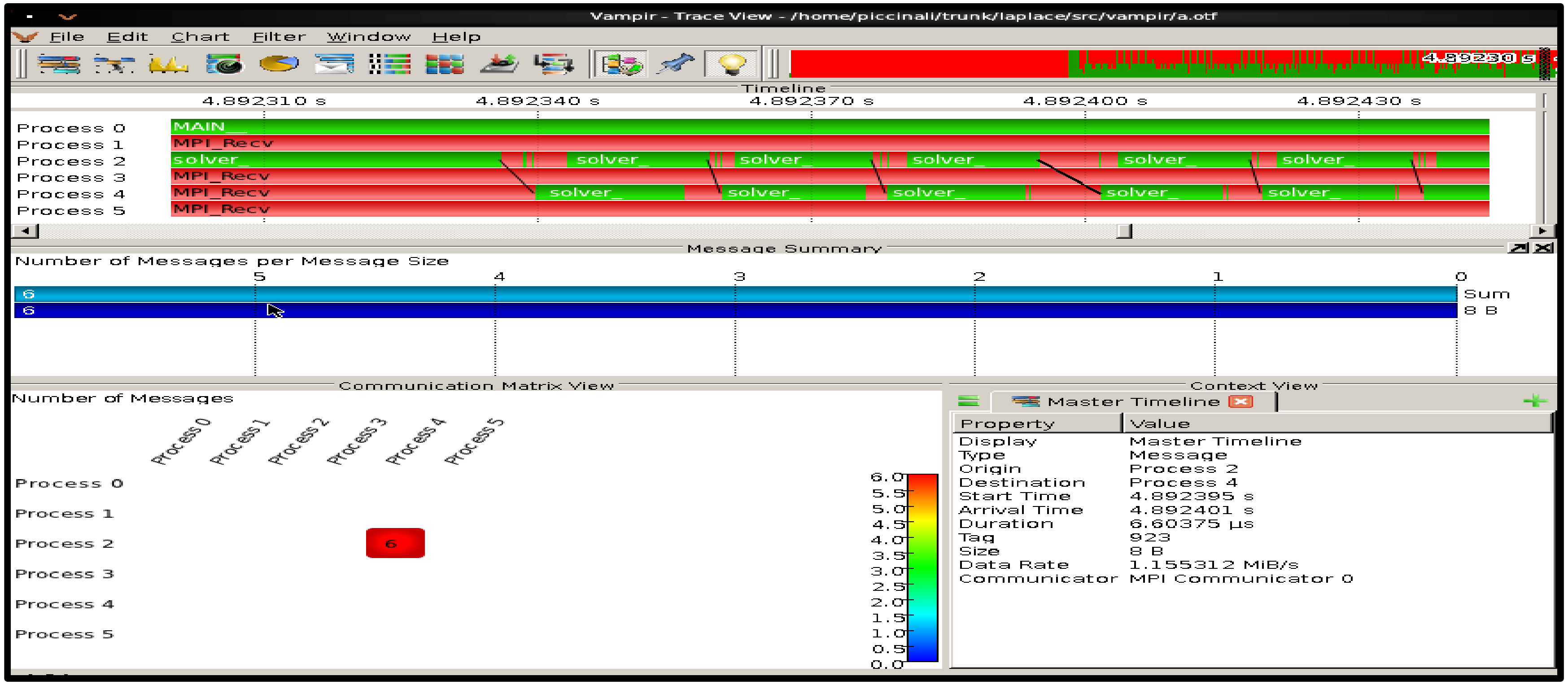


Property	Value
Display	Function Summary
Function Group	VT API (2)
Accumulated Exclusive Time	0.632187 s (1.64%)

ETH
Eidgenössische Technische Hochschule Z
Swiss Federal Institute of Technology Zu

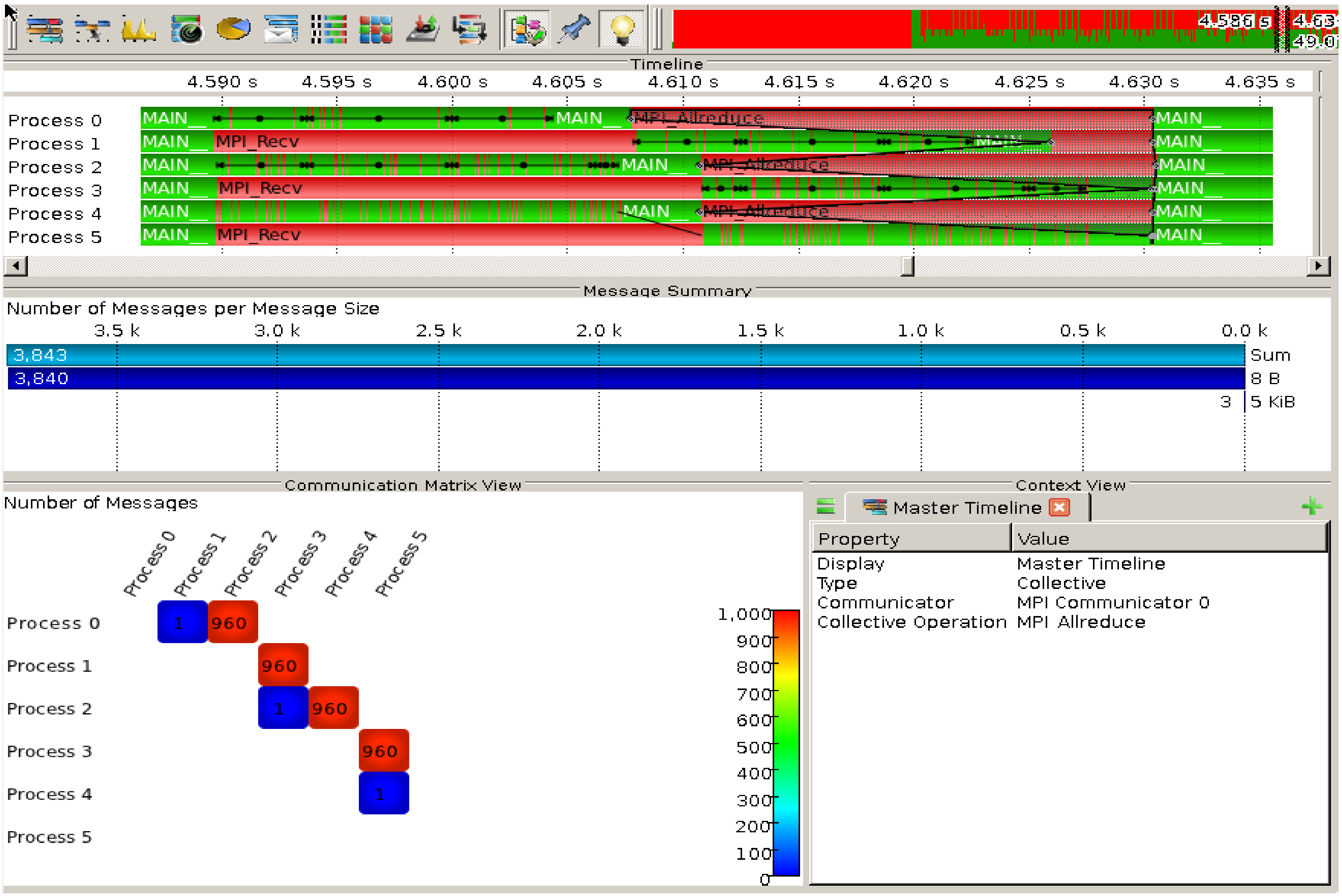
VAMPIR





Property	Value
Display	Master Timeline
Type	Message
Origin	Process 2
Destination	Process 4
Start Time	4.892395 s
Arrival Time	4.892401 s
Duration	6.60375 µs
Tag	923
Size	8 B
Data Rate	1.155312 MiB/s
Communicator	MPI Communicator 0

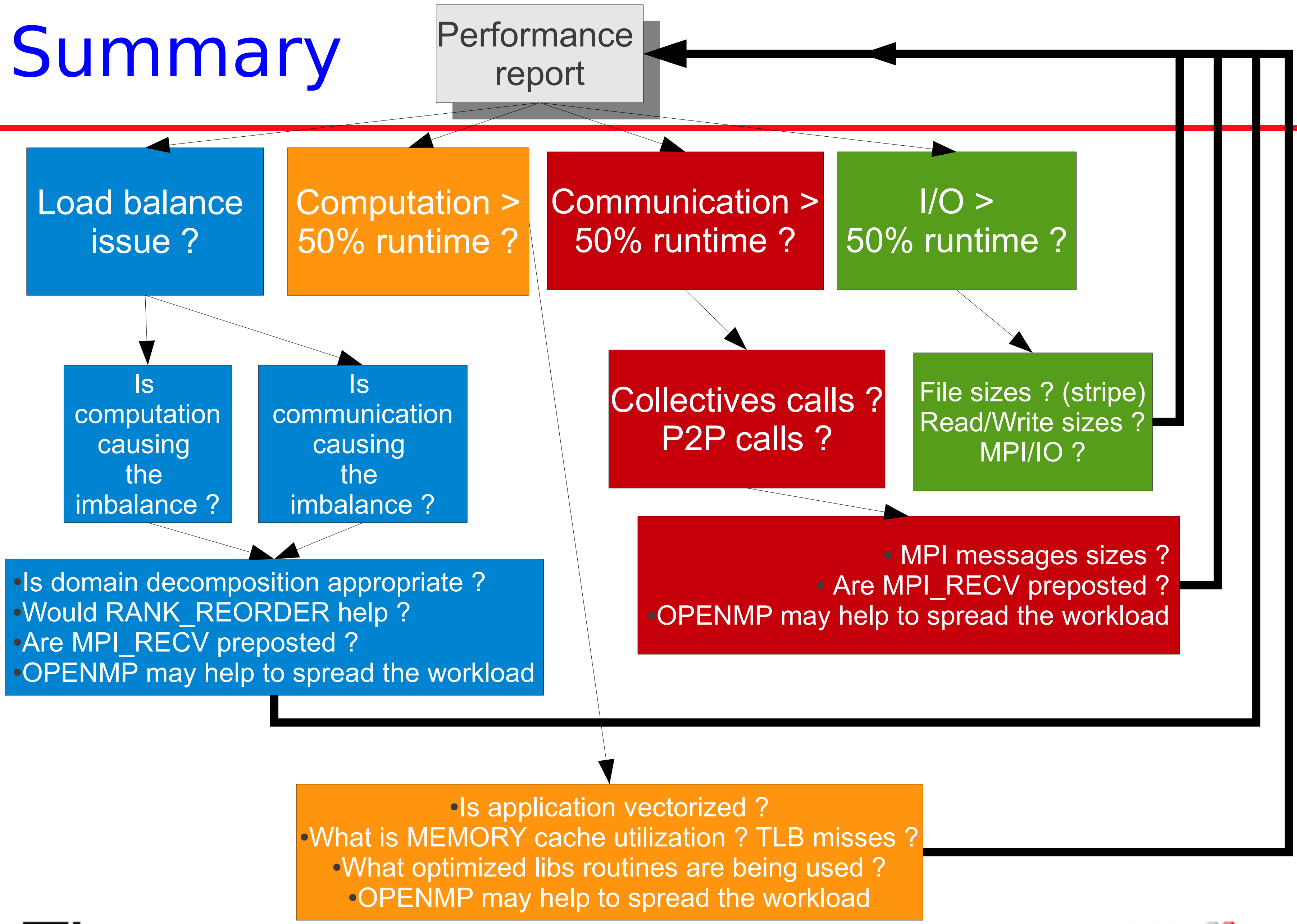
VAMPIR



	Process 0	Process 1	Process 2	Process 3	Process 4	Process 5
Process 0	1	960				
Process 1			960			
Process 2			1	960		
Process 3					960	
Process 4					1	
Process 5						

Property	Value
Display	Master Timeline
Type	Collective
Communicator	MPI Communicator 0
Collective Operation	MPI_Allreduce

Summary


Performance report
Load balance issue ?
Computation > 50% runtime ?
Communication > 50% runtime ?
I/O > 50% runtime ?
Is computation causing the imbalance ?
Is communication causing the imbalance ?
Collectives calls ? P2P calls ?
File sizes ? (stripe) Read/Write sizes ? MPI/IO ?
•Is domain decomposition appropriate ?
•Would RANK_REORDER help ?
•Are MPI_RECV preposted ?
•OPENMP may help to spread the workload
• MPI messages sizes ?
• Are MPI_RECV preposted ?
•OPENMP may help to spread the workload
•Is application vectorized ?
•What is MEMORY cache utilization ? TLB misses ?
•What optimized libs routines are being used ?
•OPENMP may help to spread the workload