جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

SHAHEEN
SUPERCOMPUTING LABORATORY

KAUST Supercomputing Laboratory

Introduction to Performance Analysis tools on Shaheen II

George Markomanolis
Computational Scientist
April 17th, 2016

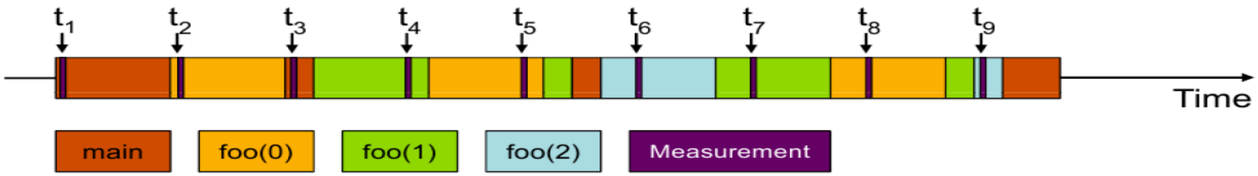
Outline

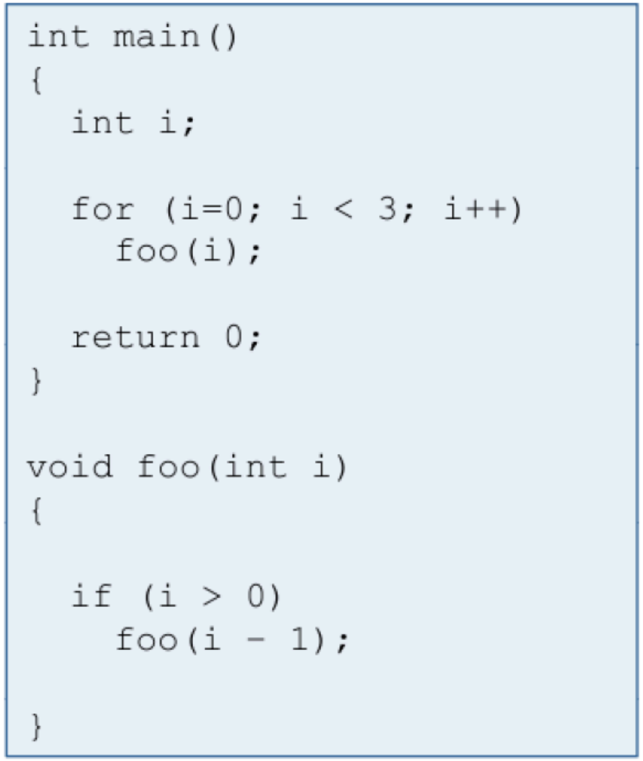
- ❖ Introduction
- ❖ Test cases
- ❖ Cray tools
  - Perftools
  - Cray Apprentice 2
  - Reveal
- ❖ Extrae/Paraver (briefly)

Introduction

- Why performance analysis?
  - Investigate the bottlenecks of an application
  - Identify potential improvements
  - Better usage of the hardware
- Profiling
  - Sampling
    - Lightweight
    - Overhead depends on the sampling frequency
    - Can lack resolution if there are small function calls
  - Event Tracing
    - Detailed information
    - Captures every event
    - Can capture communication events
    - Drawbacks, overhead and large amounts of data

Sampling

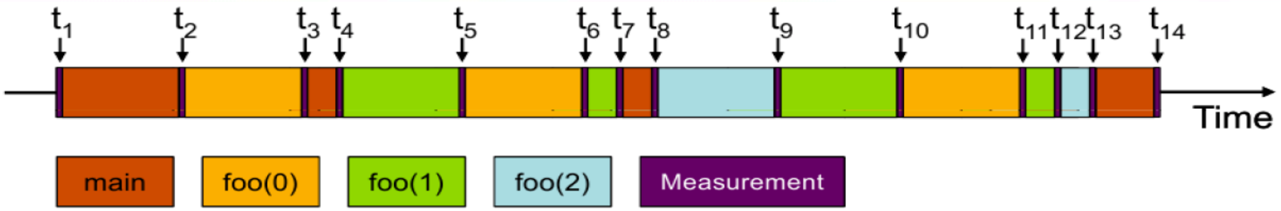


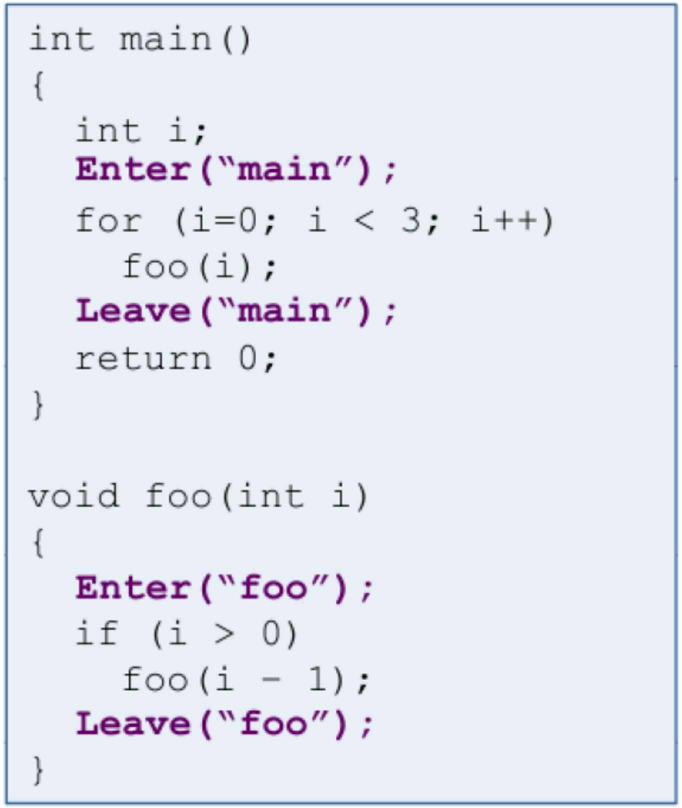


```
int main()
{
  int i;

  for (i=0; i < 3; i++)
    foo(i);

  return 0;
}

void foo(int i)
{

  if (i > 0)
    foo(i - 1);

}
```

- Statistical inference of program behavior
- Not very detailed information
- Mainly for long-running applications

Tracing





```
int main()
{
  int i;
  Enter("main");
  for (i=0; i < 3; i++)
    foo(i);
  Leave("main");
  return 0;
}

void foo(int i)
{
  Enter("foo");
  if (i > 0)
    foo(i - 1);
  Leave("foo");
}
```

- Every event is captured
- Detailed information
- Overhead (depends on many factors)

Studying case

- NAS Parallel Benchmarks (NPB) consist of five kernels and three pseudo-applications, developed by NASA Advanced Supercomputing Division

- Why NPB/LU?
  - LU stands for Lower-Upper Gauss-Seidel solver
  - Simple application for testing purposes which combines computation and communication
  - Compile with Cray, Intel, GNU compilers and fast

CrayPat overview

- ❖ Assist the user with application performance analysis and optimization
  - Provides concrete suggestions instead of just reporting
- ❖ Basic functionalities apply for all the compilers on the system
- ❖ Requires no source code or Makefile modification (for most of the cases)

Components of CrayPat

- Module perftools-base
  - pat_build – Instruments the program to be analyzed
  - pat_report – Generates text reports from the performance data captured during program execution and exports data for use in other programs.
  - Cray Apprentice2 – A graphical analysis tool that can be used to visualize and explore the performance data captured during program, execution
  - Reveal – A graphical source code analysis tool that can be used to correlate performance analysis data with annotated source code listings, to identify key opportunities for optimization (it works only with Cray compiler)
  - grid_order – Generates MPI rank order information that can be used with the MPICH_RANK_REORDER
  - pat_help – Help system which provides extensive usage information

Files generated during regular profiling

- A.out+pat+PID-node[s|t].xf: raw data files
  - Depending on the profiling approach and conditions the execution of an instrumented application can create one or more .xf files where:
    - a.out is the name of the original program
    - PID is the process ID assigned to the instrumented program at runtime
    - Node is the physical node ID upon which the rank zero process executed
    - s|t is a letter code indicating the type of experiment performed, either **s** for sampling or **t** for tracing
  - Pat_report tool dump the .xf file or export to another file format for use with other applications, i.e, *.ap2 files
- *.ap2 files: self contained compressed performance files
  - Normally about 5 times smaller than the corresponding *.xf files
  - Only one *.ap2 per experiment in comparison to potentially multiple *.xf files

Prepare for the tutorial

- Connect to Shaheen II and copy the material:
  - ssh –X username@shaheen.kaust.edu.sa
  - cp /scratch/tmp/performance_workshop.tgz **.**
  - tar zxvf performance_workshop.tgz
  - cd performance_workshop/NPB3.3-MPI
  - slides located in the folder performance_workshop/

How to use CrayPat

- Load Perftools
  - *module unload darshan*
  - *module load perftools-base/6.3.2*
  - *module load perftools/6.3.2*

- Compile the code
  - *make clean*
  - *make LU NPROCS=64 CLASS=C*
    - *"WARNING: PerfTools is saving object files from a temporary directory into directory…"*
  - *cd bin*
- The new binary is called lu.C.64 is not instrumented yet

Sampling instrumentation I

- Execute the application
  - *sbatch --reservation=*s1001_85 *submit.sh*
  - Check the output files (lu_C_64_out_...txt)

- Build the instrumented binary with sampling instrumentation
  - *pat_build –S lu.C.64*

- The instrumented binary is called *lu.C.64+pat*
- *Some results of the current presentation are acquired with 128 MPI processes.*

Sampling instrumentation II

- Edit the submit.sh file, comment line 13 and uncomment line 16
  - *sbatch --reservation=*s1001_85 *submit.sh*
  - *The reservation of the nodes for this workshop is called s1001_85, you need to use it every time you submit jobs during this presentation.*

- The performance data are locate in a file called with the format
  lu.C.64+PID-XXX**s**.xf (PID and XXX are numbers)

Create your first report with sampling instrumentation

- Execute the pat_report tool
  - *pat_report –o sampling_report_lu_C_64.txt lu.C.64+PID-XXX***s***.xf*
- Open the file *sampling_report_lu_C_64.txt*

-
  ```
  CrayPat/X:  Version 6.3.2 Revision rc1/6.3.2  02/25/16 18:26:21
  Number of PEs (MPI ranks):   64
  Numbers of PEs per Node:  32  PEs on each of  2  Nodes
  Numbers of Threads per PE:       1
  Number of Cores per Socket:   16
  Execution start time:  Wed Apr 13 16:57:06 2016
  System name and speed:  nid00035  2301 MHz (approx)
  Current path to data file:
   /scratch/markomg/NPB3.3.1/NPB3.3-MPI/bin/lu.C.64+pat+10974-35s.ap2
  (RTS)
  ```

Create your first report with sampling instrumentation

- Table 1: Profile by Function

```
 Samp% |   Samp |  Imb.    |  Imb.     |Group
       |        |  Samp | Samp% | Function
       |           |         |        |  PE=HIDE

 100.0% | 1,039.1 |      -- |     -- |Total
|------------------------------------------------------------------
|  72.3% |   751.5 |       -- |     -- |USER
||-----------------------------------------------------------------
||  33.5% |   347.9 |  45.1 | 11.6% |rhs_
||   8.0% |   83.4 |  24.6 | 23.0% |blts_
||   7.9% |   82.1 |  18.9 | 18.9% |buts_
||   7.8% |   81.2 |  23.8 | 22.9% |jacld_
||   7.4% |   77.4 |  24.6 | 24.3% |jacu_
||   4.6% |   47.8 |  26.2 | 35.7% |exchange_3_
||   2.2% |   23.2 |  15.8 | 40.8% |ssor_
||==================================================================
```

Create your first report with sampling instrumentation (MPI with sampling is not helpful)

- Table 1: Profile by Function

```
Samp%   |   Samp |  Imb.    |  Imb.     |Group
        |        |  Samp | Samp% | Function
        |        |          |           |  PE=HIDE

 |  18.6% |   192.9 | -- |      -- |MPI
||------------------------------------------------------------------
||   6.3% |      65.1 | 154.9 | 70.9% |MPIDI_Cray_shared_mem_coll_bcast
||   4.0% |      42.0 |  59.0 | 58.9% |MPIDI_CH3I_Progress
||   2.2% |      22.5 |  99.5 | 82.2% |MPIDI_Cray_shared_mem_coll_barrier
||   1.8% |      19.0 |  51.0 | 73.5% |MPID_nem_gni_poll
||   1.5% |      15.2 |  39.8 | 72.9% |MPID_nem_gni_check_localCQ
||==================================================================
|   4.5% |      47.0 | -- |      -- |GNI
||------------------------------------------------------------------
||   4.3% |      44.8 | 118.2 | 73.1% |GNI_CqGetEvent
||==================================================================
|   4.4% |      45.8 | -- |      -- |ETC
||------------------------------------------------------------------
||   2.1% |      21.9 |  65.1 | 75.4% |GNII_DlaProgress
||   1.0% |      10.4 |   8.6 | 45.8% |_cray_mpi_memcpy_snb
|===================================================================
```

Profile by Group, Function, and Line

- Table 2: Profile by Group, Function, and Line

```
Samp% |   Samp |  Imb.   |  Imb.     |Group
      |        |  Samp | Samp% | Function
      |        |       |       |  Source
      |        |       |       |   Line
      |        |       |       |     PE=HIDE

 100.0% | 1,039.1 |     -- |     -- |Total
|------------------------------------------------------------------
|  72.3% |   751.5 |      -- |     -- |USER
||-----------------------------------------------------------------
||  33.5% |   347.9 |      -- |     -- |rhs_
3|      |       |       |       | NPB3.3.1/NPB3.3-MPI/LU/rhs.f
||||--------------------------------------------------------------
4|||   2.2% |22.5 |  13.5 | 37.8% |line.43
4|||   1.8% |18.2 |   9.8 | 35.2% |line.96
4|||   1.6% |16.4 |  10.6 | 39.6% |line.228
…
```

File rhs.f, line 43

```
do k = 1, nz
      do j = 1, ny
        do i = 1, nx
          do m = 1, 5
            rsd(m,i,j,k) = -
          frct(m,i,j,k)
          end do
        end do
      end do
    end do
```

More information from sampling

- Table 3: Wall Clock Time, Memory High Water Mark (limited entries shown)

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

 20.455187 | 39.18 |Total
|------------------------------
| 23.922620 |39.74 |pe.34
| 19.638636 |39.57 |pe.107
| 16.558081 |39.66 |pe.68
|==============================
```

- ======================== Additional details ========================

```
   Experiment:  samp_pc_time

  Sampling interval:  10000 microsecs
```

Automatic Profiling Analysis (APA)

- After the previous execution of the command *pat_report* two new files were created with extensions *apa* and *ap2*, the second one will be presented later.

- Open the file sampling_report_lu_C_64.apa

```
#       Collect the default PERFCTR group.

  -Drtenv=PAT_RT_PERFCTR=default

#       Alternatively, energy counters may be added to the default
#       list by commenting out the line above and enabling the
#       line below. Note that this may significantly increase the
#       runtime overhead for high trace counts. The parentheses
#       in the syntax below denote counters that are not available
#       on all platforms.

#  -Drtenv=PAT_RT_PERFCTR=default,(PM_ENERGY:NODE),(PM_ENERGY:ACC)

#       Libraries to trace.

  -g mpi
```

Automatic Profiling Analysis (APA) II

```
#         Local functions are listed for completeness, but cannot be traced.

  -w  # Enable tracing of user-defined functions.

# 33.49% 32799 bytes
          -T rhs_

# 8.02% 3379 bytes
          -T blts_

# 7.90% 3863 bytes
          -T buts_

# 7.81% 14983 bytes
          -T jacld_
…
-o lu.C.128+apa # New instrumented program.
```

Automatic Profiling Analysis (APA) III

- In order to create the new binary with regard to APA, execute the following
  - ***pat_build -O sampling_report_lu_C_64.apa***

```
WARNING: Tracing small, frequently called functions can add excessive overhead.
WARNING: To set a minimum size, say 1200 bytes, for traced functions, use:
    -D trace-text-size=1200.
INFO: A total of 7 selected non-group functions were traced.
INFO: A maximum of 105 functions from group 'mpi' will be traced.
```

- The new instrumented binary is called *lu.C.64+apa*

- Edit the submit.sh file, comment line 16 and uncomment line 19
  - *sbatch --reservation=s1001_85 submit.sh*

- The new performance file is called *lu.C.64+**apa**+PID-XXX**t**.xf*

- Use the tool *pat_report*
  - *pat_report -o report_apa_lu_C_64.txt lu.C.64+apa+PID-XXXt.xf*

- Open the file report_apa_lu_C_64.txt

Performance report I

```
Table 1:  Profile by Function Group and Function

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

 100.0% | 12.081612 |        -- |       -- | 455,387.7 |Total
|-----------------------------------------------------------------
|  73.8% |  8.922097 |        -- |       -- | 161,404.0 |USER
||----------------------------------------------------------------
||  28.6% |  3.450003 | 0.416838 |  10.9% |       253.0 |rhs_
||  10.4% |  1.260820 | 0.153597 |  10.9% |  40,160.0 |buts_
||  10.4% |  1.259256 | 0.144344 |  10.4% |  40,160.0 |blts_
||   7.5% |  0.909228 | 0.122412 |  12.0% |  40,160.0 |jacld_
||   7.1% |  0.861425 | 0.130527 |  13.3% |  40,160.0 |jacu_
||   5.7% |  0.684862 | 0.139784 |  17.1% |         2.0 |ssor_
||   3.7% |  0.451014 | 0.295409 |  39.9% |       508.0 |exchange_3_
||================================================================
```

Performance report II

```
Table 1:  Profile by Function Group and Function

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

|  17.8% |  2.148878 |        -- |      -- | 293,958.7 |MPI
||-----------------------------------------------------------------
||  11.9% |  1.432456 | 3.029769 |  68.4% | 145,580.0 |MPI_RECV
||   3.8% |  0.465076 | 0.411500 |  47.3% | 146,502.9 |MPI_SEND
||   2.0% |  0.241474 | 1.003594 |  81.2% |       922.9 |mpi_wait
||================================================================
|   8.4% |  1.010618 |         -- |      -- |      24.0 |MPI_SYNC
||-----------------------------------------------------------------
||   8.2% |  0.991427 | 0.991319 | 100.0% |       1.0 |mpi_init_(sync)
|=================================================================
```

- If needed disable MPI Sync with
  - *export PAT_RT_MPI_SYNC=0*

MPI topology

- MPI Grid Detection:

  There appears to be point-to-point MPI communication in a 8 X 16 grid pattern. The 17.8% of the total execution time spent in MPI functions might be reduced with a rank order that maximizes communication between ranks on the same node. The effect of several rank orders is estimated below.

  A file named MPICH_RANK_ORDER.Grid was generated along with this report and contains usage instructions and the Hilbert rank order from the following table.

Rank Order	On-Node Bytes/PE	On-Node Bytes/PE% of Total Bytes/PE	MPICH_RANK_REORDER_METHOD
Hilbert	3.039e+10	87.40%	3
SMP	2.947e+10	84.75%	1
Fold	1.685e+10	48.46%	2
RoundRobin	1.106e+10	31.82%	0

- Example for 128 MPI processes

0,1,17,16,32,48...
68,84,85,69,70,71…

How to use the new MPI topology file:

1. **cp MPICH_RANK_ORDER.XXX MPICH_RANK_ORDER**
2. **export MPICH_RANK_REORDER_METHOD=3**

Hardware counters

D1 cache utilization:

All instrumented functions with significant execution time had D1 cache hit ratios above the desirable minimum of 75.0%.

D1 + D2 cache utilization:

All instrumented functions with significant execution time had combined D1 and D2 cache hit ratios above the desirable minimum of 80.0%.

TLB utilization:

All instrumented functions with significant execution time had more than the desirable minimum of 200 data references per TLB miss.

Find more about hardware performance counters

- ❖ Execute:
  - *pat_help*
  - *counters haswell groups*

Hardware counters

```
Total
------------------------------------------------------------------------------
  Time%                                                 100.0%
  Time                                        12.081612 secs
  Imb. Time                                             -- secs
  Imb. Time%                                            --
  Calls                         0.038M/sec                        455,387.7 calls
  CPU_CLK_THREAD_UNHALTED:THREAD_P                                 47,351,574,846
  CPU_CLK_THREAD_UNHALTED:REF_XCLK                                 2,124,810,371
  DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK                                       6,686,929
  DTLB_STORE_MISSES:MISS_CAUSES_A_WALK                                      2,823,391
  L1D:REPLACEMENT                                                    1,404,754,113
  L2_RQSTS:ALL_DEMAND_DATA_RD                                        515,418,048
  L2_RQSTS:DEMAND_DATA_RD_HIT                                        197,719,491
  MEM_UOPS_RETIRED:ALL_LOADS                                      20,512,449,601
  CPU_CLK                       2.23GHz
  TLB utilization               2,156.86 refs/miss              4.21 avg uses
  D1 cache hit,miss ratios              93.2% hits              6.8% misses
  D1 cache utilization (misses)         14.60 refs/miss         1.83 avg hits
  D2 cache hit,miss ratio       77.4% hits              22.6% misses
  D1+D2 cache hit,miss ratio            98.5% hits              1.5% misses
  D1+D2 cache utilization               64.57 refs/miss         8.07 avg hits
  D2 to D1 bandwidth            2,603.843MiB/sec  32,986,755,044 bytes
  Average Time per Call                                 0.000027 secs
  CrayPat Overhead : Time               8.0%
```

Hardware Counters - Description

Hardware performance counter events:

CPU_CLK_THREAD_UNHALTED:REF_XCLK Count core clock cycles whenever the clock signal on the specificcore is running (not halted):Cases when the core is unhalted at 100Mhz

CPU_CLK_THREAD_UNHALTED:THREAD_P Count core clock cycles whenever the clock signal on the specificcore is running (not halted):Cycles when thread is not halted

DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK Data TLB load misses:Misses in all DTLB levels that cause page walks

DTLB_STORE_MISSES:MISS_CAUSES_A_WALK Data TLB store misses:Misses in all DTLB levels that cause page walks

L1D:REPLACEMENT L1D cache:L1D Data line replacements

L2_RQSTS:ALL_DEMAND_DATA_RD L2 requests:Any data read request to L2 cache

L2_RQSTS:DEMAND_DATA_RD_HIT L2 requests:Demand Data Read requests that hit L2 cache

MEM_UOPS_RETIRED:ALL_LOADS Memory uops retired (Precise Event):All load uops retired
PM_ENERGY:NODE Compute node accumulated energy
CYCLES_RTC User Cycles (approx, from rtc)

Load Balance with MPI Message stats

Table 3: Load Balance with MPI Message Stats (limited entries shown)

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

 100.0% | 12.081612 | 146,522.9 | 271,667,585.0 | 1,854.10 |Total
|-----------------------------------------------------------------
|  73.8% |  8.922097 |       0.0 |           0.0 |       -- |USER
||----------------------------------------------------------------
||  80.6% |  9.739499 |       0.0 |           0.0 |       -- |pe.26
||  75.8% |  9.160217 |       0.0 |           0.0 |       -- |pe.61
||  45.1% |  5.442844 |       0.0 |           0.0 |       -- |pe.127
||================================================================
|  17.8% |  2.148878 | 146,522.9 | 271,667,585.0 | 1,854.10 |MPI
||----------------------------------------------------------------
||  48.8% |  5.891394 |  80,852.0 | 143,737,828.0 | 1,777.79 |pe.127
||  15.5% |  1.874838 | 161,678.0 | 293,895,236.0 | 1,817.78 |pe.43
||  10.5% |  1.263484 | 161,678.0 | 303,691,732.0 | 1,878.37 |pe.26
||================================================================
|   8.4% |  1.010618 |       0.0 |           0.0 |       -- |MPI_SYNC
||----------------------------------------------------------------
||  22.0% |  2.653814 |       0.0 |           0.0 |       -- |pe.103
||   7.4% |  0.895974 |       0.0 |           0.0 |       -- |pe.123
||   0.1% |  0.012597 |       0.0 |           0.0 |       -- |pe.0
```

Load Balance with MPI message stats by caller

```
Table 4:  MPI Message Stats by Caller (limited entries shown)

        MPI | MPI Msg Bytes |   MPI Msg | MsgSz |  16<=  | 256<= | 64KiB<= |Function
        Msg |               |    Count  |   <16  | MsgSz | MsgSz |   MsgSz | Caller
 Bytes%    |                |           | Count   |  <256 | <4KiB |   <1MiB |PE=[mmm]
           |                |           |         | Count |    Count |   Count |
 100.0% | 271,667,585.0     | 146,522.9 |  14.0   |   6.9   | 145,581.3 |   920.8 |Total
|-----------------------------------------------------------------------------
| 100.0% | 271,667,261.0 | 146,502.9 |   0.0 |   0.9 | 145,581.3 |   920.8 |MPI_SEND
||----------------------------------------------------------------------------
||  67.5% | 183,314,340.0 |      920.8 |   0.0 |   0.0 |       0.0 |   920.8 |exchange_3_
3|  67.2% | 182,592,630.0 |      917.1 |   0.0 |   0.0 |       0.0 |   917.1 | rhs_
4|       |               |       |       |       |       |       |  ssor_
5|       |               |       |       |       |       |       |   applu_
||||||-------------------------------------------------------------------------
6|||||  77.2% | 209,848,320.0 |   1,012.0 |   0.0 |   0.0 |     0.0 | 1,012.0 |pe.17
6|||||  72.4% | 196,732,800.0 |   1,012.0 |   0.0 |   0.0 |     0.0 | 1,012.0 |pe.88
6|||||  36.2% |  98,366,400.0 |   506.0 |   0.0 |   0.0 |       0.0 |   506.0 |pe.127
```

❖ In order to adjust the size of the MPI eager mode (default 8KB, max value 128KB) according to the MPI message stats, use the following command in your job script, where

- export MPICH_GNI_MAX_EAGER_MSG_SIZE=131072
- export MPICH_ENV_DISPLAY=1

Wall clock and memory high water mark

```
Table 5:  Wall Clock Time, Memory High Water Mark (limited entries shown)

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

 20.166938 |        48.25 |Total
|------------------------------
| 23.813279 |       48.70 |pe.98
| 20.039177 |       49.79 |pe.82
| 17.694283 |       49.70 |pe.0
|==============================
```

- In order to extract the profling information for all the processes and not aggregate data, the pat_report tool can be used as following:
  - *pat_report -s pe=ALL -o sampling_results_all.txt txt lu.C.64+apa+PID-XXXt.xf*
  - *pat_report -s filter_input='pe<=5' ...*
  - *pat_report -s filter_input='pe%2==0' ...*

Apprentice2

A GUI for the raw data

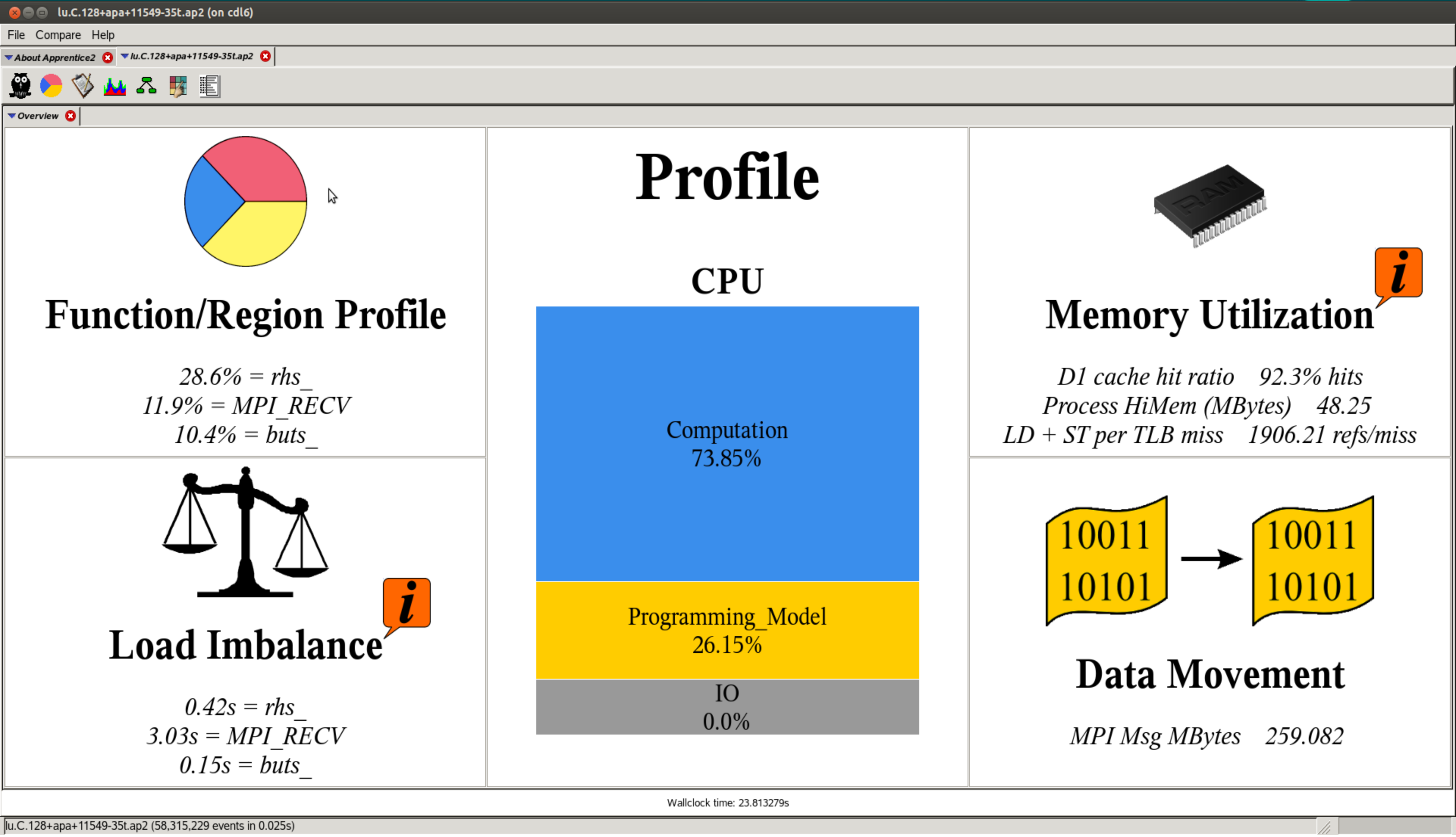
How to start with Apprentice2

- The *pat_report* tool has created one file with extension *ap2*
  - ls –ltr *.ap2

- In order to visualize the performance data
  - Connect to Shaheen II with “ssh –X …”
  - module load perftools-base/6.3.2
  - app2 lu.C.64+**apa**+PID-XX**t**.ap2

- The example of the presentation is for lu.C.128

Apprentice2 – Generic view



lu.C.128+apa+11549-35t.ap2 (on cdl6)

Function/Region Profile

*28.6% = rhs_*
11.9% = MPI_RECV
*10.4% = buts_*

Load Imbalance

*0.42s = rhs_*
3.03s = MPI_RECV
*0.15s = buts_*

Profile

CPU

CPU	
Computation	73.85%
Programming_Model	26.15%
IO	0.0%

Memory Utilization

D1 cache hit ratio	*92.3% hits*
Process HiMem (MBytes)	*48.25*
LD + ST per TLB miss	*1906.21 refs/miss*

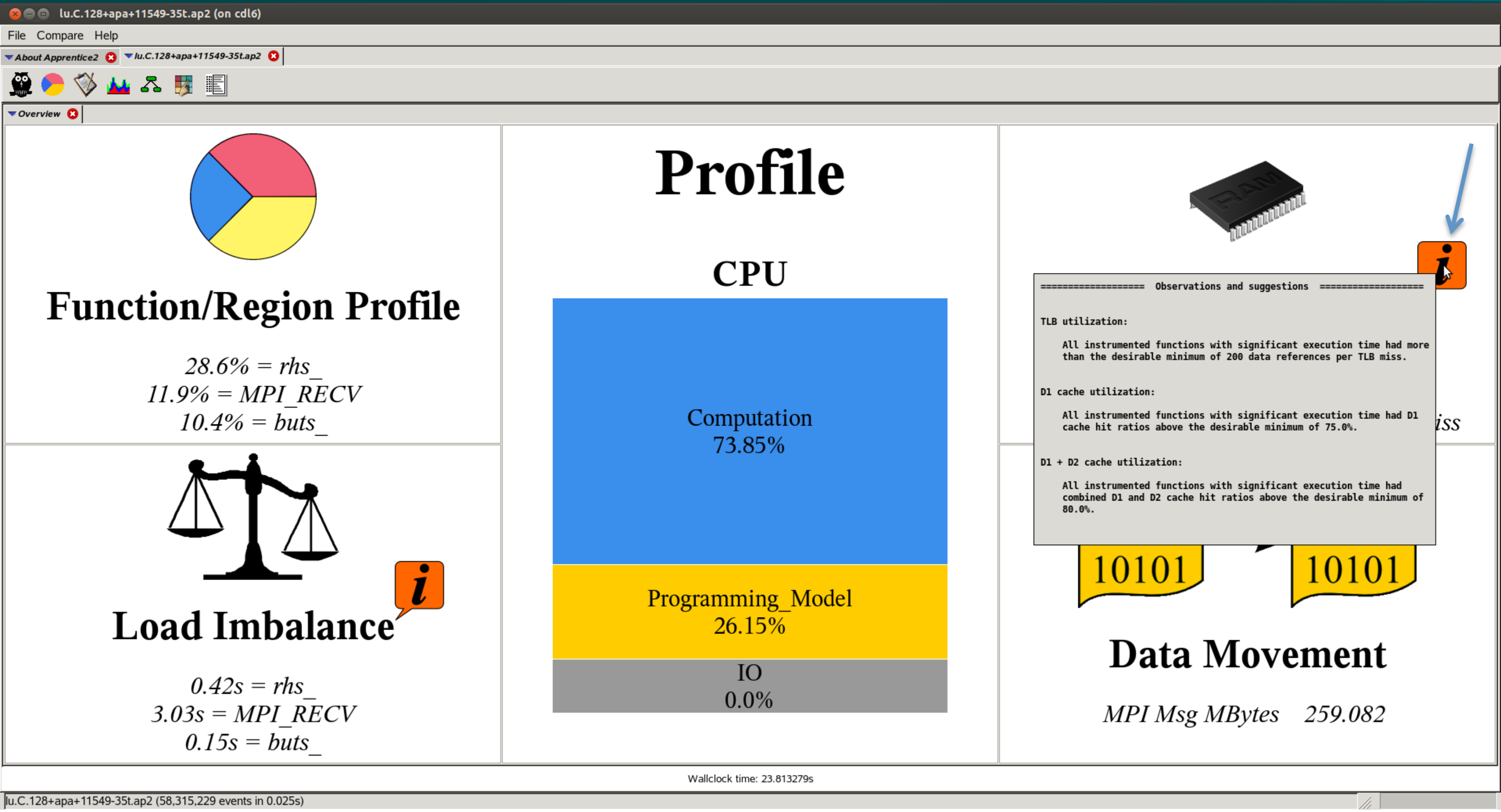
Data Movement

MPI Msg MBytes *259.082*

Wallclock time: 23.813279s

lu.C.128+apa+11549-35t.ap2 (58,315,229 events in 0.025s)

Apprentice2 – Generic view


lu.C.128+apa+11549-35t.ap2 (on cdl6)
File
Compare
Help
About Apprentice2
lu.C.128+apa+11549-35t.ap2
Overview
RAM
10101
10101

Function/Region Profile

*28.6% = rhs_*
11.9% = MPI_RECV
*10.4% = buts_*

Load Imbalance

*0.42s = rhs_*
3.03s = MPI_RECV
*0.15s = buts_*

Profile

CPU

Computation 73.85%
Programming_Model 26.15%
IO 0.0%

```
===================  Observations and suggestions  ====================

TLB utilization:

    All instrumented functions with significant execution time had more
    than the desirable minimum of 200 data references per TLB miss.


D1 cache utilization:

    All instrumented functions with significant execution time had D1
    cache hit ratios above the desirable minimum of 75.0%.


D1 + D2 cache utilization:

    All instrumented functions with significant execution time had
    combined D1 and D2 cache hit ratios above the desirable minimum of
    80.0%.
```

iss

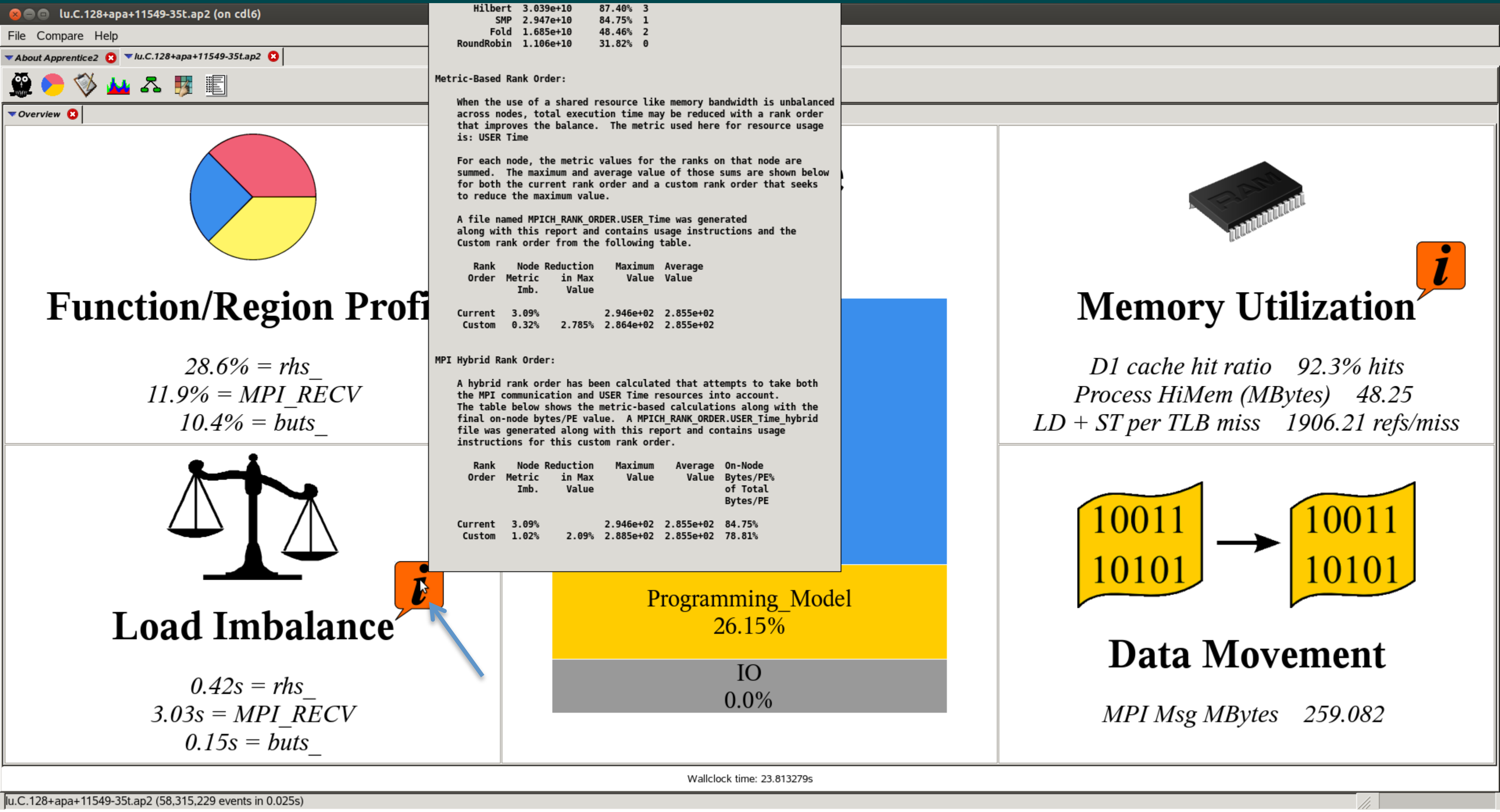
Data Movement

MPI Msg MBytes *259.082*

Wallclock time: 23.813279s

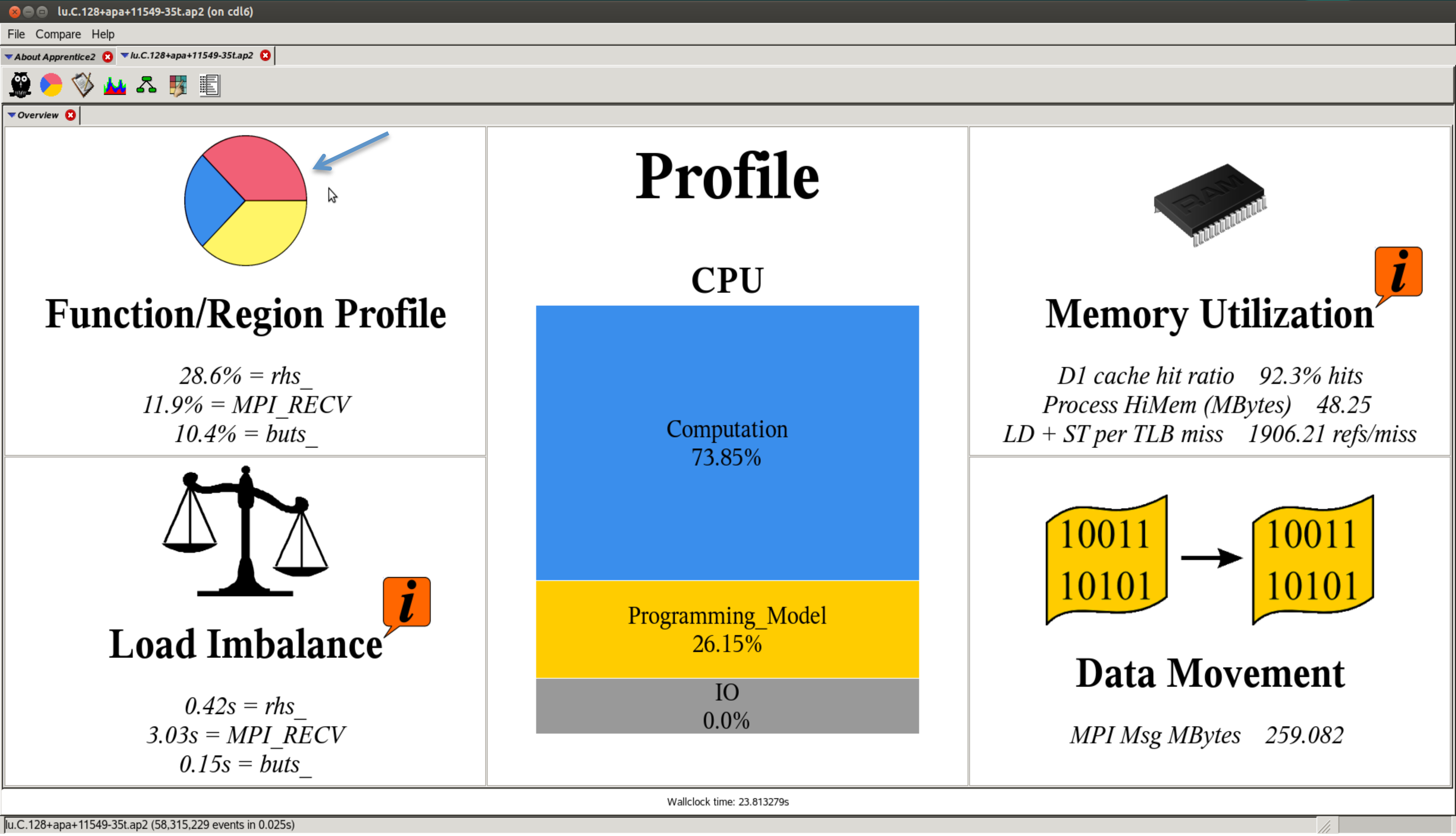
lu.C.128+apa+11549-35t.ap2 (58,315,229 events in 0.025s)

Apprentice2 – Generic view



```
      Hilbert  3.039e+10     87.40%  3
          SMP  2.947e+10     84.75%  1
         Fold  1.685e+10     48.46%  2
   RoundRobin  1.106e+10     31.82%  0


Metric-Based Rank Order:

    When the use of a shared resource like memory bandwidth is unbalanced
    across nodes, total execution time may be reduced with a rank order
    that improves the balance.  The metric used here for resource usage
    is: USER Time

    For each node, the metric values for the ranks on that node are
    summed.  The maximum and average value of those sums are shown below
    for both the current rank order and a custom rank order that seeks
    to reduce the maximum value.

    A file named MPICH_RANK_ORDER.USER_Time was generated
    along with this report and contains usage instructions and the
    Custom rank order from the following table.

       Rank    Node Reduction    Maximum  Average
      Order  Metric    in Max      Value  Value
               Imb.     Value

    Current   3.09%             2.946e+02  2.855e+02
     Custom   0.32%    2.785%  2.864e+02  2.855e+02


MPI Hybrid Rank Order:

    A hybrid rank order has been calculated that attempts to take both
    the MPI communication and USER Time resources into account.
    The table below shows the metric-based calculations along with the
    final on-node bytes/PE value.  A MPICH_RANK_ORDER.USER_Time_hybrid
    file was generated along with this report and contains usage
    instructions for this custom rank order.

       Rank    Node Reduction    Maximum    Average  On-Node
      Order  Metric    in Max      Value      Value  Bytes/PE%
               Imb.     Value                        of Total
                                                     Bytes/PE

    Current   3.09%             2.946e+02  2.855e+02  84.75%
     Custom   1.02%     2.09%  2.885e+02  2.855e+02  78.81%
```

Apprentice2 – Generic view



lu.C.128+apa+11549-35t.ap2 (on cdl6)
File Compare Help
About Apprentice2
lu.C.128+apa+11549-35t.ap2
Overview

Function/Region Profile

*28.6% = rhs_*
11.9% = MPI_RECV
*10.4% = buts_*

Load Imbalance

*0.42s = rhs_*
3.03s = MPI_RECV
*0.15s = buts_*

Profile

CPU

Memory Utilization

D1 cache hit ratio 92.3% hits
Process HiMem (MBytes) 48.25
LD + ST per TLB miss 1906.21 refs/miss

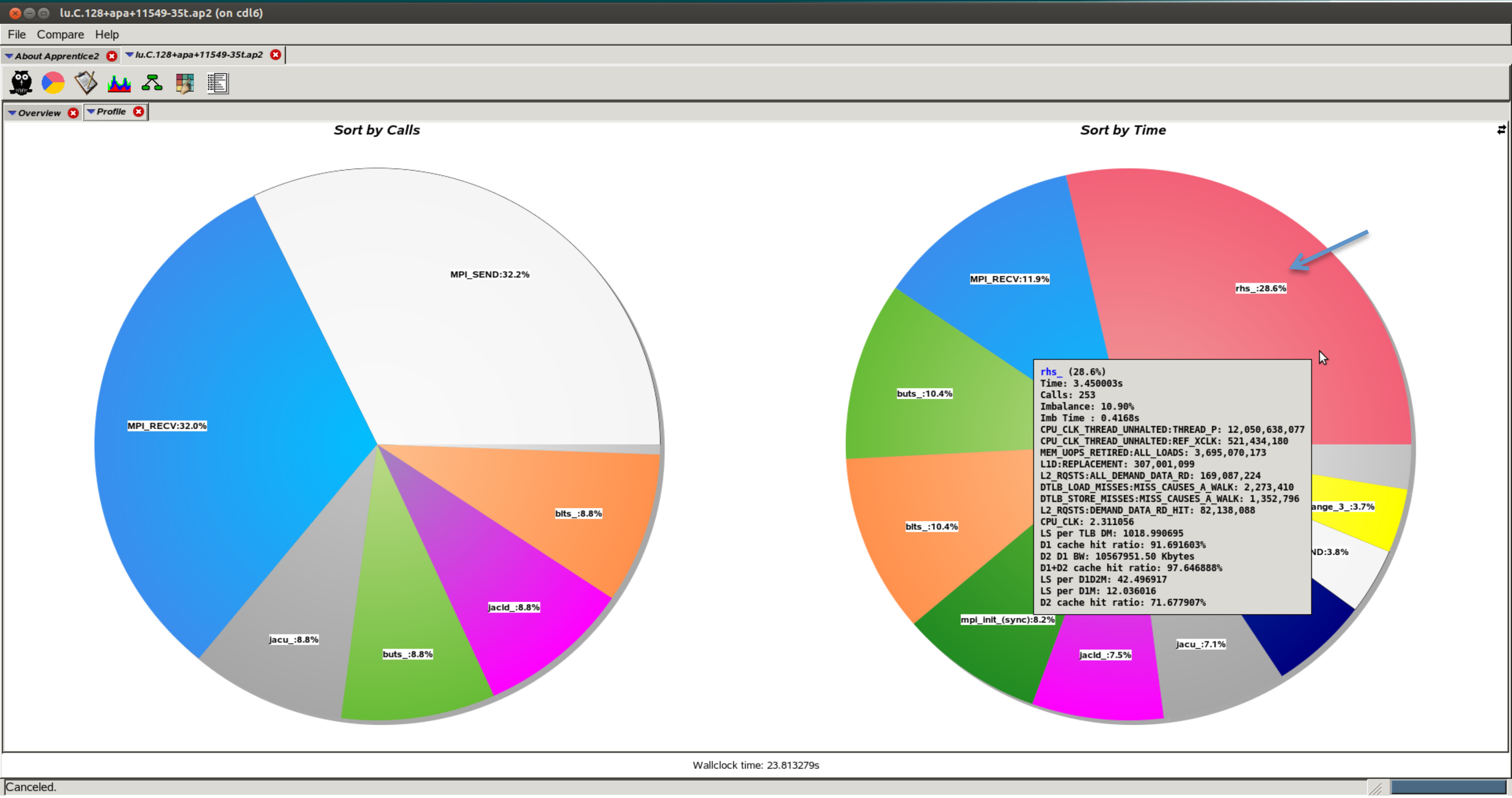
Data Movement

MPI Msg MBytes 259.082

Wallclock time: 23.813279s

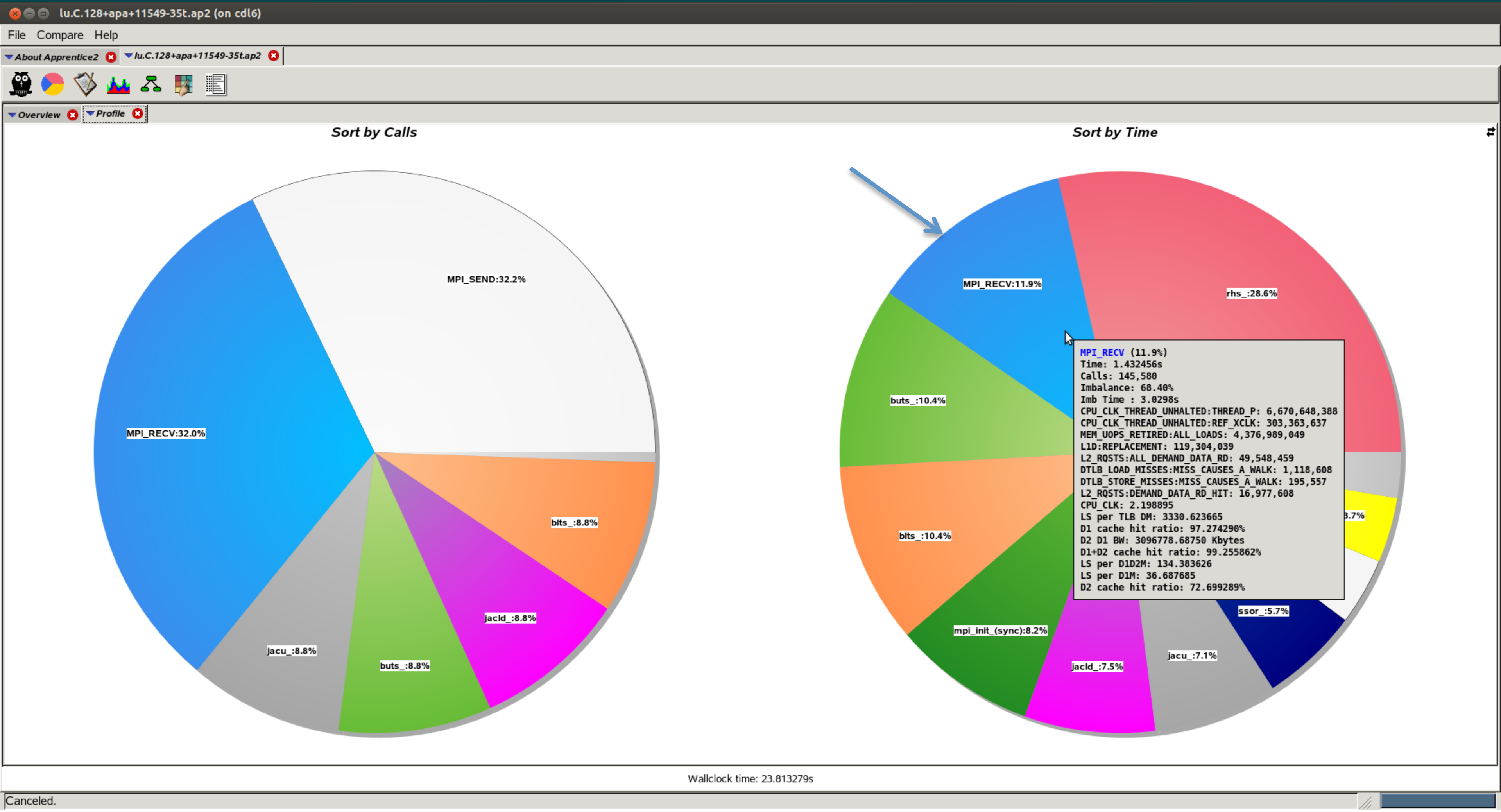
lu.C.128+apa+11549-35t.ap2 (58,315,229 events in 0.025s)

Apprentice2 – Profile I



```
rhs_ (28.6%)
Time: 3.450003s
Calls: 253
Imbalance: 10.90%
Imb Time : 0.4168s
CPU_CLK_THREAD_UNHALTED:THREAD_P: 12,050,638,077
CPU_CLK_THREAD_UNHALTED:REF_XCLK: 521,434,180
MEM_UOPS_RETIRED:ALL_LOADS: 3,695,070,173
L1D:REPLACEMENT: 307,001,099
L2_RQSTS:ALL_DEMAND_DATA_RD: 169,087,224
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK: 2,273,410
DTLB_STORE_MISSES:MISS_CAUSES_A_WALK: 1,352,796
L2_RQSTS:DEMAND_DATA_RD_HIT: 82,138,088
CPU_CLK: 2.311056
LS per TLB DM: 1018.990695
D1 cache hit ratio: 91.691603%
D2 D1 BW: 10567951.50 Kbytes
D1+D2 cache hit ratio: 97.646888%
LS per D1D2M: 42.496917
LS per D1M: 12.036016
D2 cache hit ratio: 71.677907%
```

Apprentice2 – Profile II

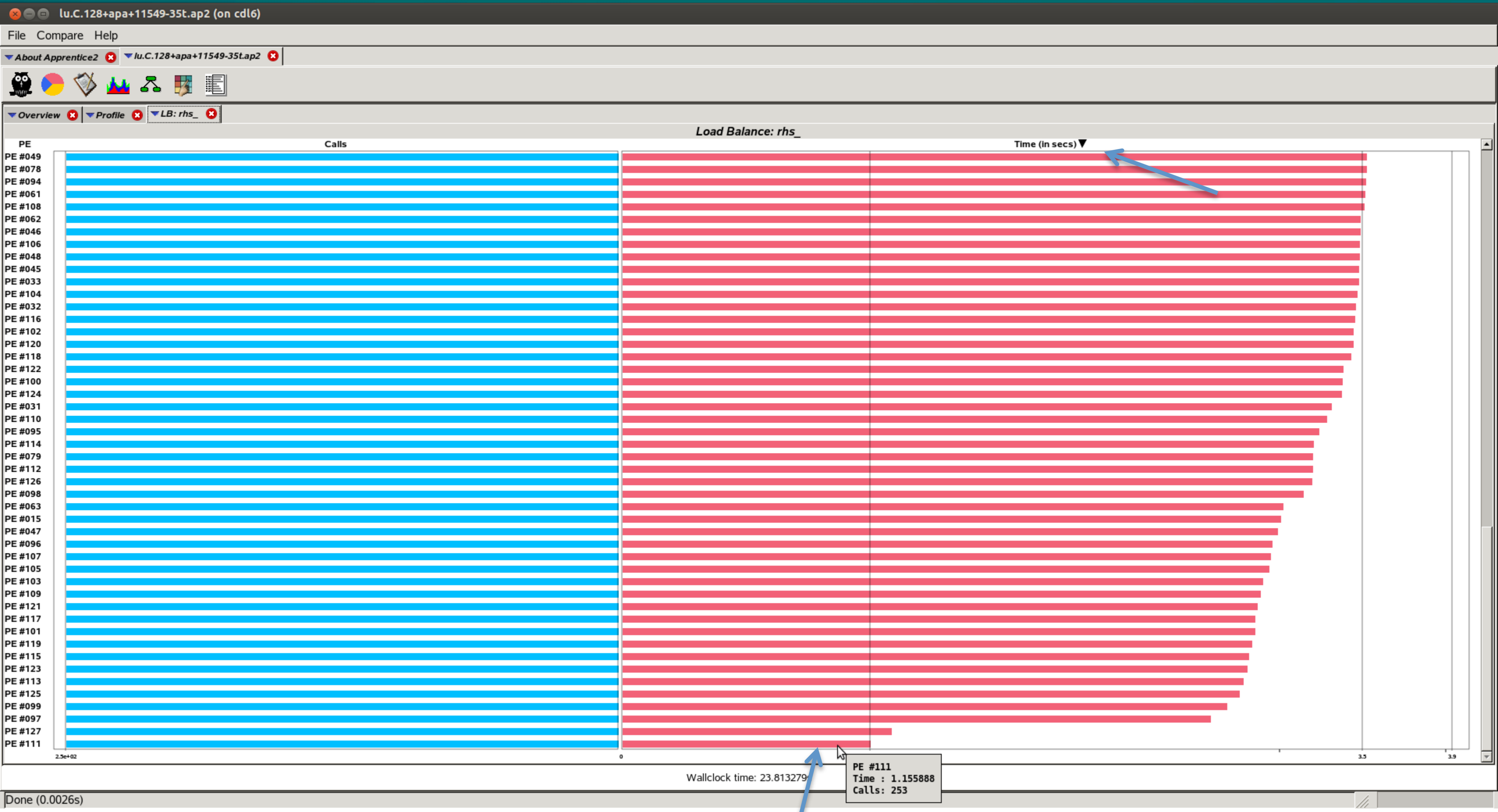


```
MPI_RECV (11.9%)
Time: 1.432456s
Calls: 145,580
Imbalance: 68.40%
Imb Time : 3.0298s
CPU_CLK_THREAD_UNHALTED:THREAD_P: 6,670,648,388
CPU_CLK_THREAD_UNHALTED:REF_XCLK: 303,363,637
MEM_UOPS_RETIRED:ALL_LOADS: 4,376,989,049
L1D:REPLACEMENT: 119,304,039
L2_RQSTS:ALL_DEMAND_DATA_RD: 49,548,459
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK: 1,118,608
DTLB_STORE_MISSES:MISS_CAUSES_A_WALK: 195,557
L2_RQSTS:DEMAND_DATA_RD_HIT: 16,977,608
CPU_CLK: 2.198895
LS per TLB DM: 3330.623665
D1 cache hit ratio: 97.274290%
D2 D1 BW: 3096778.68750 Kbytes
D1+D2 cache hit ratio: 99.255862%
LS per D1D2M: 134.383626
LS per D1M: 36.687685
D2 cache hit ratio: 72.699289%
```

Wallclock time: 23.813279s

Canceled.

Apprentice2 – Load Balance I

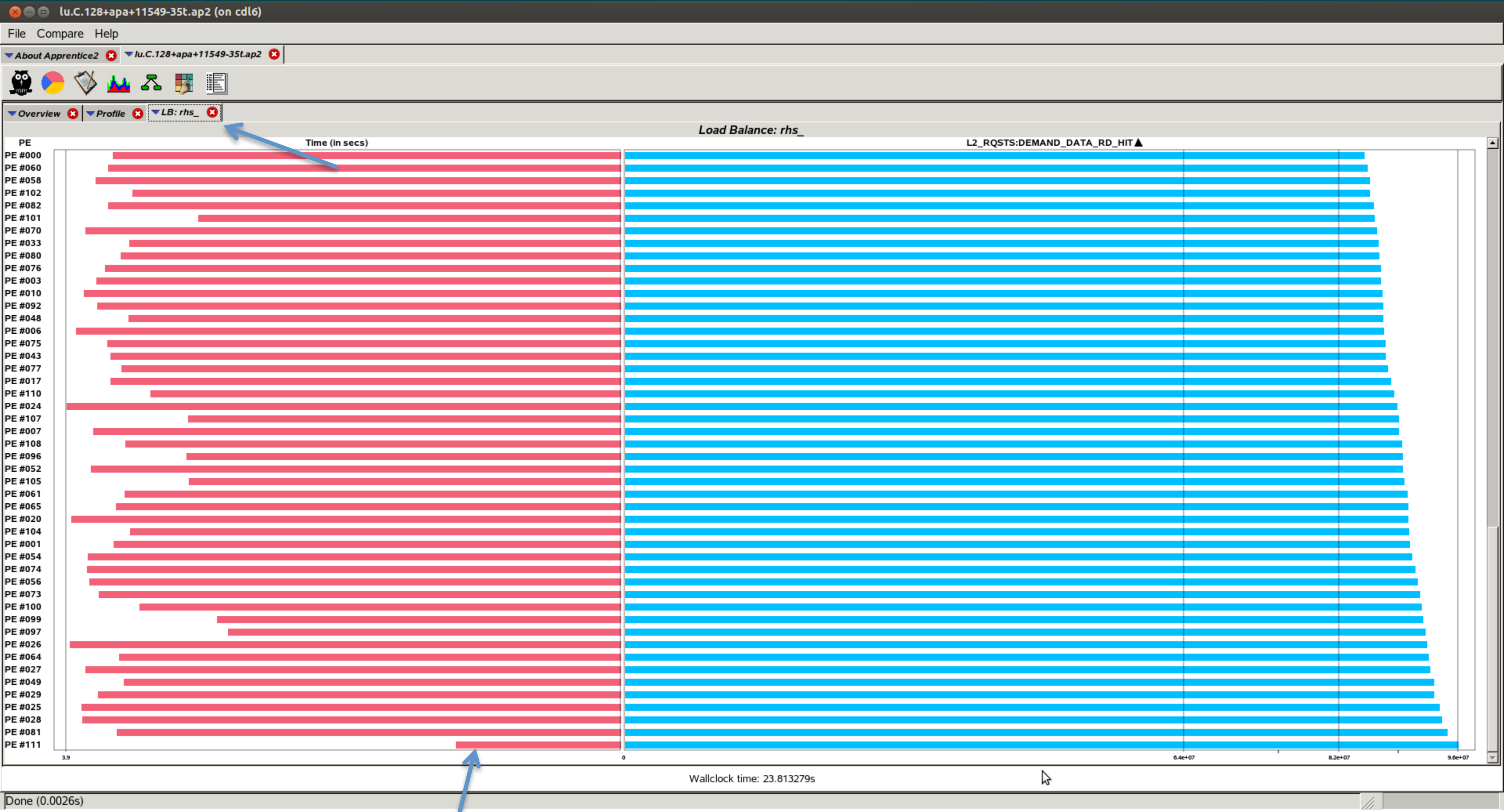


```
PE #111
Time : 1.155888
Calls: 253
```

Wallclock time: 23.813279

Done (0.0026s)

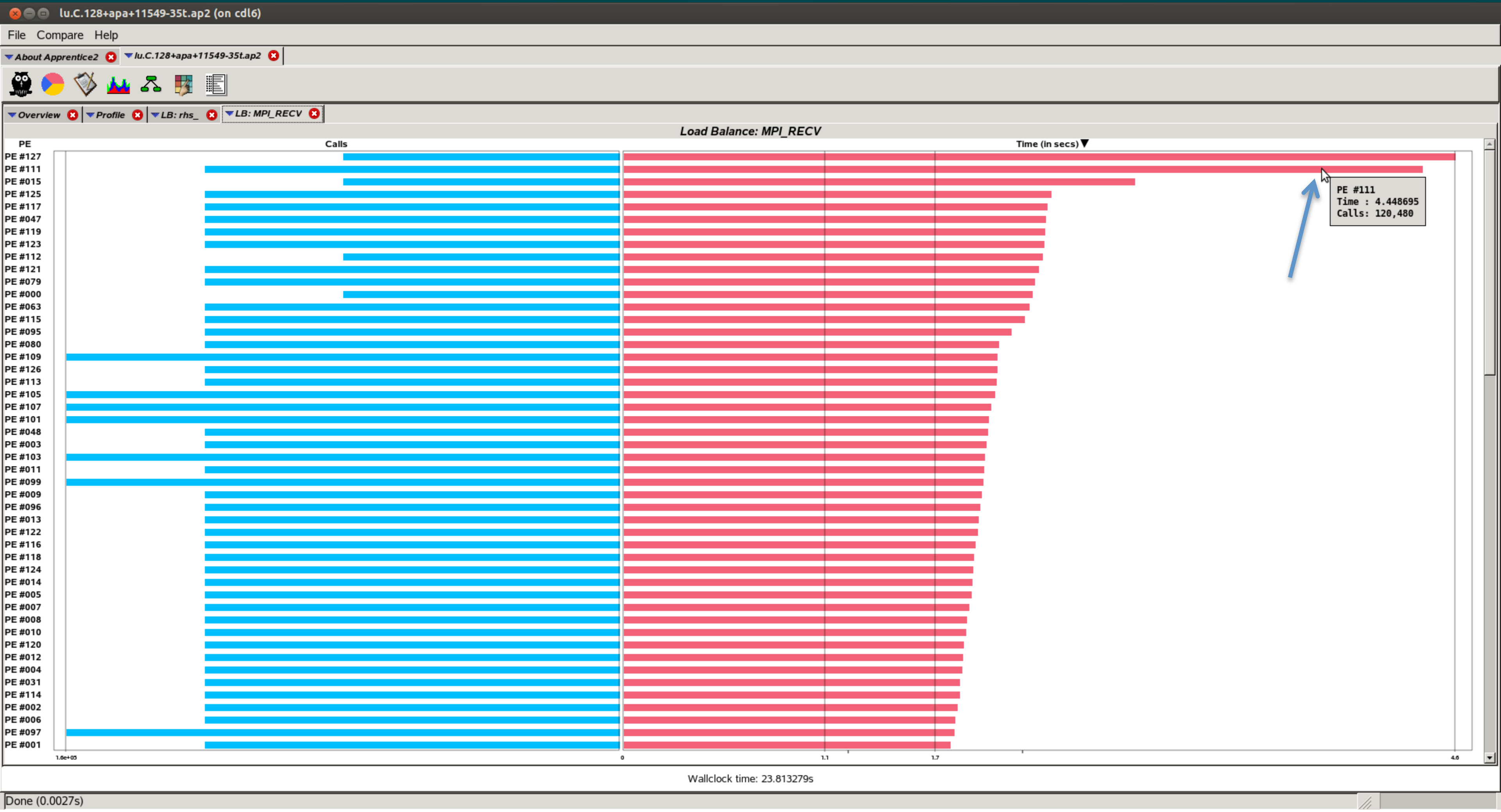
Apprentice2 – Load Balance II



Wallclock time: 23.813279s

Done (0.0026s)

Apprentice2 – Load Balance III

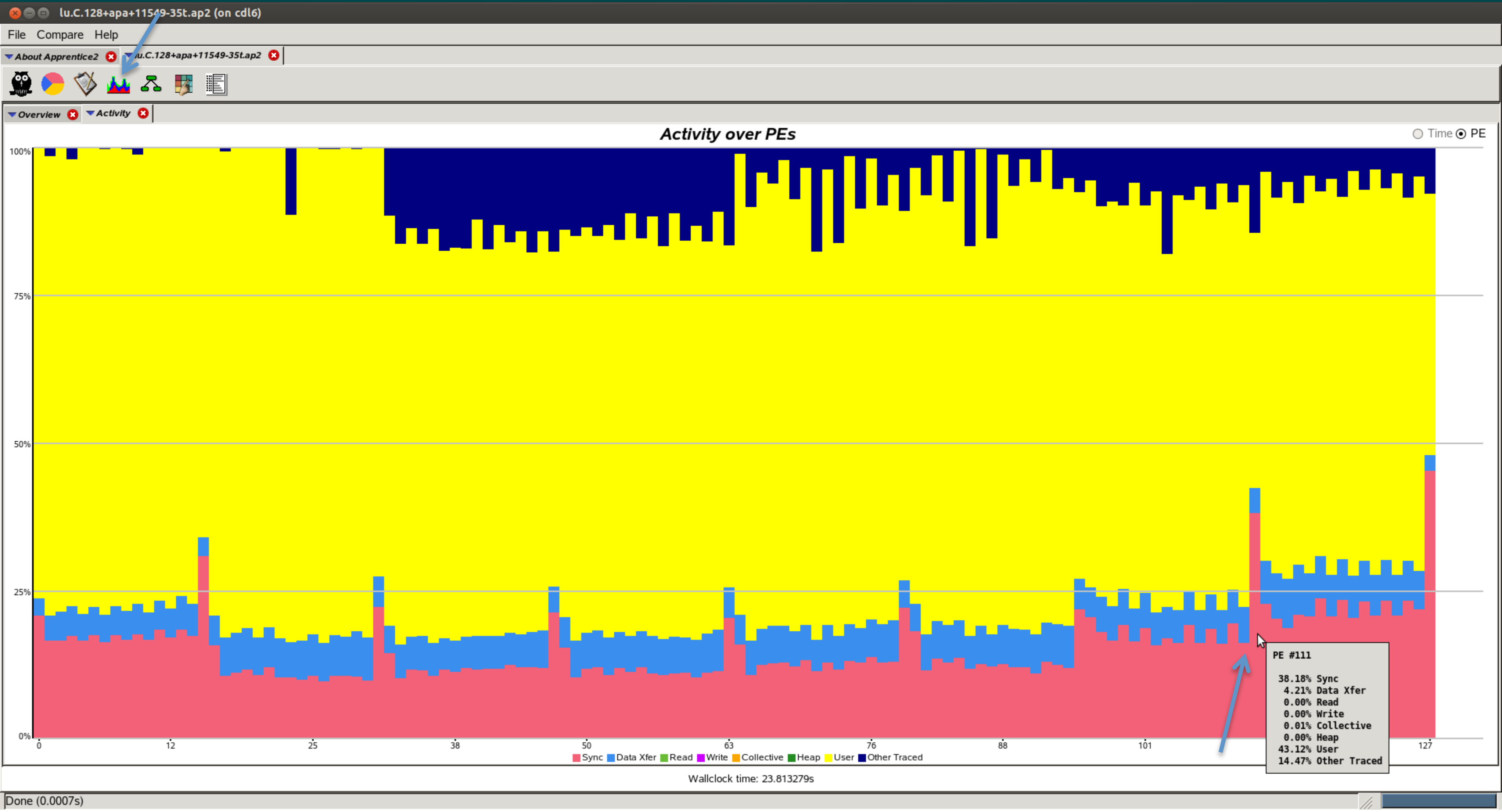


```
PE #111
Time : 4.448695
Calls: 120,480
```

Wallclock time: 23.813279s

Done (0.0027s)

Apprentice2 – Activity

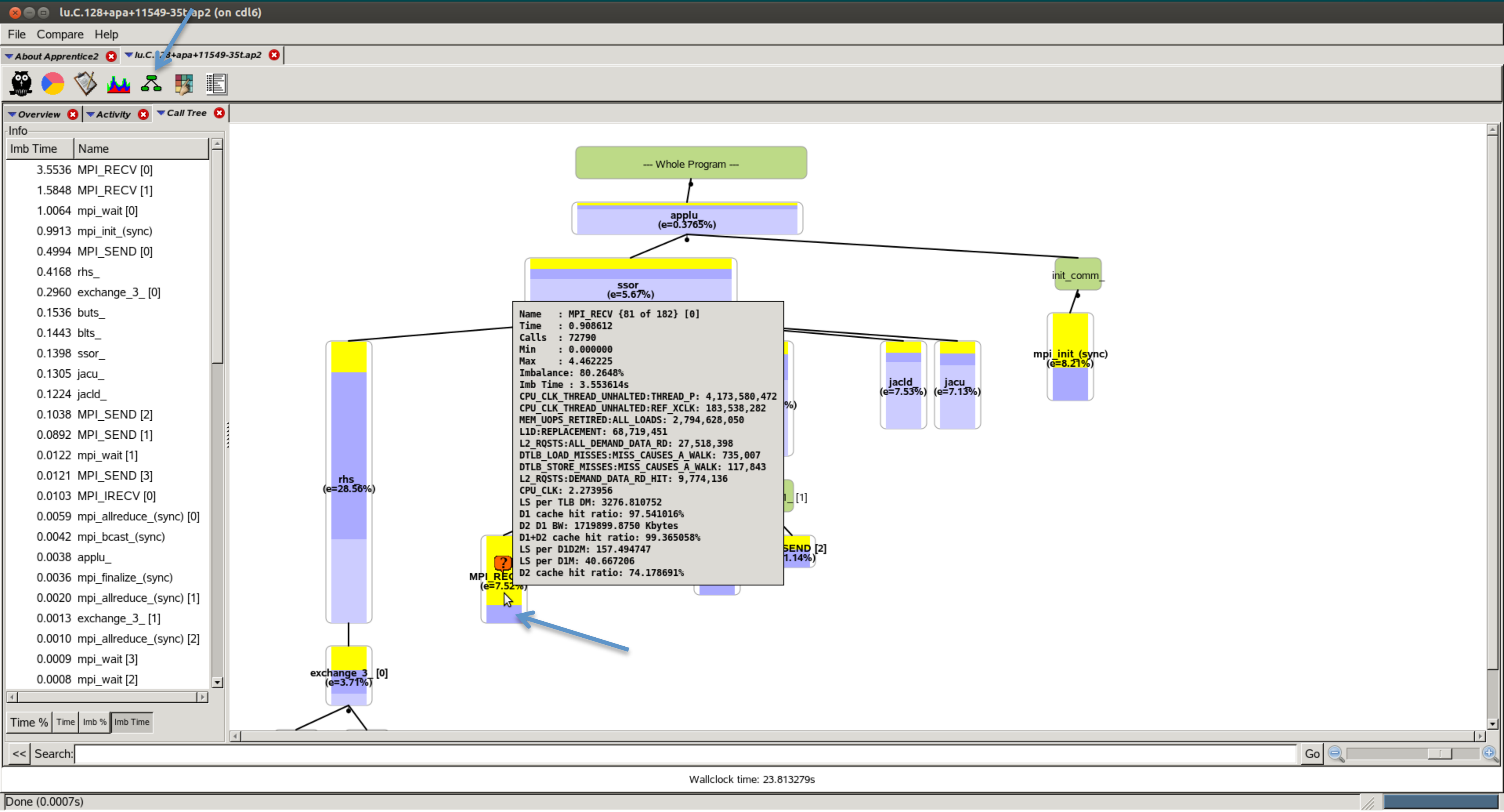


```
PE #111

 38.18% Sync
  4.21% Data Xfer
  0.00% Read
  0.00% Write
  0.01% Collective
  0.00% Heap
 43.12% User
 14.47% Other Traced
```

Wallclock time: 23.813279s

Done (0.0007s)

Apprentice2 – Call Tree

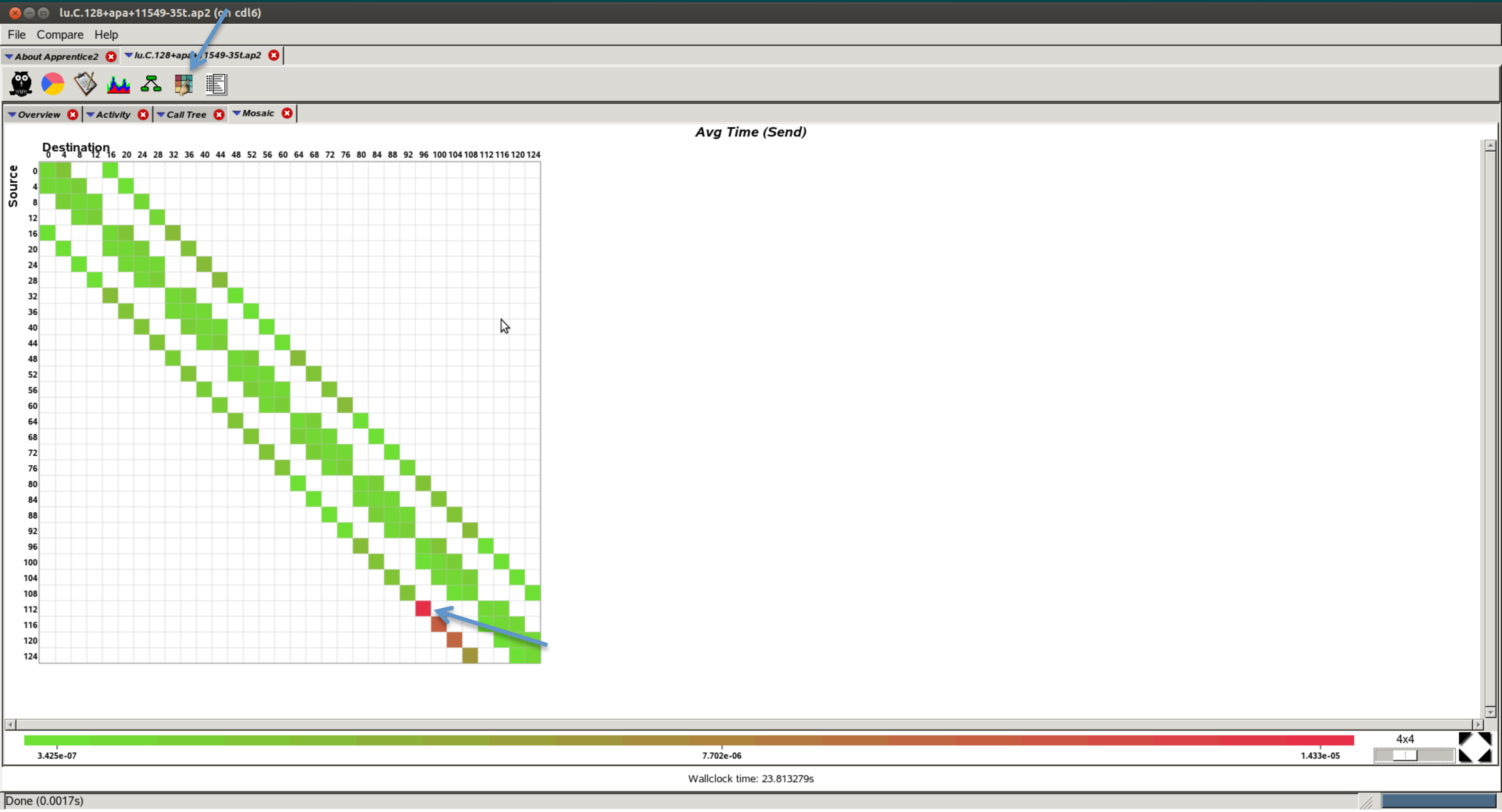


Imb Time	Name
3.5536	MPI_RECV [0]
1.5848	MPI_RECV [1]
1.0064	mpi_wait [0]
0.9913	mpi_init_(sync)
0.4994	MPI_SEND [0]
0.4168	rhs_
0.2960	exchange_3_ [0]
0.1536	buts_
0.1443	blts_
0.1398	ssor_
0.1305	jacu_
0.1224	jacld_
0.1038	MPI_SEND [2]
0.0892	MPI_SEND [1]
0.0122	mpi_wait [1]
0.0121	MPI_SEND [3]
0.0103	MPI_IRECV [0]
0.0059	mpi_allreduce_(sync) [0]
0.0042	mpi_bcast_(sync)
0.0038	applu_
0.0036	mpi_finalize_(sync)
0.0020	mpi_allreduce_(sync) [1]
0.0013	exchange_3_ [1]
0.0010	mpi_allreduce_(sync) [2]
0.0009	mpi_wait [3]
0.0008	mpi_wait [2]

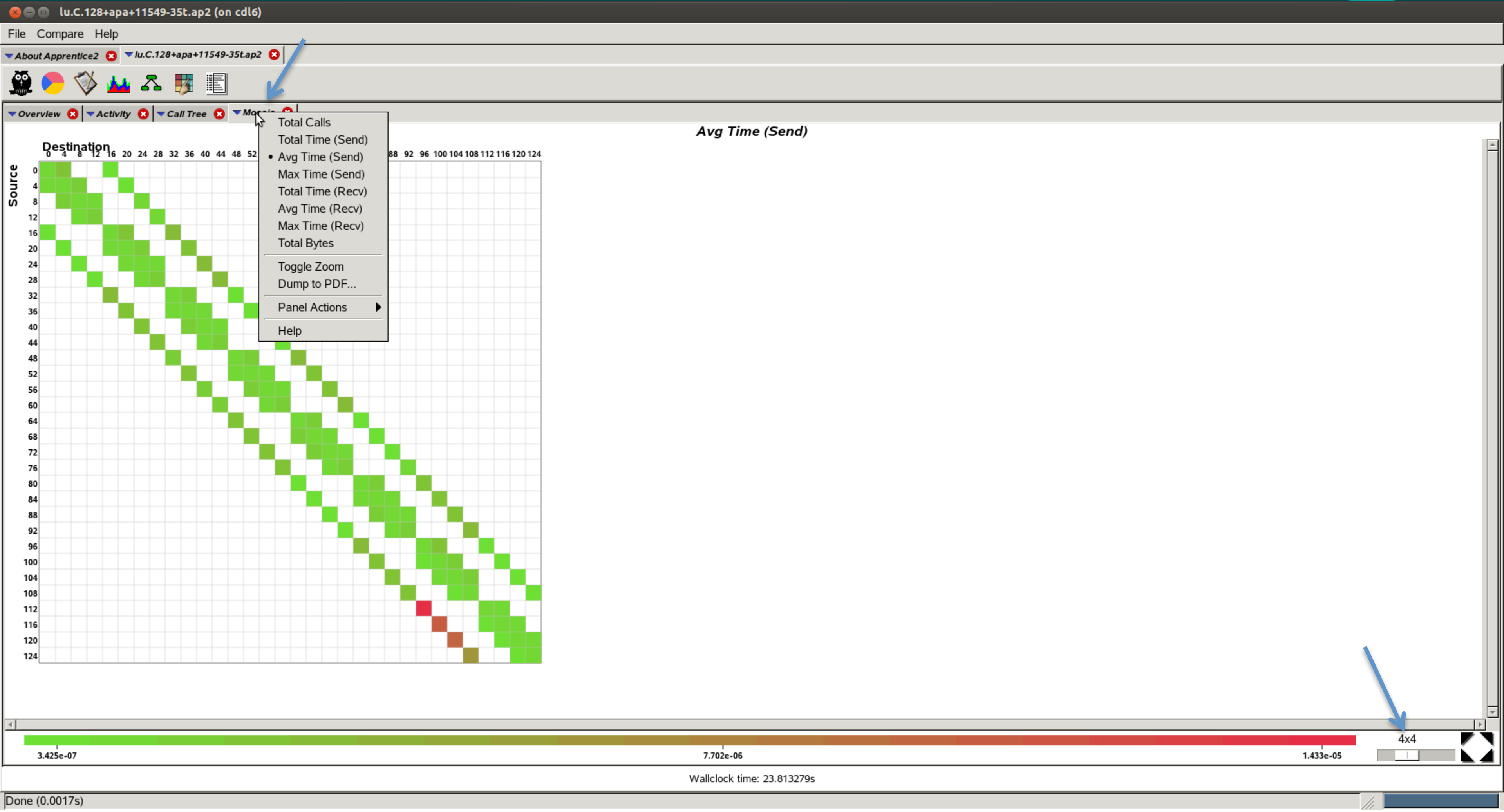
```
Name   : MPI_RECV {81 of 182} [0]
Time   : 0.908612
Calls  : 72790
Min    : 0.000000
Max    : 4.462225
Imbalance: 80.2648%
Imb Time : 3.553614s
CPU_CLK_THREAD_UNHALTED:THREAD_P: 4,173,580,472
CPU_CLK_THREAD_UNHALTED:REF_XCLK: 183,538,282
MEM_UOPS_RETIRED:ALL_LOADS: 2,794,628,050
L1D:REPLACEMENT: 68,719,451
L2_RQSTS:ALL_DEMAND_DATA_RD: 27,518,398
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK: 735,007
DTLB_STORE_MISSES:MISS_CAUSES_A_WALK: 117,843
L2_RQSTS:DEMAND_DATA_RD_HIT: 9,774,136
CPU_CLK: 2.273956
LS per TLB DM: 3276.810752
D1 cache hit ratio: 97.541016%
D2 D1 BW: 1719899.8750 Kbytes
D1+D2 cache hit ratio: 99.365058%
LS per D1D2M: 157.494747
LS per D1M: 40.667206
D2 cache hit ratio: 74.178691%
```

Wallclock time: 23.813279s

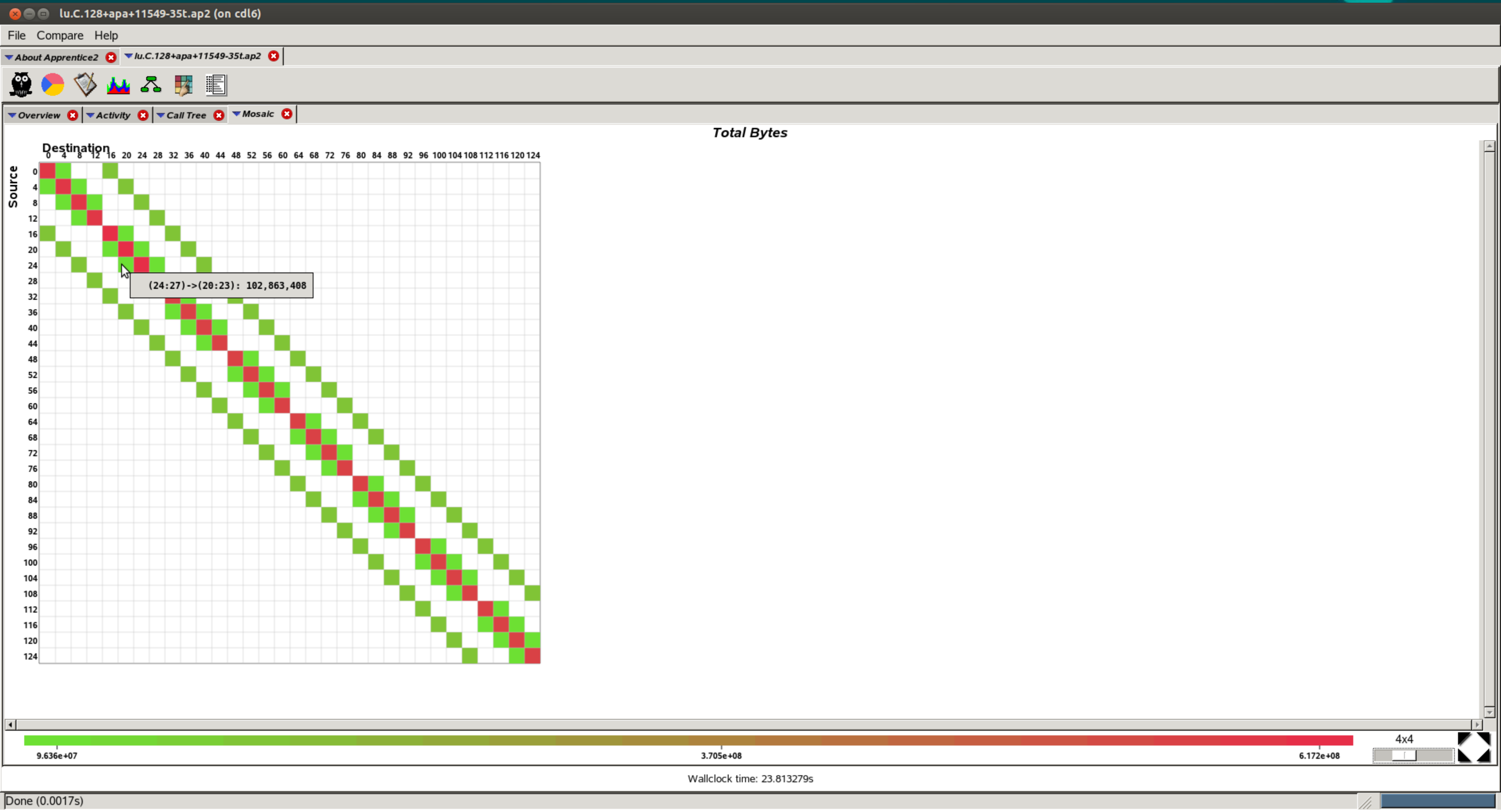
Apprentice2 – Mosaic I



Apprentice2 – Mosaic II



Apprentice2 – Mosaic IV

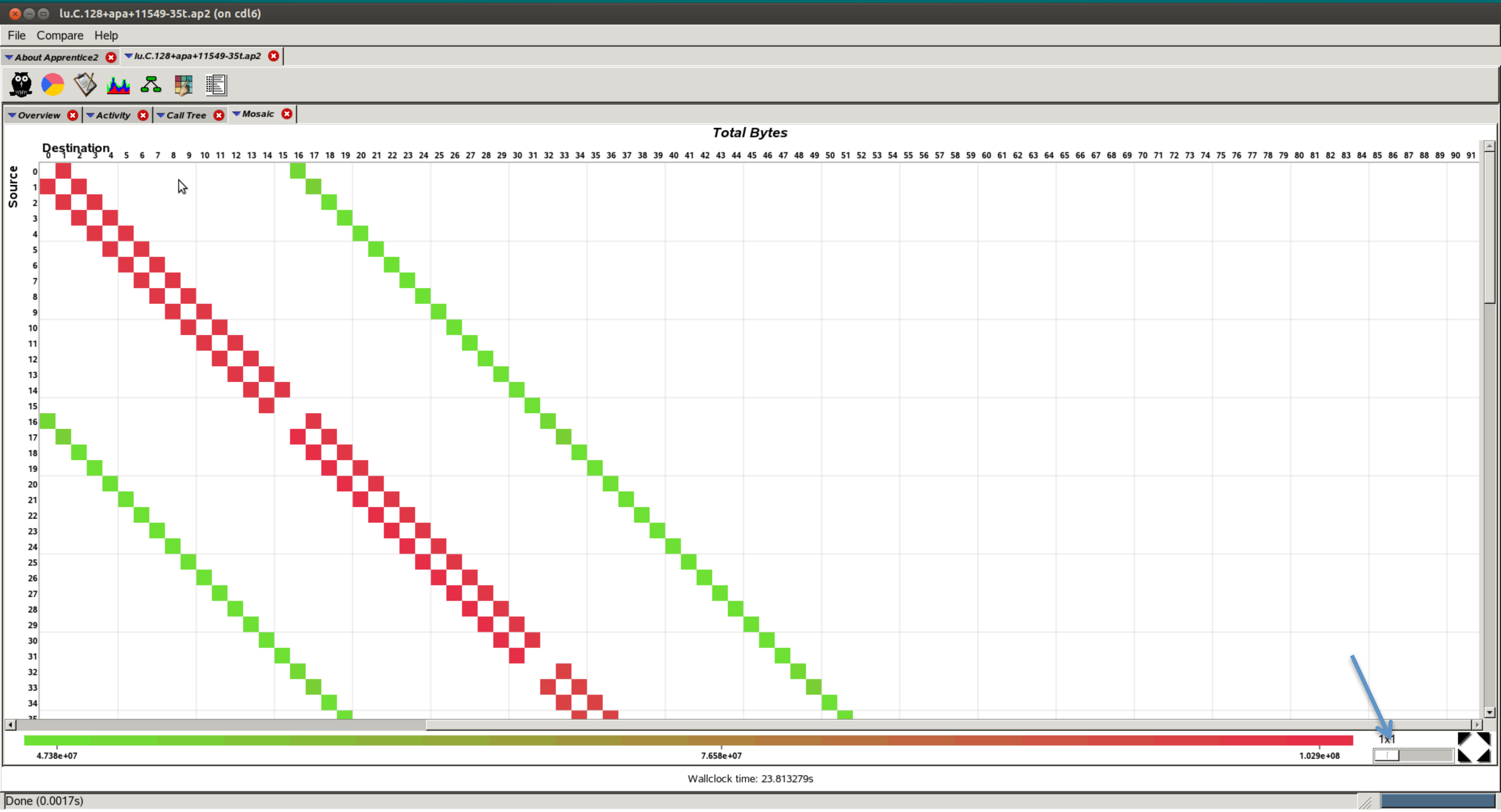


```
(24:27)->(20:23): 102,863,408
```

Wallclock time: 23.813279s

Done (0.0017s)

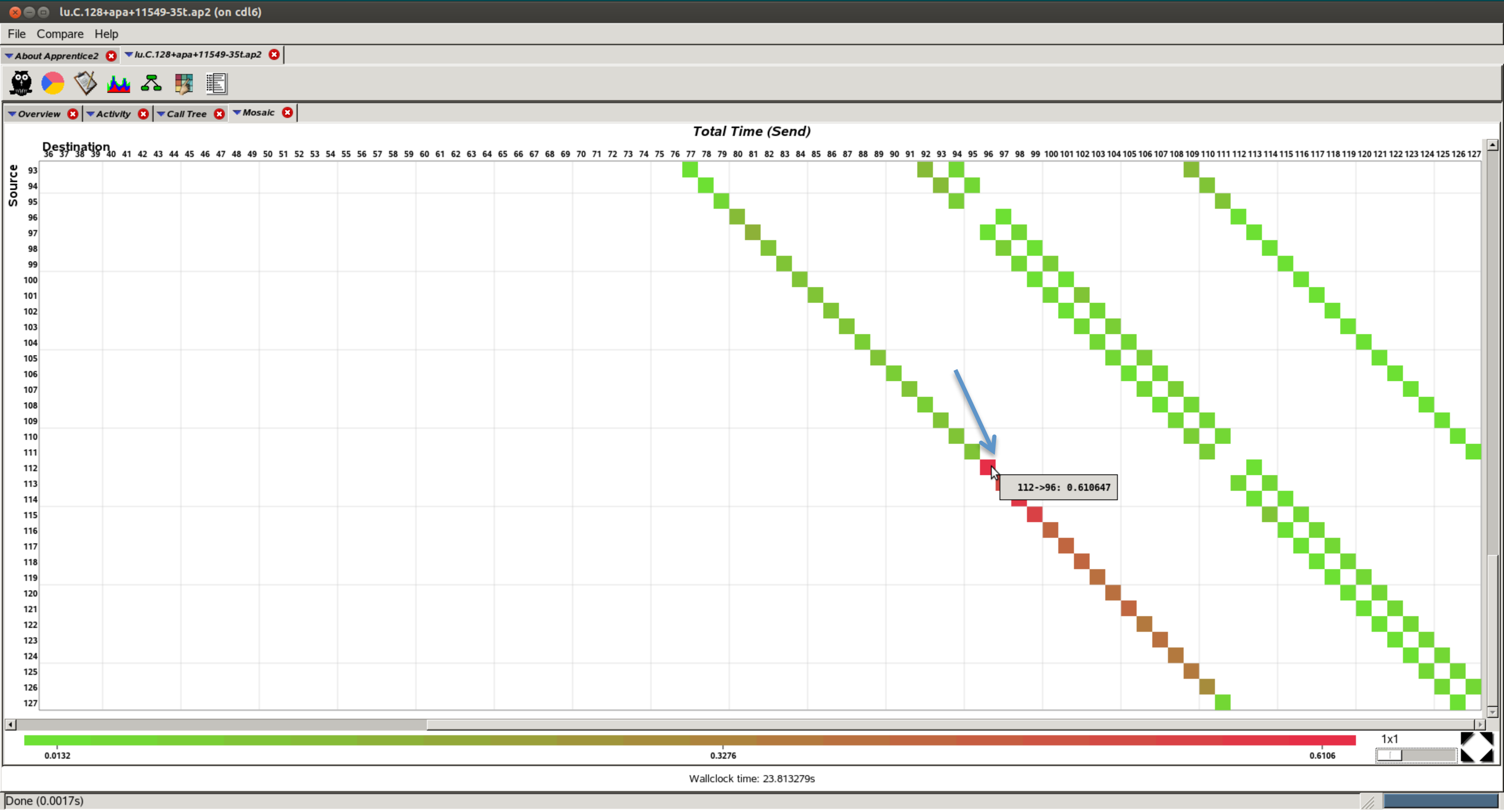
Apprentice2 – Mosaic V



Wallclock time: 23.813279s

Done (0.0017s)

Apprentice2 – Mosaic VI

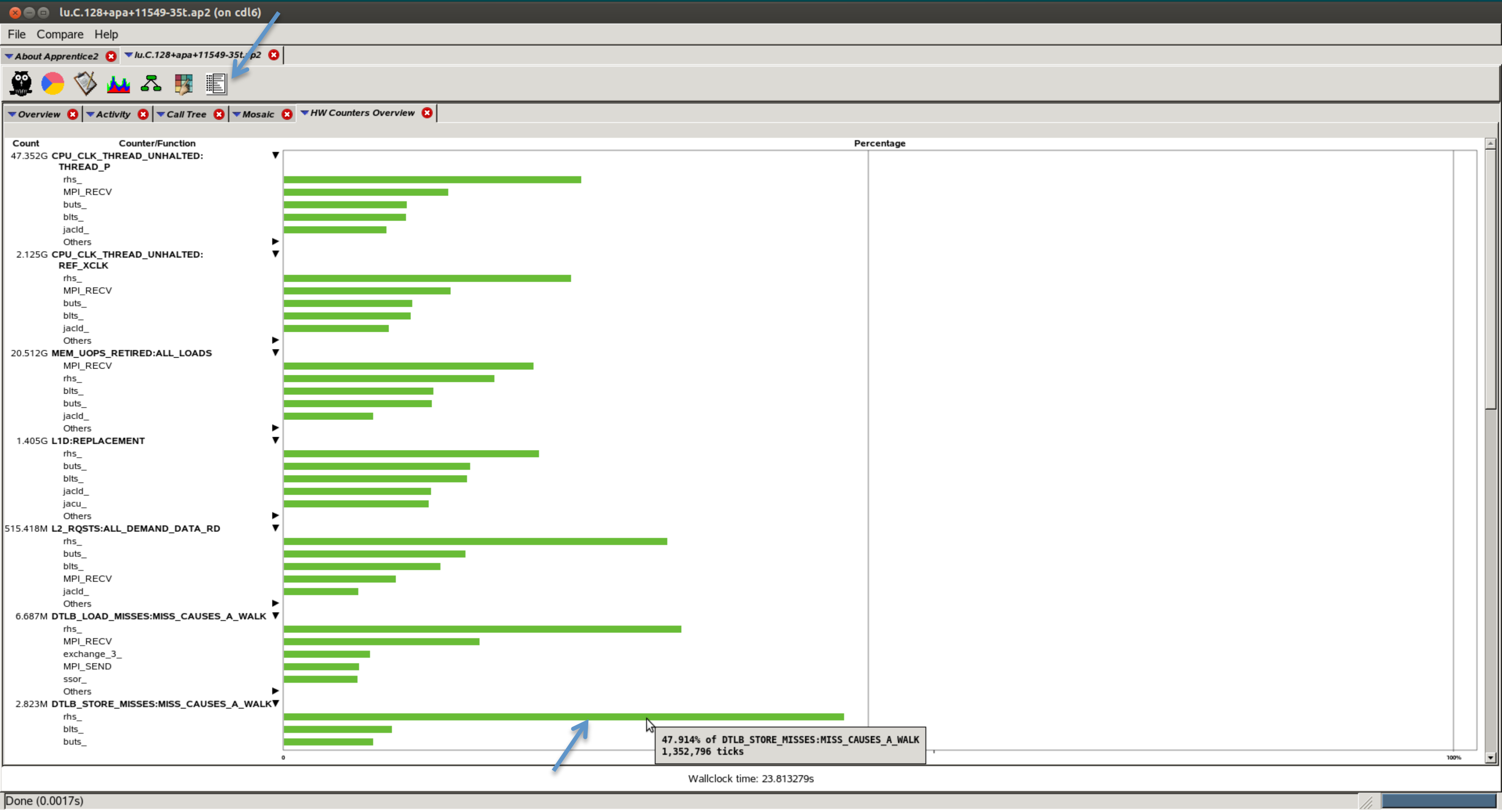


112->96: 0.610647

Wallclock time: 23.813279s

Done (0.0017s)

Apprentice2 – Hardware counters overview



lu.C.128+apa+11549-35t.ap2 (on cdl6)

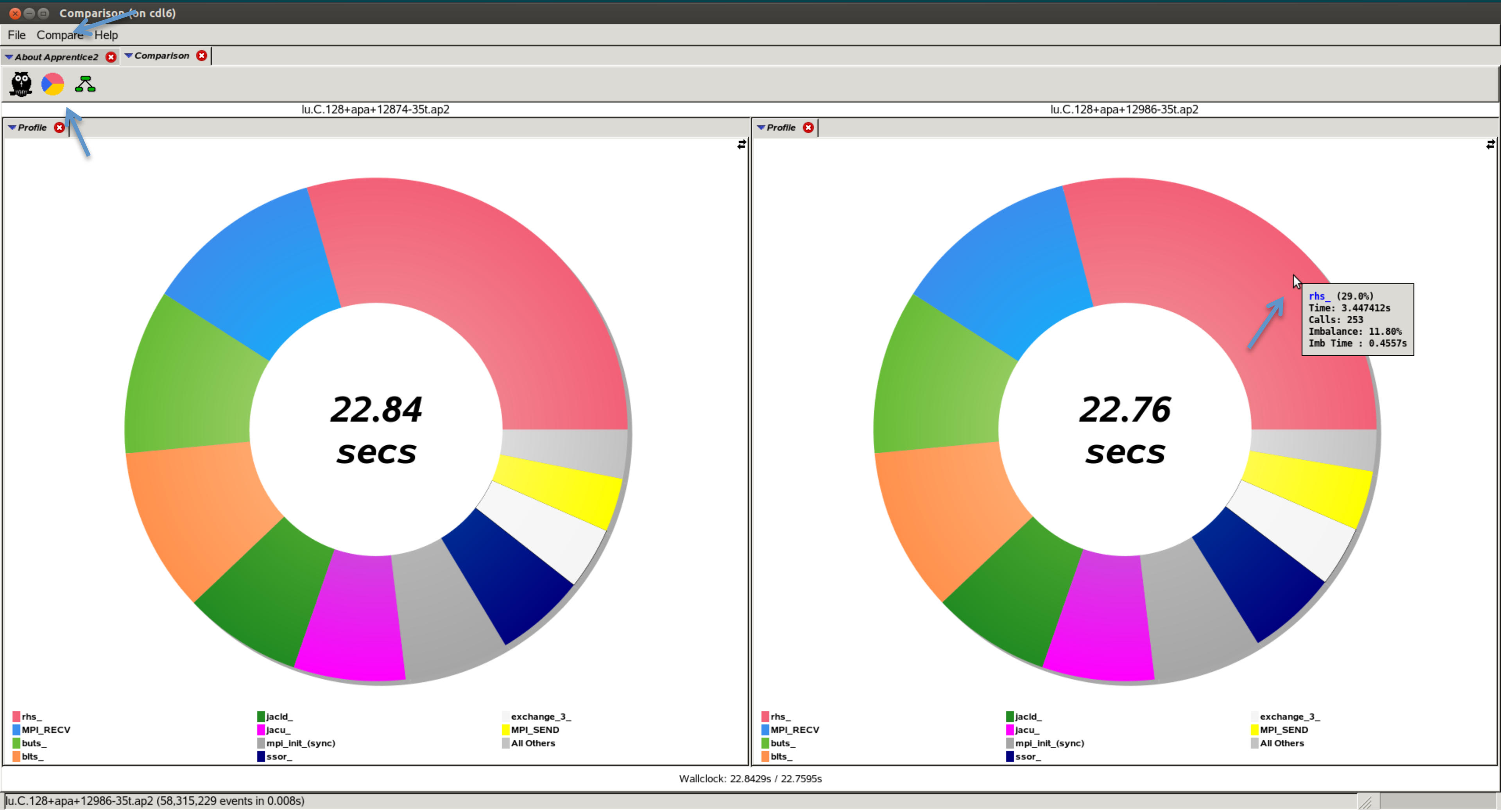
Count	Counter/Function
47.352G	CPU_CLK_THREAD_UNHALTED: THREAD_P
	rhs_
	MPI_RECV
	buts_
	blts_
	jacld_
	Others
2.125G	CPU_CLK_THREAD_UNHALTED: REF_XCLK
	rhs_
	MPI_RECV
	buts_
	blts_
	jacld_
	Others
20.512G	MEM_UOPS_RETIRED:ALL_LOADS
	MPI_RECV
	rhs_
	blts_
	buts_
	jacld_
	Others
1.405G	L1D:REPLACEMENT
	rhs_
	buts_
	blts_
	jacld_
	jacu_
	Others
515.418M	L2_RQSTS:ALL_DEMAND_DATA_RD
	rhs_
	buts_
	blts_
	MPI_RECV
	jacld_
	Others
6.687M	DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK
	rhs_
	MPI_RECV
	exchange_3_
	MPI_SEND
	ssor_
	Others
2.823M	DTLB_STORE_MISSES:MISS_CAUSES_A_WALK
	rhs_
	blts_
	buts_

```
47.914% of DTLB_STORE_MISSES:MISS_CAUSES_A_WALK
1,352,796 ticks
```

Wallclock time: 23.813279s

Done (0.0017s)

Apprentice2 – Profile comparison (v6.3.2)

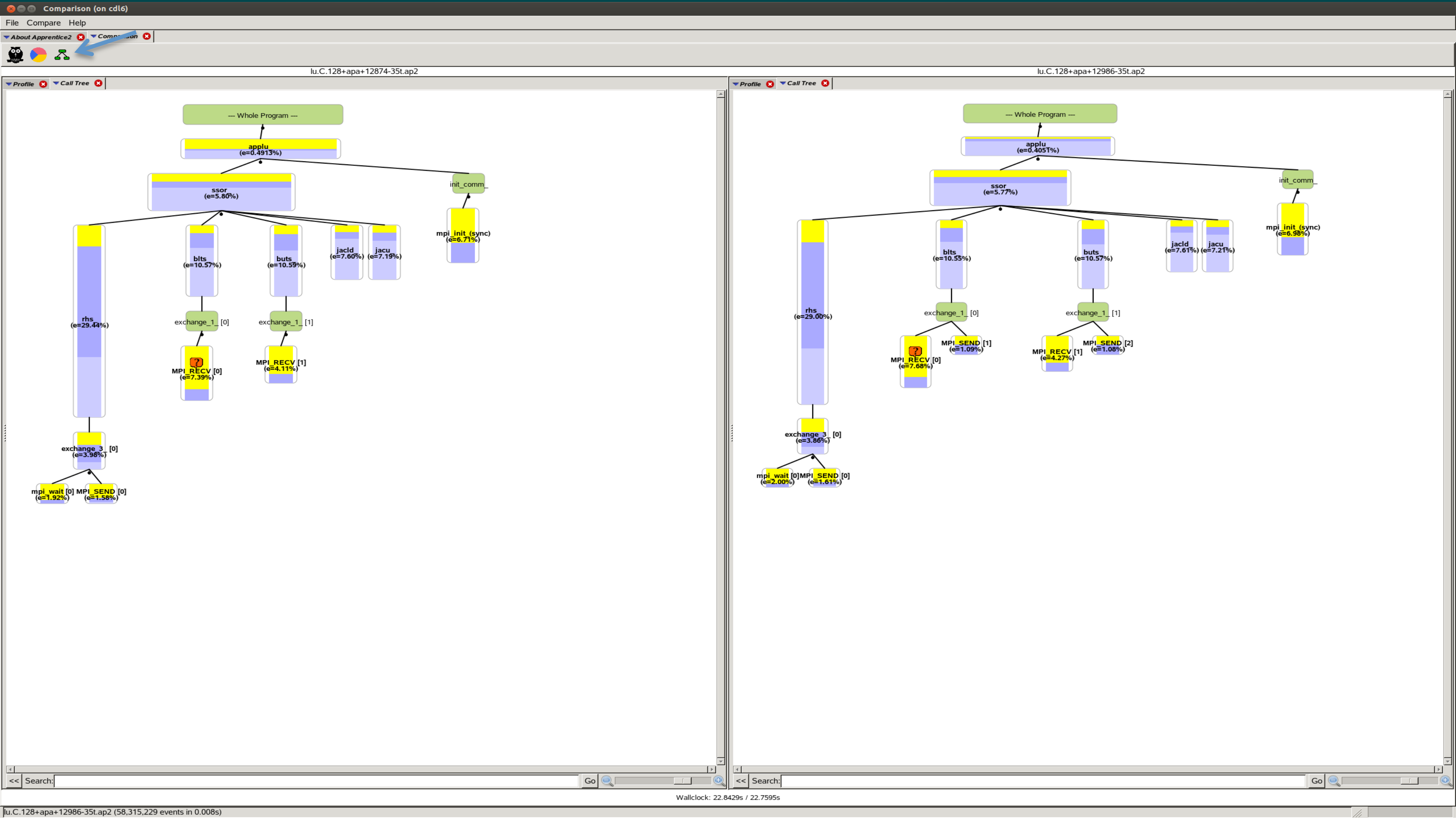


```
rhs_ (29.0%)
Time: 3.447412s
Calls: 253
Imbalance: 11.80%
Imb Time : 0.4557s
```

Wallclock: 22.8429s / 22.7595s

lu.C.128+apa+12986-35t.ap2 (58,315,229 events in 0.008s)

Apprentice2 – Profile comparison



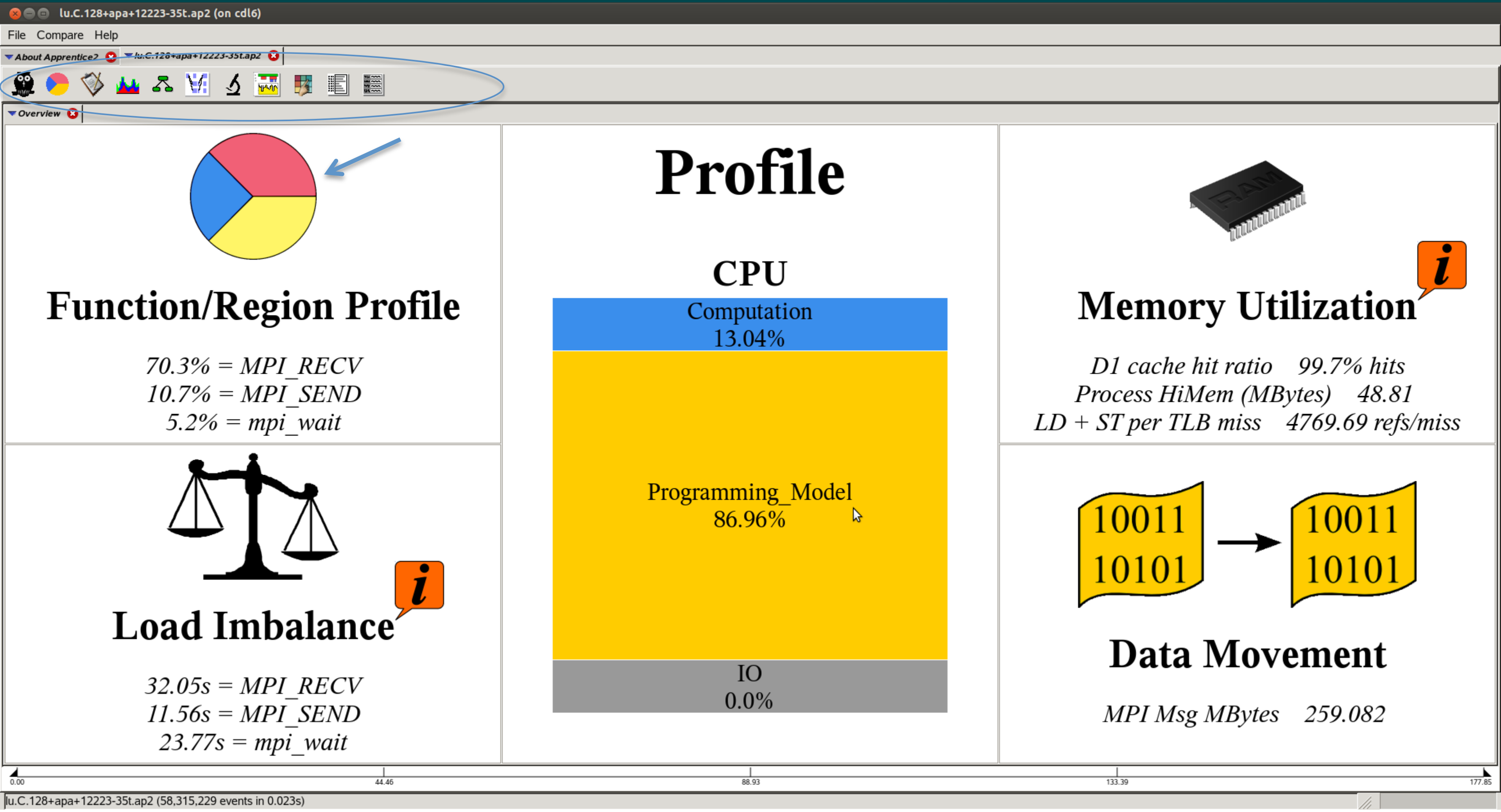
Wallclock: 22.8429s / 22.7595s

lu.C.128+apa+12986-35t.ap2 (58,315,229 events in 0.008s)

Detailed instrumentation

- Do not follow these instructions during the hands-on session
- Disable the summary of the performance data and create one file per node
  - *export PAT_RT_SUMMARY=0*
  - *export PAT_RT_EXPFILE_MAX=0*
  - *sbatch --reservation=s001_85 submit.sh*
- Expect more overhead, the trace file size can increase from some MB to GB
- Create the *ap2* file
  - *pat_report –o detailed_report_lu_C_64.txt lu.C.64+apa+PID-XXt*
- *Use Apprentice2*
  - *app2 lu.C.64+apa+PID-XXt.ap2*

Detailed instrumentation – Example LU.C.16



lu.C.128+apa+12223-35t.ap2 (on cdl6)
File Compare Help
About Apprentice2
lu.C.128+apa+12223-35t.ap2
Overview

Function/Region Profile

70.3% = MPI_RECV
10.7% = MPI_SEND
5.2% = mpi_wait

Load Imbalance

32.05s = MPI_RECV
11.56s = MPI_SEND
23.77s = mpi_wait

Profile

CPU

Computation 13.04%

Programming_Model 86.96%

IO 0.0%

Memory Utilization

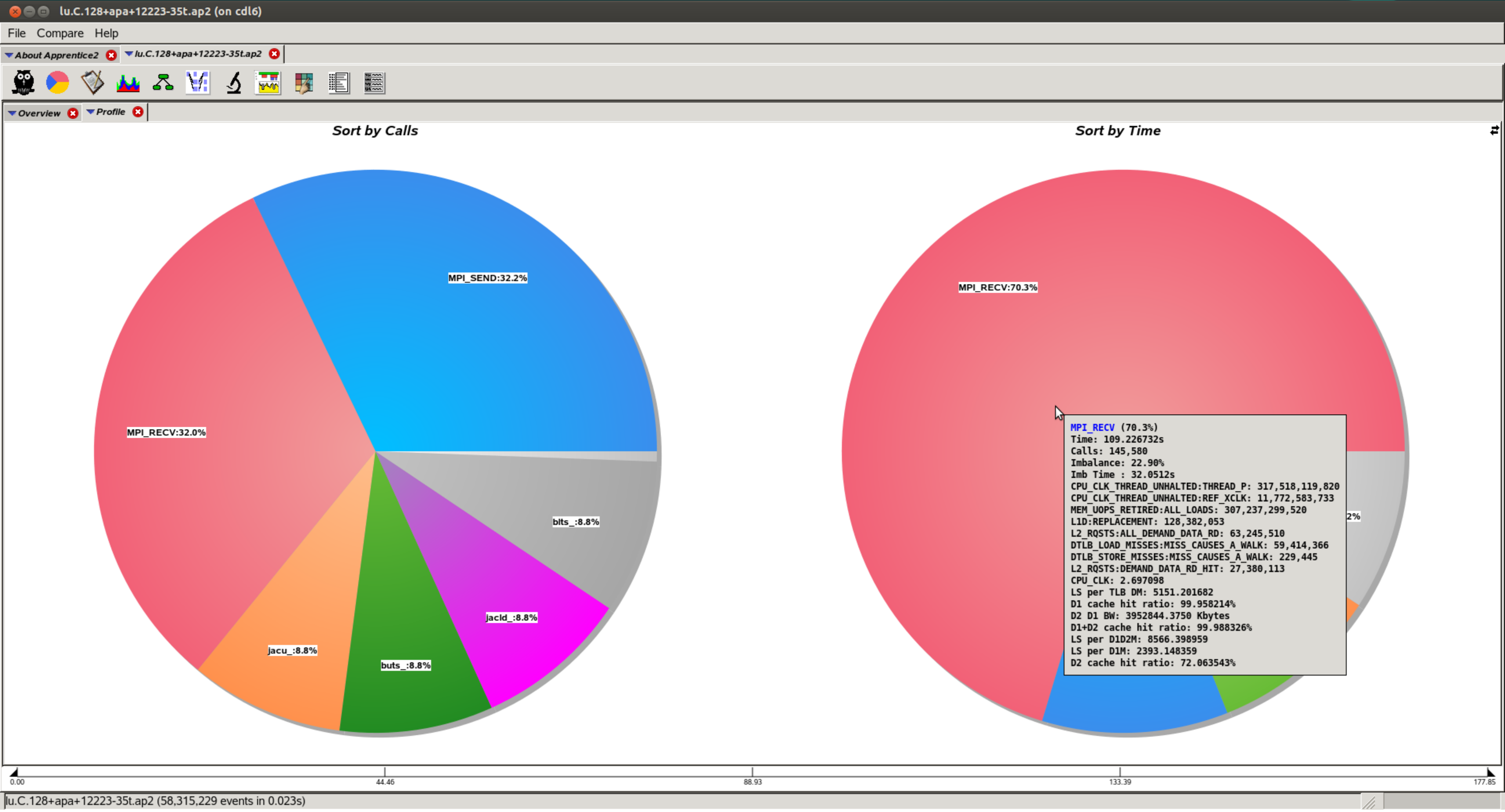
D1 cache hit ratio 99.7% hits
Process HiMem (MBytes) 48.81
LD + ST per TLB miss 4769.69 refs/miss

Data Movement

MPI Msg MBytes 259.082

lu.C.128+apa+12223-35t.ap2 (58,315,229 events in 0.023s)

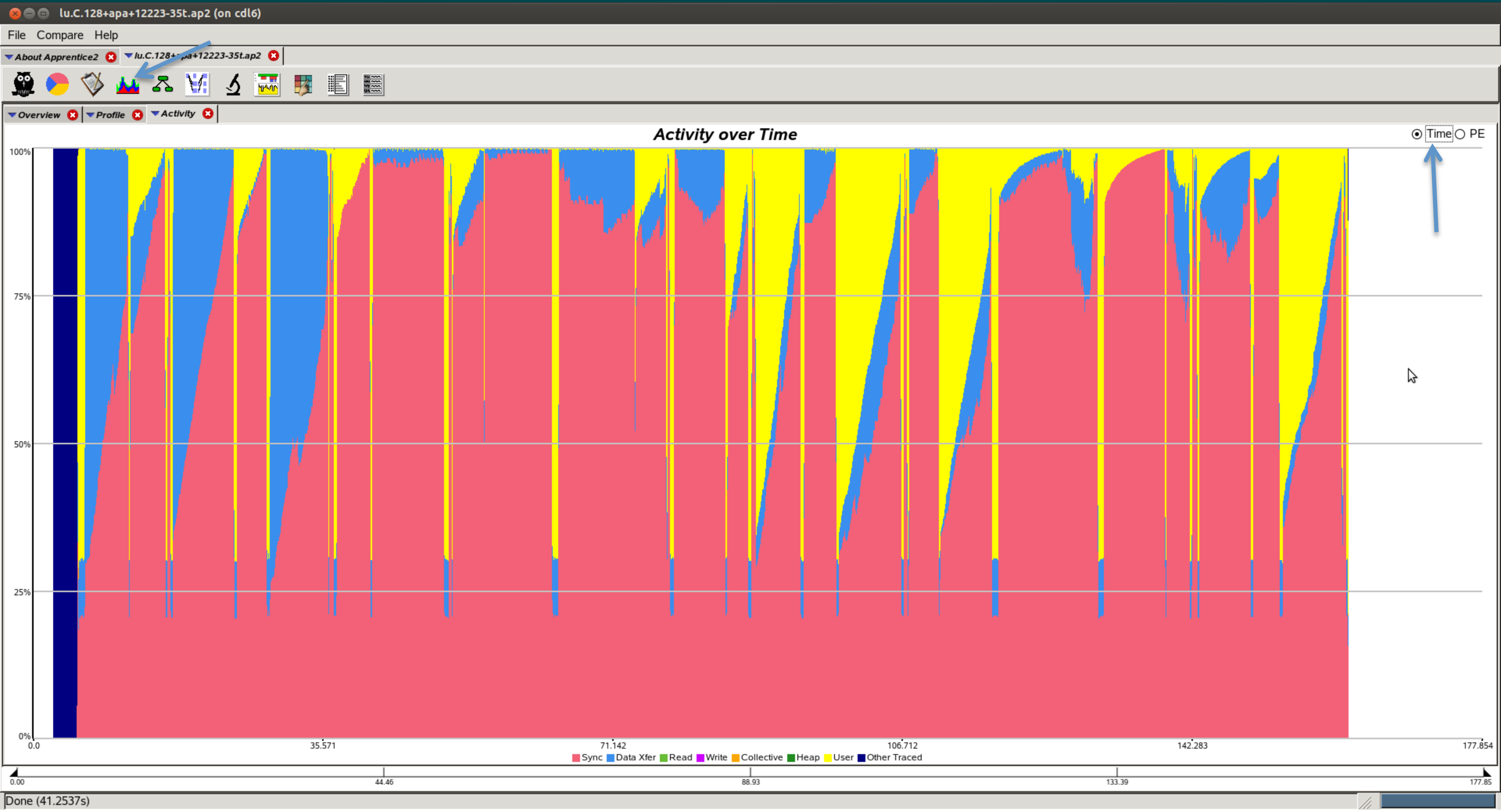
Detailed instrumentation – Profile



```
MPI_RECV (70.3%)
Time: 109.226732s
Calls: 145,580
Imbalance: 22.90%
Imb Time : 32.0512s
CPU_CLK_THREAD_UNHALTED:THREAD_P: 317,518,119,820
CPU_CLK_THREAD_UNHALTED:REF_XCLK: 11,772,583,733
MEM_UOPS_RETIRED:ALL_LOADS: 307,237,299,520
L1D:REPLACEMENT: 128,382,053
L2_RQSTS:ALL_DEMAND_DATA_RD: 63,245,510
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK: 59,414,366
DTLB_STORE_MISSES:MISS_CAUSES_A_WALK: 229,445
L2_RQSTS:DEMAND_DATA_RD_HIT: 27,380,113
CPU_CLK: 2.697098
LS per TLB DM: 5151.201682
D1 cache hit ratio: 99.958214%
D2 D1 BW: 3952844.3750 Kbytes
D1+D2 cache hit ratio: 99.988326%
LS per D1D2M: 8566.398959
LS per D1M: 2393.148359
D2 cache hit ratio: 72.063543%
```

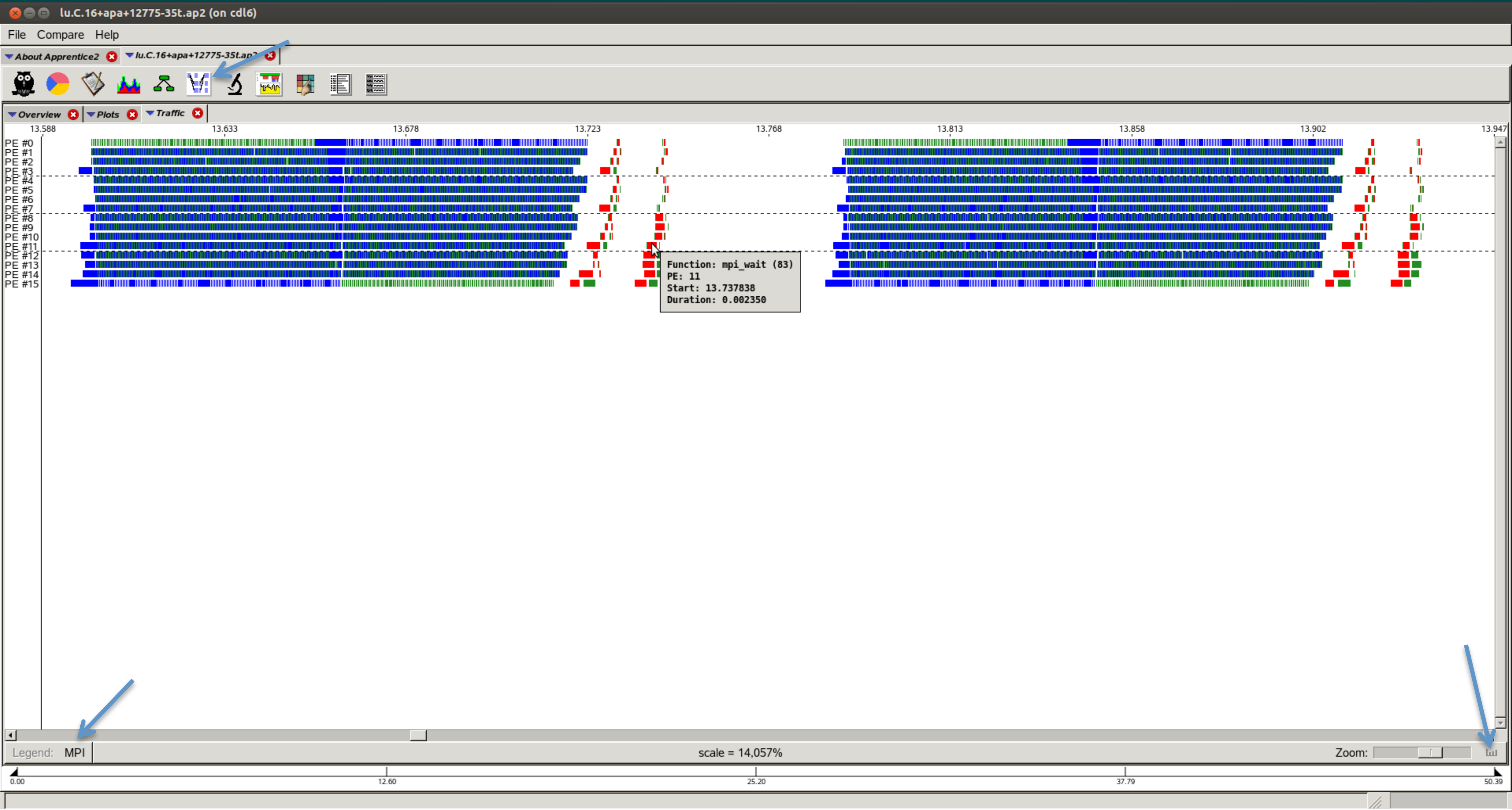
lu.C.128+apa+12223-35t.ap2 (58,315,229 events in 0.023s)

Detailed instrumentation – Activity over time



Done (41.2537s)

Detailed instrumentation – Traffic Report



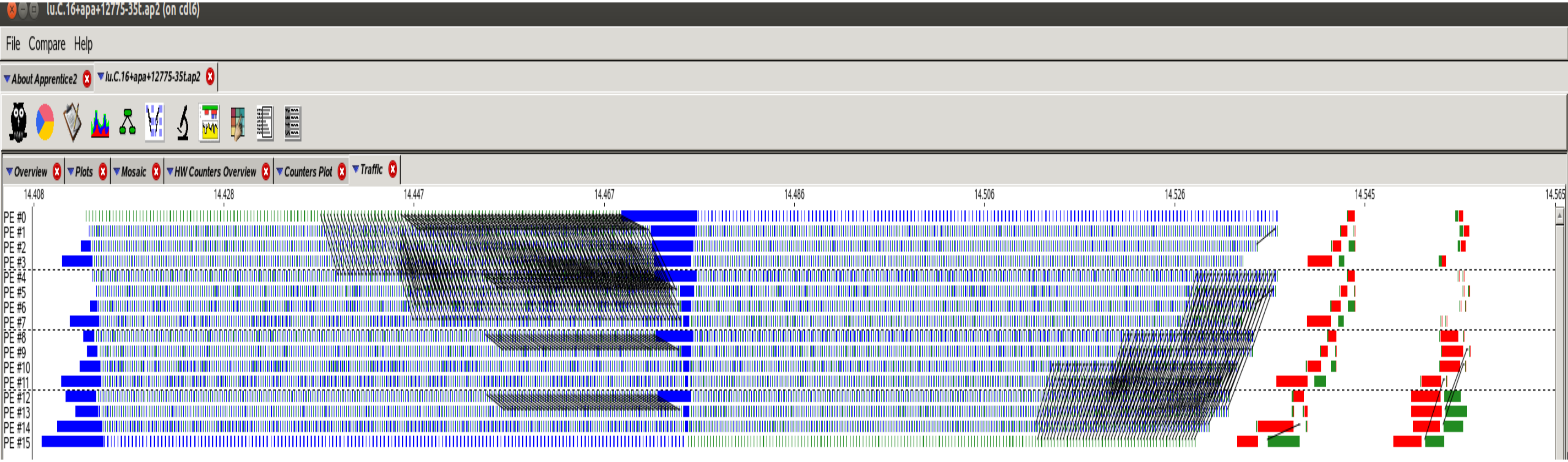
lu.C.16+apa+12775-35t.ap2 (on cdl6)

```
Function: mpi_wait (83)
PE: 11
Start: 13.737838
Duration: 0.002350
```

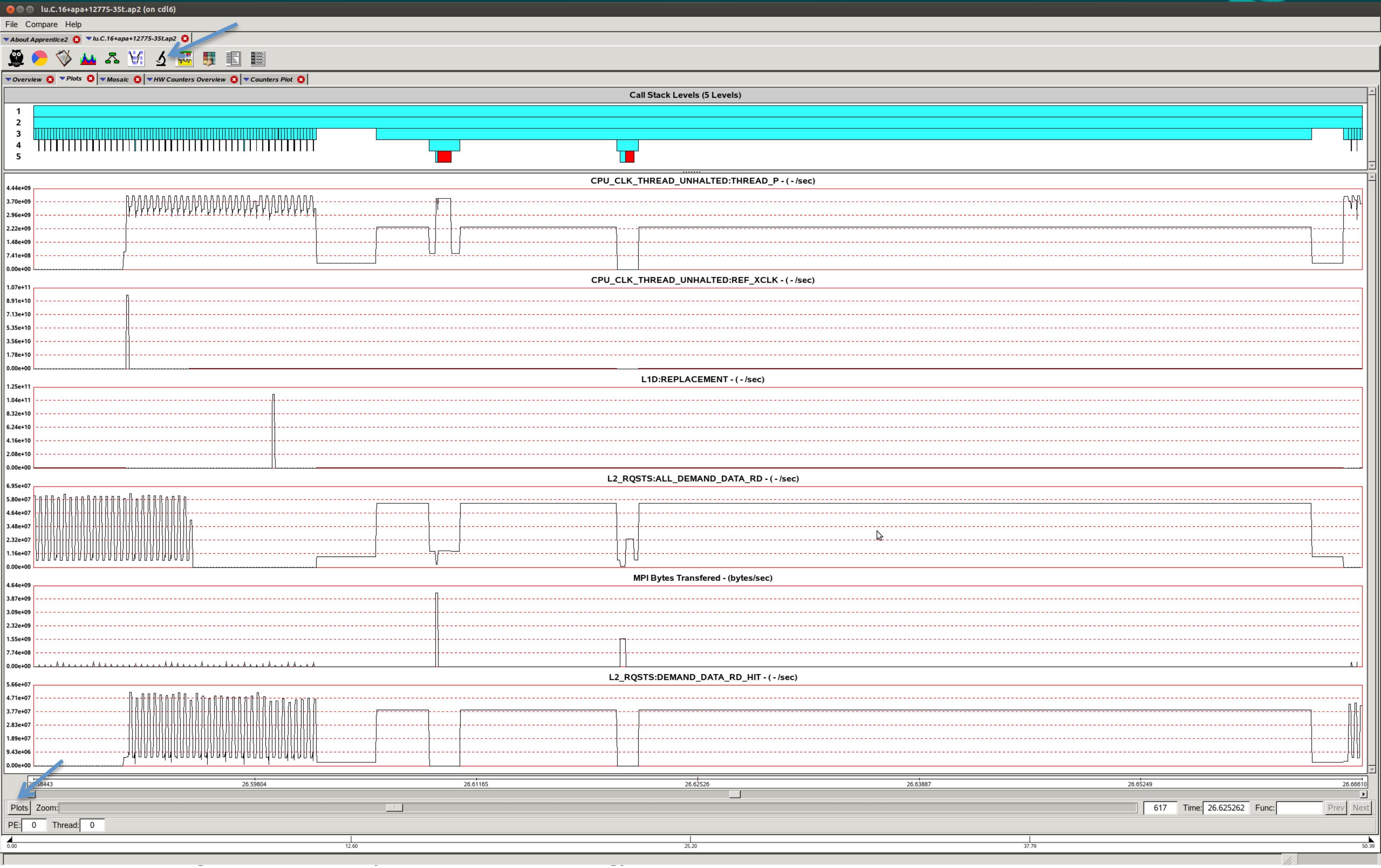
Legend: MPI

scale = 14,057%

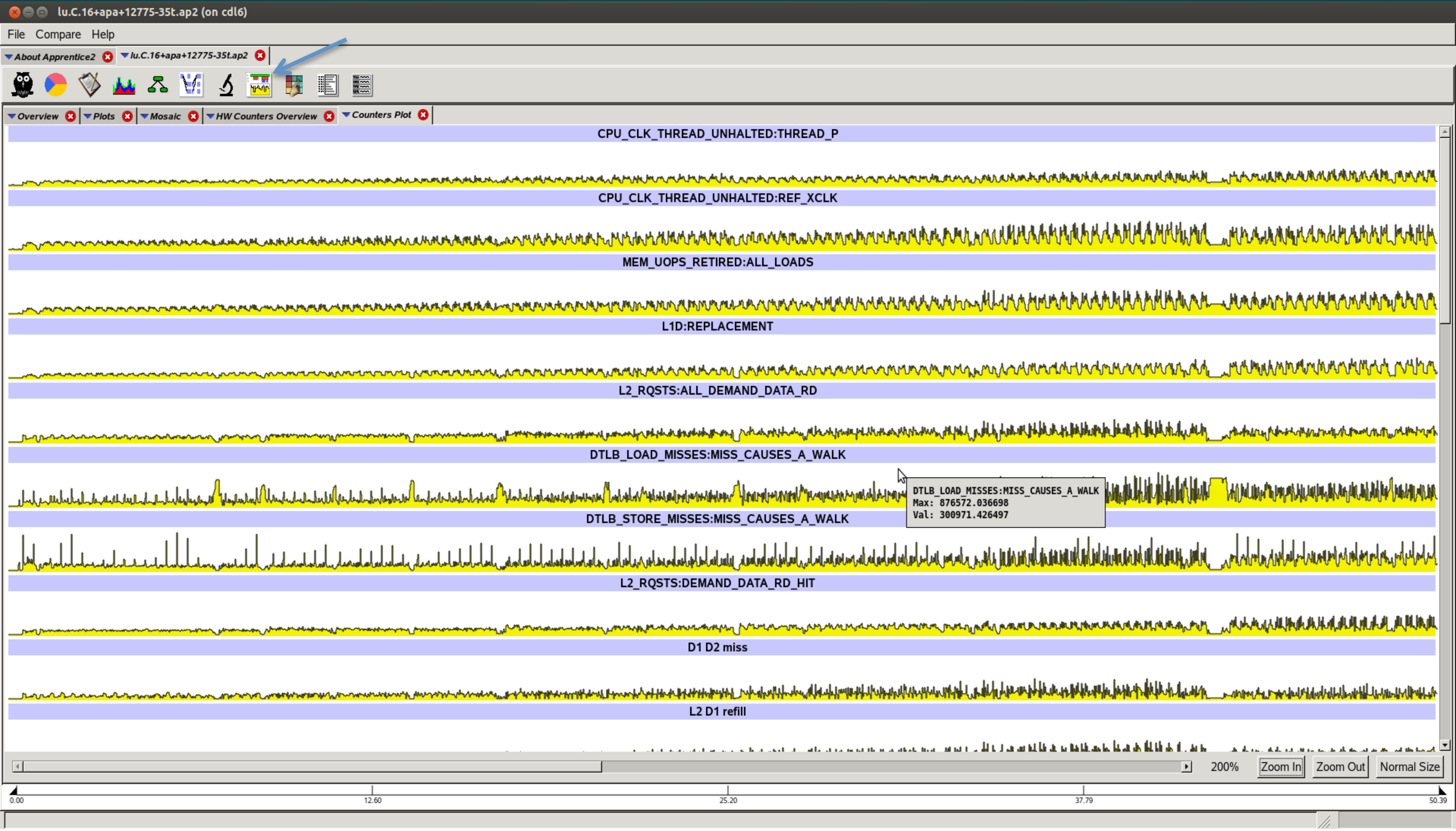
Detailed instrumentation – Traffic Report with links



Detailed instrumentation – Plots



Detailed instrumentation – Counters Plot


lu.C.16+apa+12775-35t.ap2 (on cdl6)
File
Compare
Help
About Apprentice2
lu.C.16+apa+12775-35t.ap2
Overview
Plots
Mosaic
HW Counters Overview
Counters Plot
CPU_CLK_THREAD_UNHALTED:THREAD_P
CPU_CLK_THREAD_UNHALTED:REF_XCLK
MEM_UOPS_RETIRED:ALL_LOADS
L1D:REPLACEMENT
L2_RQSTS:ALL_DEMAND_DATA_RD
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK
DTLB_STORE_MISSES:MISS_CAUSES_A_WALK
L2_RQSTS:DEMAND_DATA_RD_HIT
D1 D2 miss
L2 D1 refill
200%
Zoom In
Zoom Out
Normal Size
0.00
12.60
25.20
37.79
50.39

```
DTLB_LOAD_MISSES:MISS_CAUSES_A_WALK
Max: 876572.036698
Val: 300971.426497
```

Reveal

A tool to port your application to OpenMP

Reveal

- ❖ Reveal is Cray’s next-generation integrated performance analysis and code optimization tool.

  - Source code navigation using whole program analysis (data provided by the Cray compilation environment only)
  - Coupling with performance data collected during execution by CrayPAT. Understand which high level serial loops could benefit from parallelism.
  - Enhanced loop mark listing functionality.
  - Dependency information for targeted loops
  - Assist users optimize code by providing variable scoping feedback and suggested compile directives.

Prepare for Reveal

- Load Perftools
  - *module unload darshan*
  - *module load perftools-base/6.3.2*
  - *module load perftools/6.3.2*

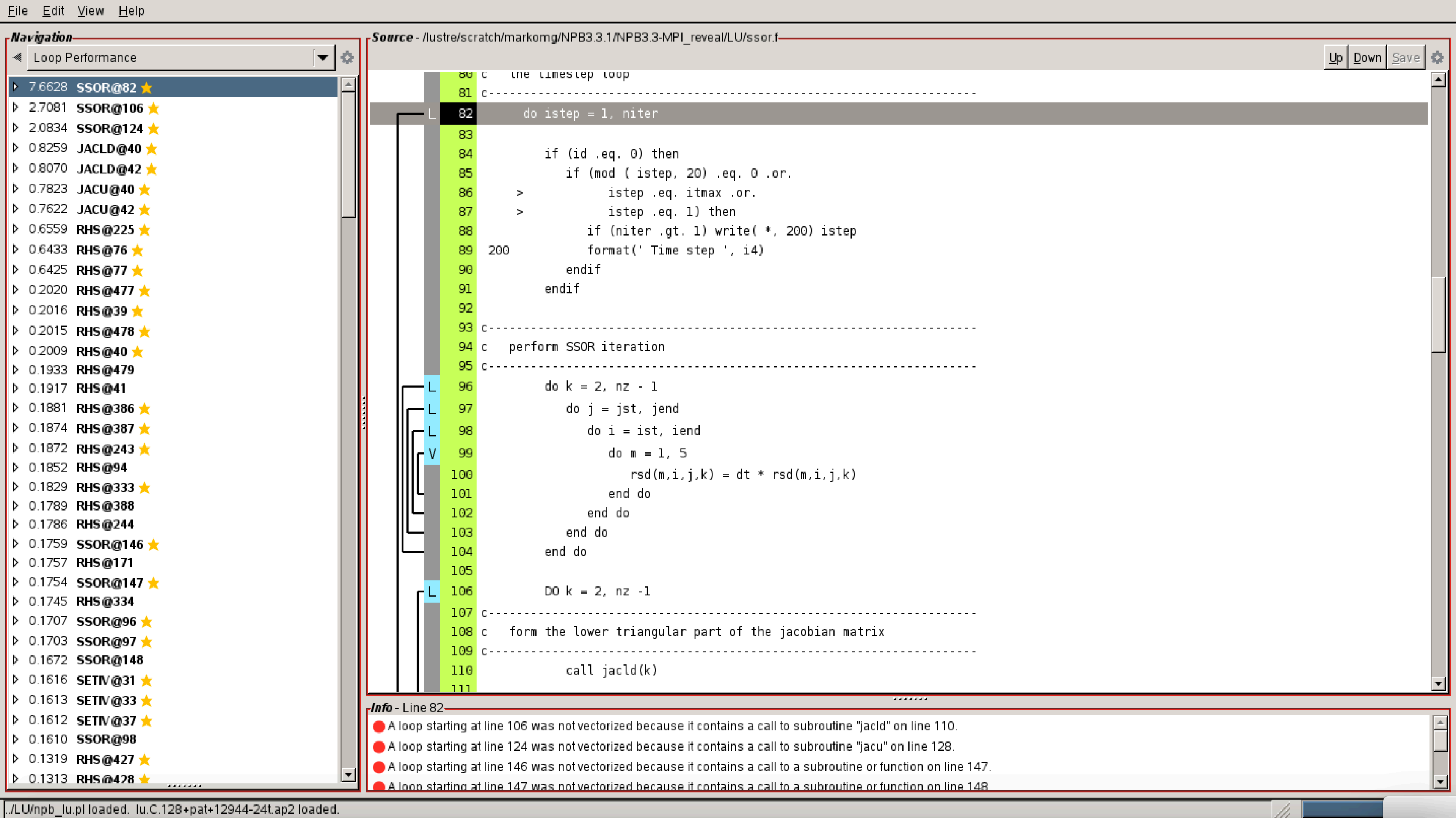
- Compile the code
  - *cd performance_workshop/NPB3.3-MPI_reveal*
  - *make clean*
  - In *the config.make.def* file
    - *MPIF77 = ftn -h profile_generate -hpl=npb_lu.pl -h noomp -h noacc*
    - *FMPI_LIB = -h profile_generate -hpl=npb_lu.pl -h noomp -h noacc*
  - *make LU NPROCS=64 CLASS=C*
    - *"WARNING: PerfTools is saving object files from a temporary directory into directory…"*
  - *cd bin*
- The new binary is called lu.C.64 is not instrumented yet

Prepare and load Reveal

- ❖ Prepare the binary for tracing
  - *pat_build –w lu.C.64*

- ❖ Uncomment the line 16 in file submit.sh (the one with lu.C.64+pat)
- ❖ sbatch --reservation=s1001_85 submit.sh
- ❖ pat_report -o reveal.txt lu.C.64+pat+PID-XXt.xf
- ❖ reveal ../LU/npb_lu.pl ./lu.C.64+pat+PID-XXt.ap2

Reveal – Loop Performance



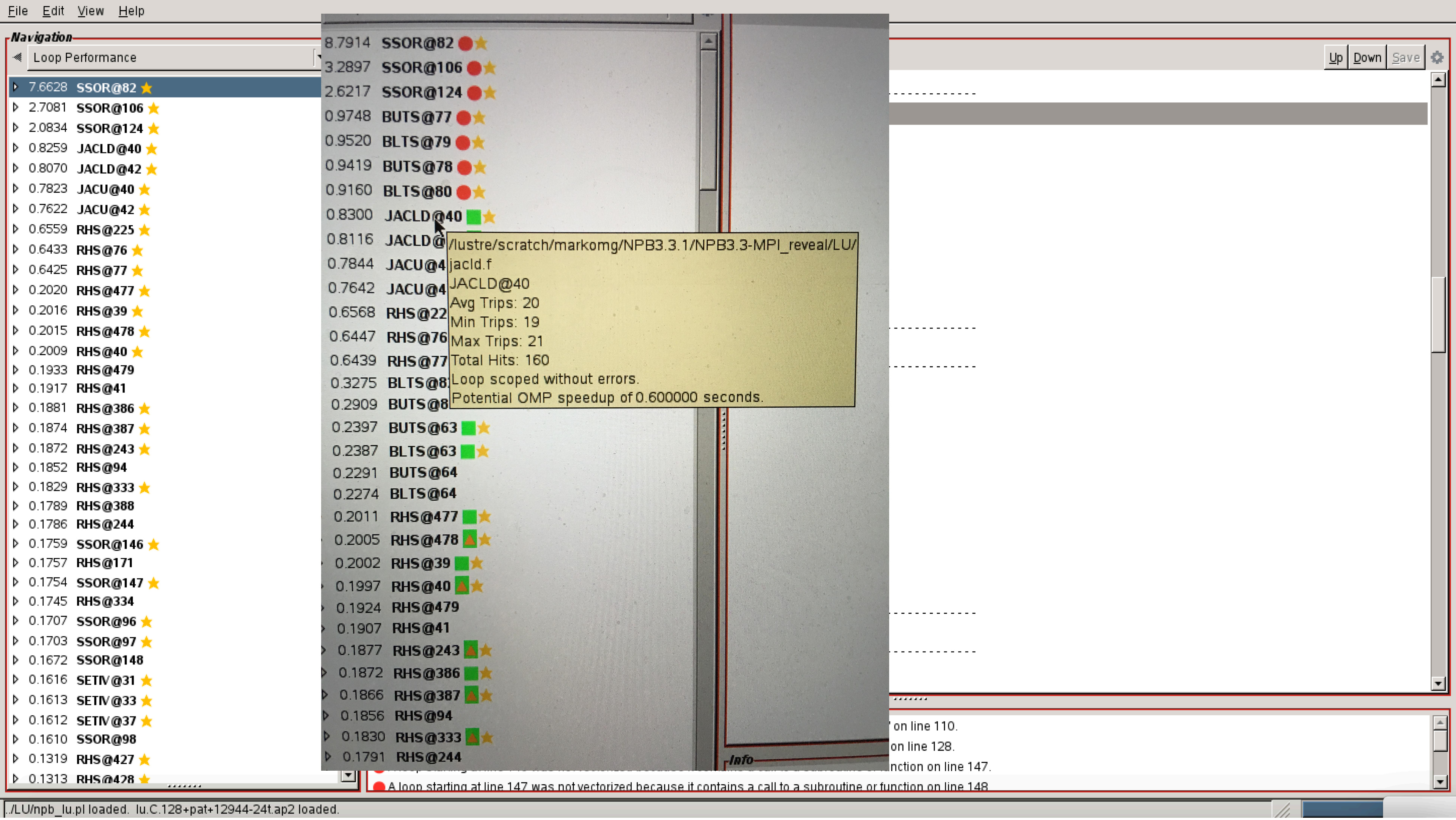
```
7.6628  SSOR@82 ★
2.7081  SSOR@106 ★
2.0834  SSOR@124 ★
0.8259  JACLD@40 ★
0.8070  JACLD@42 ★
0.7823  JACU@40 ★
0.7622  JACU@42 ★
0.6559  RHS@225 ★
0.6433  RHS@76 ★
0.6425  RHS@77 ★
0.2020  RHS@477 ★
0.2016  RHS@39 ★
0.2015  RHS@478 ★
0.2009  RHS@40 ★
0.1933  RHS@479
0.1917  RHS@41
0.1881  RHS@386 ★
0.1874  RHS@387 ★
0.1872  RHS@243 ★
0.1852  RHS@94
0.1829  RHS@333 ★
0.1789  RHS@388
0.1786  RHS@244
0.1759  SSOR@146 ★
0.1757  RHS@171
0.1754  SSOR@147 ★
0.1745  RHS@334
0.1707  SSOR@96 ★
0.1703  SSOR@97 ★
0.1672  SSOR@148
0.1616  SETIV@31 ★
0.1613  SETIV@33 ★
0.1612  SETIV@37 ★
0.1610  SSOR@98
0.1319  RHS@427 ★
0.1313  RHS@428 ★
```

```
     80 c   the timestep loop
     81 c---------------------------------------------------------------------
L    82          do istep = 1, niter
     83
     84             if (id .eq. 0) then
     85                if (mod ( istep, 20) .eq. 0 .or.
     86     >             istep .eq. itmax .or.
     87     >             istep .eq. 1) then
     88                   if (niter .gt. 1) write( *, 200) istep
     89  200              format(' Time step ', i4)
     90                endif
     91             endif
     92
     93 c---------------------------------------------------------------------
     94 c   perform SSOR iteration
     95 c---------------------------------------------------------------------
L    96             do k = 2, nz - 1
L    97                do j = jst, jend
L    98                   do i = ist, iend
V    99                      do m = 1, 5
    100                         rsd(m,i,j,k) = dt * rsd(m,i,j,k)
    101                      end do
    102                   end do
    103                end do
    104             end do
    105
L   106             DO k = 2, nz -1
    107 c---------------------------------------------------------------------
    108 c   form the lower triangular part of the jacobian matrix
    109 c---------------------------------------------------------------------
    110                call jacld(k)
    111
```

- ● A loop starting at line 106 was not vectorized because it contains a call to subroutine "jacld" on line 110.
- ● A loop starting at line 124 was not vectorized because it contains a call to subroutine "jacu" on line 128.
- ● A loop starting at line 146 was not vectorized because it contains a call to a subroutine or function on line 147.
- ● A loop starting at line 147 was not vectorized because it contains a call to a subroutine or function on line 148

./LU/npb_lu.pl loaded. lu.C.128+pat+12944-24t.ap2 loaded.

Reveal – Loop performance – Potential Speedup



```
7.6628  SSOR@82 ★
2.7081  SSOR@106 ★
2.0834  SSOR@124 ★
0.8259  JACLD@40 ★
0.8070  JACLD@42 ★
0.7823  JACU@40 ★
0.7622  JACU@42 ★
0.6559  RHS@225 ★
0.6433  RHS@76 ★
0.6425  RHS@77 ★
0.2020  RHS@477 ★
0.2016  RHS@39 ★
0.2015  RHS@478 ★
0.2009  RHS@40 ★
0.1933  RHS@479
0.1917  RHS@41
0.1881  RHS@386 ★
0.1874  RHS@387 ★
0.1872  RHS@243 ★
0.1852  RHS@94
0.1829  RHS@333 ★
0.1789  RHS@388
0.1786  RHS@244
0.1759  SSOR@146 ★
0.1757  RHS@171
0.1754  SSOR@147 ★
0.1745  RHS@334
0.1707  SSOR@96 ★
0.1703  SSOR@97 ★
0.1672  SSOR@148
0.1616  SETIV@31 ★
0.1613  SETIV@33 ★
0.1612  SETIV@37 ★
0.1610  SSOR@98
0.1319  RHS@427 ★
0.1313  RHS@428 ★
```

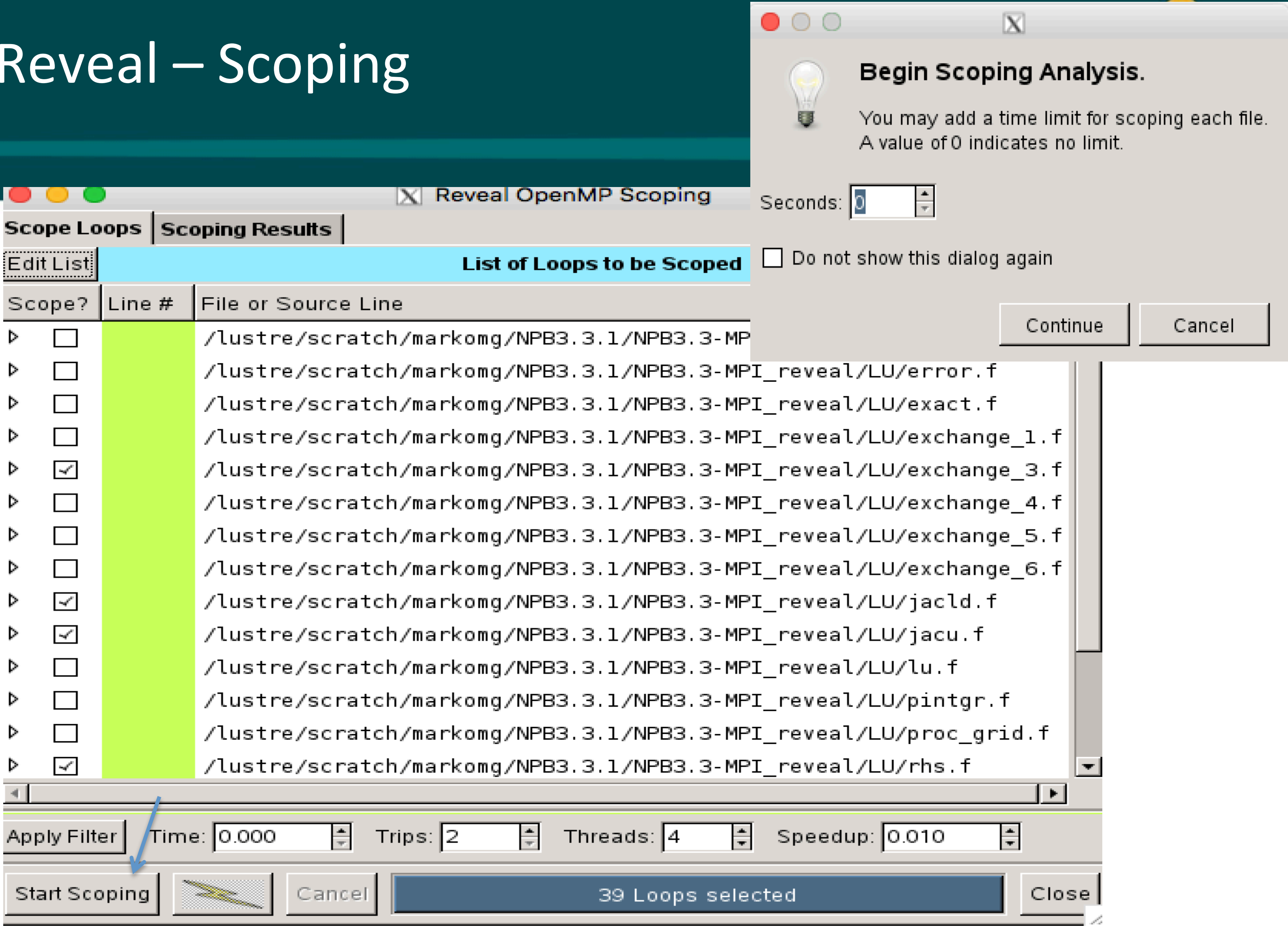
```
8.7914  SSOR@82 ●★
3.2897  SSOR@106 ●★
2.6217  SSOR@124 ●★
0.9748  BUTS@77 ●★
0.9520  BLTS@79 ●★
0.9419  BUTS@78 ●★
0.9160  BLTS@80 ●★
0.8300  JACLD@40 🟩★
0.8116  JACLD@
0.7844  JACU@4
0.7642  JACU@4
0.6568  RHS@22
0.6447  RHS@76
0.6439  RHS@77
0.3275  BLTS@8
0.2909  BUTS@8
0.2397  BUTS@63 🟩★
0.2387  BLTS@63 🟩★
0.2291  BUTS@64
0.2274  BLTS@64
0.2011  RHS@477 🟩★
0.2005  RHS@478 ▲★
0.2002  RHS@39 🟩★
0.1997  RHS@40 ▲★
0.1924  RHS@479
0.1907  RHS@41
0.1877  RHS@243 ▲★
0.1872  RHS@386 🟩★
0.1866  RHS@387 ▲★
0.1856  RHS@94
0.1830  RHS@333 ▲★
0.1791  RHS@244
```

```
/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/
jacld.f
JACLD@40
Avg Trips: 20
Min Trips: 19
Max Trips: 21
Total Hits: 160
Loop scoped without errors.
Potential OMP speedup of 0.600000 seconds.
```

```
' on line 110.
on line 128.
nction on line 147.
● A loop starting at line 147 was not vectorized because it contains a call to a subroutine or function on line 148
```

```
../LU/npb_lu.pl loaded.  lu.C.128+pat+12944-24t.ap2 loaded.
```

Reveal – Scoping



Begin Scoping Analysis.

You may add a time limit for scoping each file.
A value of 0 indicates no limit.

Seconds: 0

☐ Do not show this dialog again

Reveal OpenMP Scoping

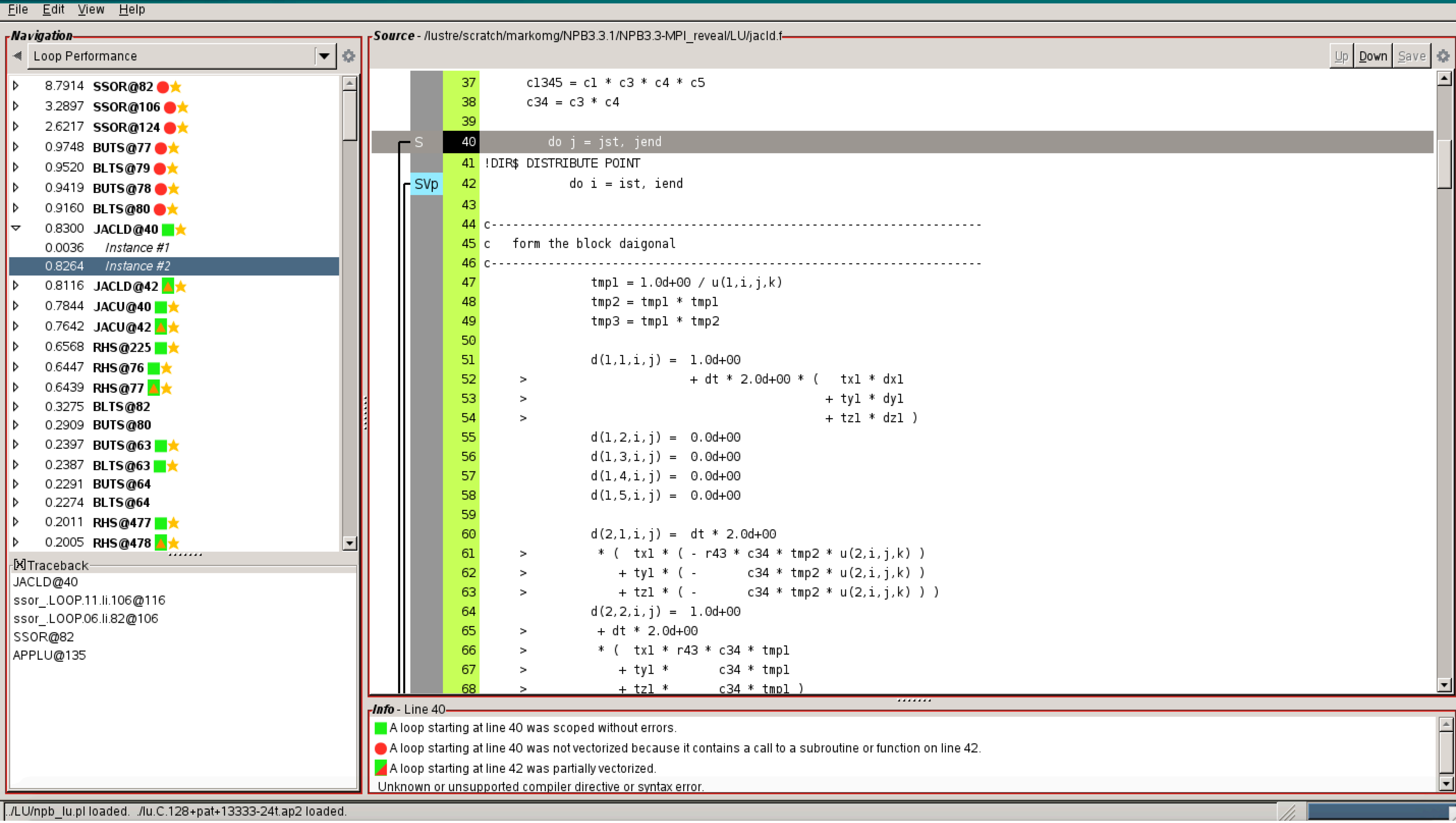
List of Loops to be Scoped

	Scope?	Line #	File or Source Line
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MP
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/error.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exact.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exchange_1.f
▷	☑		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exchange_3.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exchange_4.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exchange_5.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/exchange_6.f
▷	☑		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/jacld.f
▷	☑		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/jacu.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/lu.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/pintgr.f
▷	☐		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/proc_grid.f
▷	☑		/lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/rhs.f

Time: 0.000 Trips: 2 Threads: 4 Speedup: 0.010

39 Loops selected

Reveal – Scoping results on the Loops



8.7914	SSOR@82 ●★
3.2897	SSOR@106 ●★
2.6217	SSOR@124 ●★
0.9748	BUTS@77 ●★
0.9520	BLTS@79 ●★
0.9419	BUTS@78 ●★
0.9160	BLTS@80 ●★
0.8300	JACLD@40 🟩★
0.0036	*Instance #1*
0.8264	*Instance #2*
0.8116	JACLD@42 ◩★
0.7844	JACU@40 🟩★
0.7642	JACU@42 ◩★
0.6568	RHS@225 🟩★
0.6447	RHS@76 🟩★
0.6439	RHS@77 ◩★
0.3275	BLTS@82
0.2909	BUTS@80
0.2397	BUTS@63 🟩★
0.2387	BLTS@63 🟩★
0.2291	BUTS@64
0.2274	BLTS@64
0.2011	RHS@477 🟩★
0.2005	RHS@478 ◩★

Traceback

JACLD@40
ssor_.LOOP.11.li.106@116
ssor_.LOOP.06.li.82@106
SSOR@82
APPLU@135

Source - /lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/jacld.f

```
     37         c1345 = c1 * c3 * c4 * c5
     38         c34 = c3 * c4
     39
S    40            do j = jst, jend
     41 !DIR$ DISTRIBUTE POINT
SVp  42               do i = ist, iend
     43
     44 c---------------------------------------------------------------------
     45 c   form the block daigonal
     46 c---------------------------------------------------------------------
     47               tmp1 = 1.0d+00 / u(1,i,j,k)
     48               tmp2 = tmp1 * tmp1
     49               tmp3 = tmp1 * tmp2
     50
     51               d(1,1,i,j) =  1.0d+00
     52      >                      + dt * 2.0d+00 * (   tx1 * dx1
     53      >                                         + ty1 * dy1
     54      >                                         + tz1 * dz1 )
     55               d(1,2,i,j) =  0.0d+00
     56               d(1,3,i,j) =  0.0d+00
     57               d(1,4,i,j) =  0.0d+00
     58               d(1,5,i,j) =  0.0d+00
     59
     60               d(2,1,i,j) =  dt * 2.0d+00
     61      >         * (  tx1 * ( - r43 * c34 * tmp2 * u(2,i,j,k) )
     62      >            + ty1 * ( -       c34 * tmp2 * u(2,i,j,k) )
     63      >            + tz1 * ( -       c34 * tmp2 * u(2,i,j,k) ) )
     64               d(2,2,i,j) =  1.0d+00
     65      >         + dt * 2.0d+00
     66      >         * (  tx1 * r43 * c34 * tmp1
     67      >            + ty1 *       c34 * tmp1
     68      >            + tz1 *       c34 * tmp1  )
```

Info - Line 40

🟩 A loop starting at line 40 was scoped without errors.

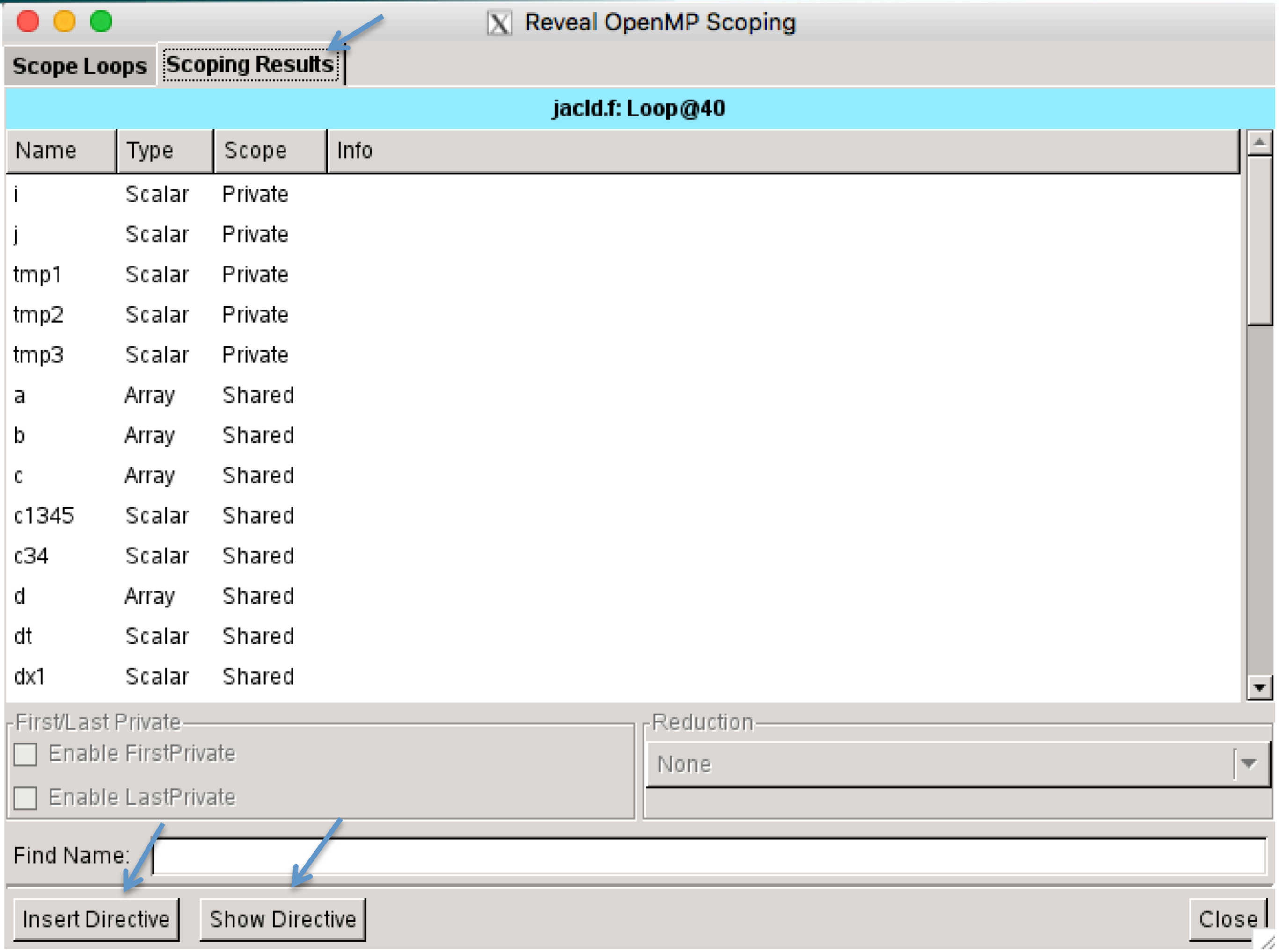
● A loop starting at line 40 was not vectorized because it contains a call to a subroutine or function on line 42.

◩ A loop starting at line 42 was partially vectorized.

Unknown or unsupported compiler directive or syntax error.

./LU/npb_lu.pl loaded. ./lu.C.128+pat+13333-24t.ap2 loaded.

Reveal – Scoping Results



jacld.f: Loop@40

Name	Type	Scope	Info
i	Scalar	Private	
j	Scalar	Private	
tmp1	Scalar	Private	
tmp2	Scalar	Private	
tmp3	Scalar	Private	
a	Array	Shared	
b	Array	Shared	
c	Array	Shared	
c1345	Scalar	Shared	
c34	Scalar	Shared	
d	Array	Shared	
dt	Scalar	Shared	
dx1	Scalar	Shared	

First/Last Private

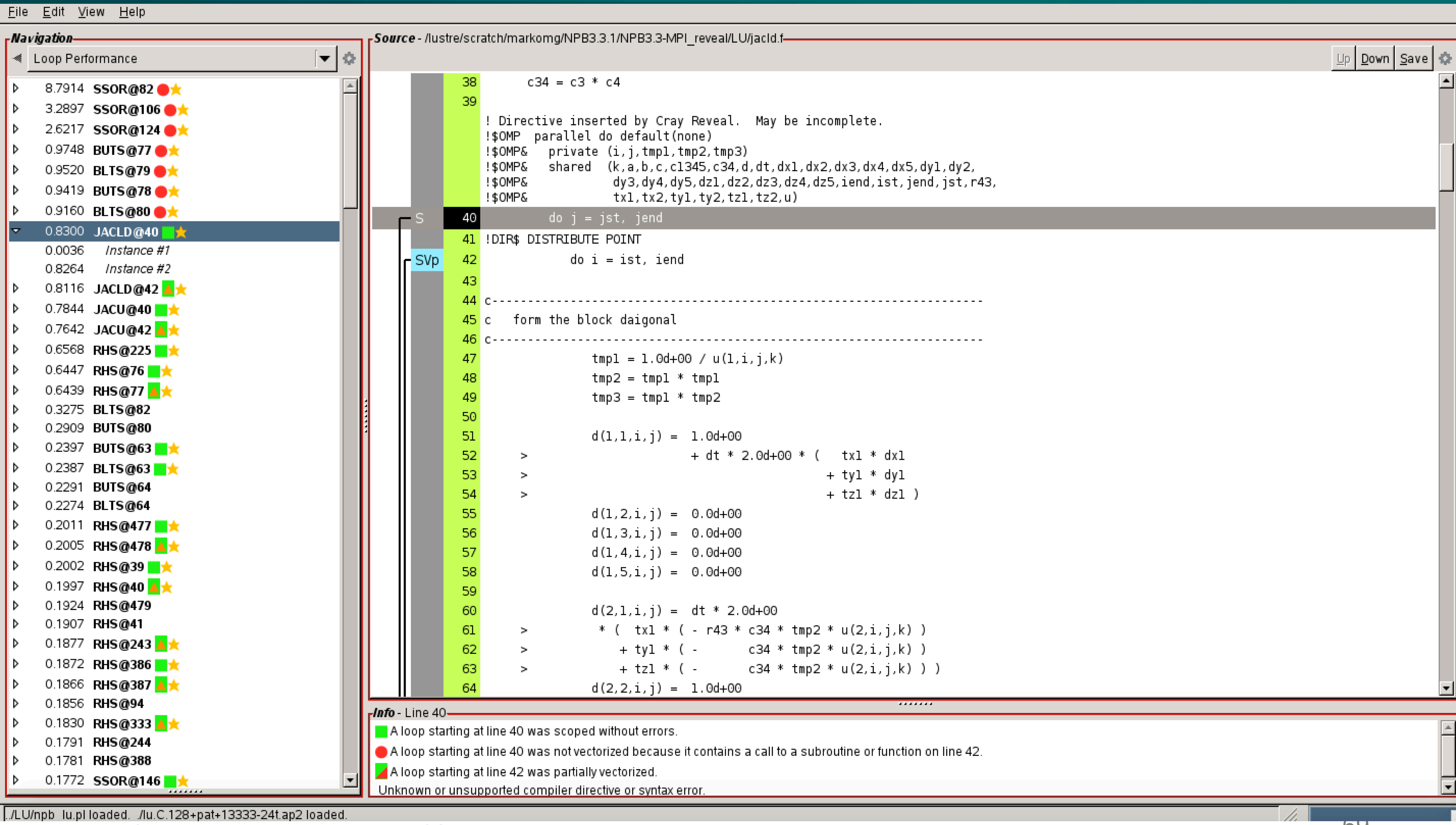
□ Enable FirstPrivate

□ Enable LastPrivate

Reduction: None

Find Name:

Reveal – OpenMP Directives



8.7914	**SSOR@82**	● ★
3.2897	**SSOR@106**	● ★
2.6217	**SSOR@124**	● ★
0.9748	**BUTS@77**	● ★
0.9520	**BLTS@79**	● ★
0.9419	**BUTS@78**	● ★
0.9160	**BLTS@80**	● ★
0.8300	**JACLD@40**	🟩 ★
0.0036	*Instance #1*	
0.8264	*Instance #2*	
0.8116	**JACLD@42**	▲ ★
0.7844	**JACU@40**	🟩 ★
0.7642	**JACU@42**	▲ ★
0.6568	**RHS@225**	🟩 ★
0.6447	**RHS@76**	🟩 ★
0.6439	**RHS@77**	▲ ★
0.3275	**BLTS@82**	
0.2909	**BUTS@80**	
0.2397	**BUTS@63**	🟩 ★
0.2387	**BLTS@63**	🟩 ★
0.2291	**BUTS@64**	
0.2274	**BLTS@64**	
0.2011	**RHS@477**	🟩 ★
0.2005	**RHS@478**	▲ ★
0.2002	**RHS@39**	🟩 ★
0.1997	**RHS@40**	▲ ★
0.1924	**RHS@479**	
0.1907	**RHS@41**	
0.1877	**RHS@243**	▲ ★
0.1872	**RHS@386**	🟩 ★
0.1866	**RHS@387**	▲ ★
0.1856	**RHS@94**	
0.1830	**RHS@333**	▲ ★
0.1791	**RHS@244**	
0.1781	**RHS@388**	
0.1772	**SSOR@146**	🟩 ★

Source - /lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/jacld.f

```
     38           c34 = c3 * c4
     39
        ! Directive inserted by Cray Reveal.  May be incomplete.
        !$OMP  parallel do default(none)
        !$OMP&   private (i,j,tmp1,tmp2,tmp3)
        !$OMP&   shared  (k,a,b,c,c1345,c34,d,dt,dx1,dx2,dx3,dx4,dx5,dy1,dy2,
        !$OMP&             dy3,dy4,dy5,dz1,dz2,dz3,dz4,dz5,iend,ist,jend,jst,r43,
        !$OMP&             tx1,tx2,ty1,ty2,tz1,tz2,u)
S    40            do j = jst, jend
     41 !DIR$ DISTRIBUTE POINT
SVp  42               do i = ist, iend
     43
     44 c-----------------------------------------------------------------------
     45 c   form the block daigonal
     46 c-----------------------------------------------------------------------
     47                  tmp1 = 1.0d+00 / u(1,i,j,k)
     48                  tmp2 = tmp1 * tmp1
     49                  tmp3 = tmp1 * tmp2
     50
     51                  d(1,1,i,j) =  1.0d+00
     52      >                        + dt * 2.0d+00 * (   tx1 * dx1
     53      >                                           + ty1 * dy1
     54      >                                           + tz1 * dz1 )
     55                  d(1,2,i,j) =  0.0d+00
     56                  d(1,3,i,j) =  0.0d+00
     57                  d(1,4,i,j) =  0.0d+00
     58                  d(1,5,i,j) =  0.0d+00
     59
     60                  d(2,1,i,j) =  dt * 2.0d+00
     61      >            * (  tx1 * ( - r43 * c34 * tmp2 * u(2,i,j,k) )
     62      >               + ty1 * ( -       c34 * tmp2 * u(2,i,j,k) )
     63      >               + tz1 * ( -       c34 * tmp2 * u(2,i,j,k) ) )
     64                  d(2,2,i,j) =  1.0d+00
```

Info - Line 40

🟩 A loop starting at line 40 was scoped without errors.

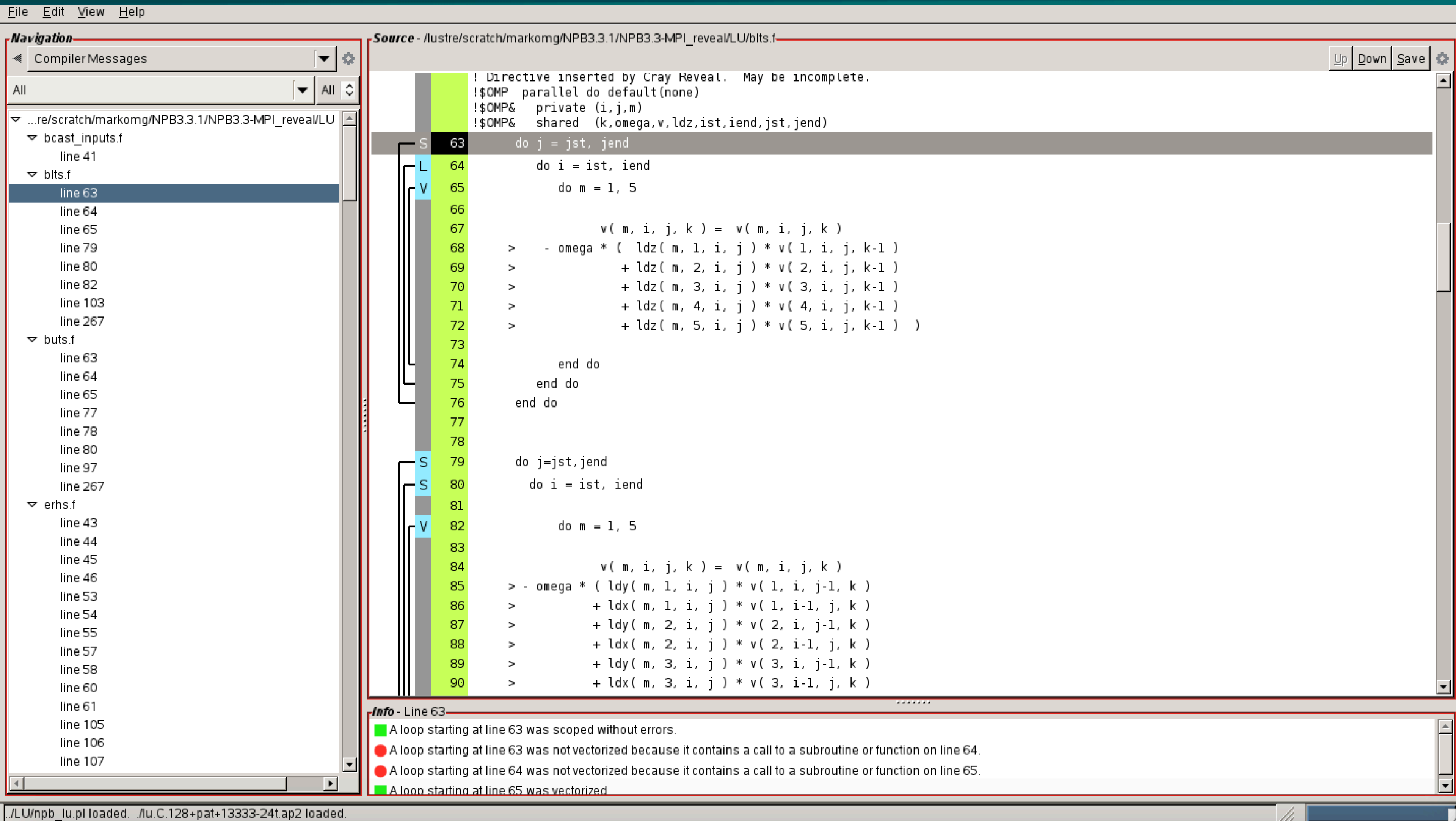
● A loop starting at line 40 was not vectorized because it contains a call to a subroutine or function on line 42.

◩ A loop starting at line 42 was partially vectorized.

Unknown or unsupported compiler directive or syntax error.

../LU/npb lu.pl loaded. ./lu.C.128+pat+13333-24t.ap2 loaded.

Reveal – Compiler messages



File Edit View Help
Navigation
Compiler Messages
All
All
Source - /lustre/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU/blts.f
Up
Down
Save

```
▽ ...re/scratch/markomg/NPB3.3.1/NPB3.3-MPI_reveal/LU
  ▽ bcast_inputs.f
      line 41
  ▽ blts.f
      line 63
      line 64
      line 65
      line 79
      line 80
      line 82
      line 103
      line 267
  ▽ buts.f
      line 63
      line 64
      line 65
      line 77
      line 78
      line 80
      line 97
      line 267
  ▽ erhs.f
      line 43
      line 44
      line 45
      line 46
      line 53
      line 54
      line 55
      line 57
      line 58
      line 60
      line 61
      line 105
      line 106
      line 107
```

```
         ! Directive inserted by Cray Reveal.  May be incomplete.
         !$OMP  parallel do default(none)
         !$OMP&   private (i,j,m)
         !$OMP&   shared  (k,omega,v,ldz,ist,iend,jst,jend)
S  63          do j = jst, jend
L  64             do i = ist, iend
V  65                do m = 1, 5
   66
   67                      v( m, i, j, k ) =  v( m, i, j, k )
   68         >     - omega * (  ldz( m, 1, i, j ) * v( 1, i, j, k-1 )
   69         >                + ldz( m, 2, i, j ) * v( 2, i, j, k-1 )
   70         >                + ldz( m, 3, i, j ) * v( 3, i, j, k-1 )
   71         >                + ldz( m, 4, i, j ) * v( 4, i, j, k-1 )
   72         >                + ldz( m, 5, i, j ) * v( 5, i, j, k-1 )  )
   73
   74                end do
   75             end do
   76          end do
   77
   78
S  79          do j=jst,jend
S  80            do i = ist, iend
   81
V  82                do m = 1, 5
   83
   84                      v( m, i, j, k ) =  v( m, i, j, k )
   85         > - omega * ( ldy( m, 1, i, j ) * v( 1, i, j-1, k )
   86         >         + ldx( m, 1, i, j ) * v( 1, i-1, j, k )
   87         >         + ldy( m, 2, i, j ) * v( 2, i, j-1, k )
   88         >         + ldx( m, 2, i, j ) * v( 2, i-1, j, k )
   89         >         + ldy( m, 3, i, j ) * v( 3, i, j-1, k )
   90         >         + ldx( m, 3, i, j ) * v( 3, i-1, j, k )
```

Info - Line 63

- 🟩 A loop starting at line 63 was scoped without errors.
- ● A loop starting at line 63 was not vectorized because it contains a call to a subroutine or function on line 64.
- ● A loop starting at line 64 was not vectorized because it contains a call to a subroutine or function on line 65.
- 🟩 A loop starting at line 65 was vectorized.

./LU/npb_lu.pl loaded. ./lu.C.128+pat+13333-24t.ap2 loaded.

Summary

- Craypat seems easy to use
- The user should be careful though
- Studying in detail the communication with Craypat is difficult
- Reveal tool could be really helpful
- Probably other tool(s) could be used for more detailed analysis

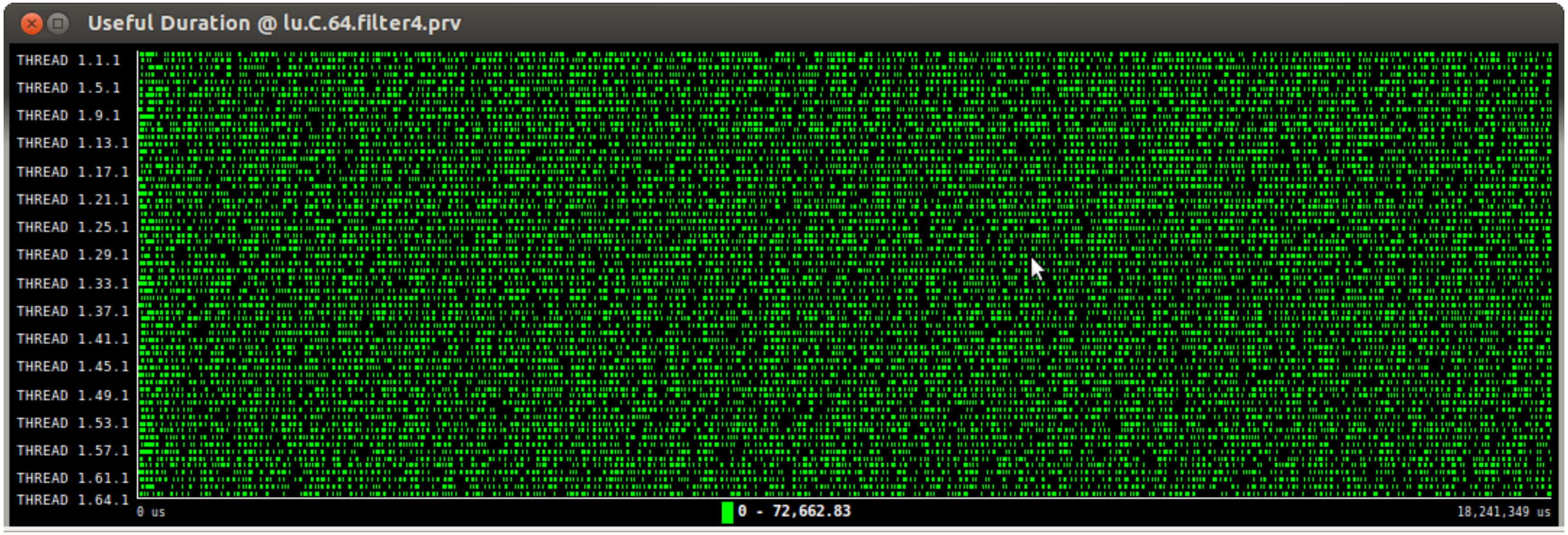
Extrae/Paraver

A profiling tool from Barcelona Supercomputing Center

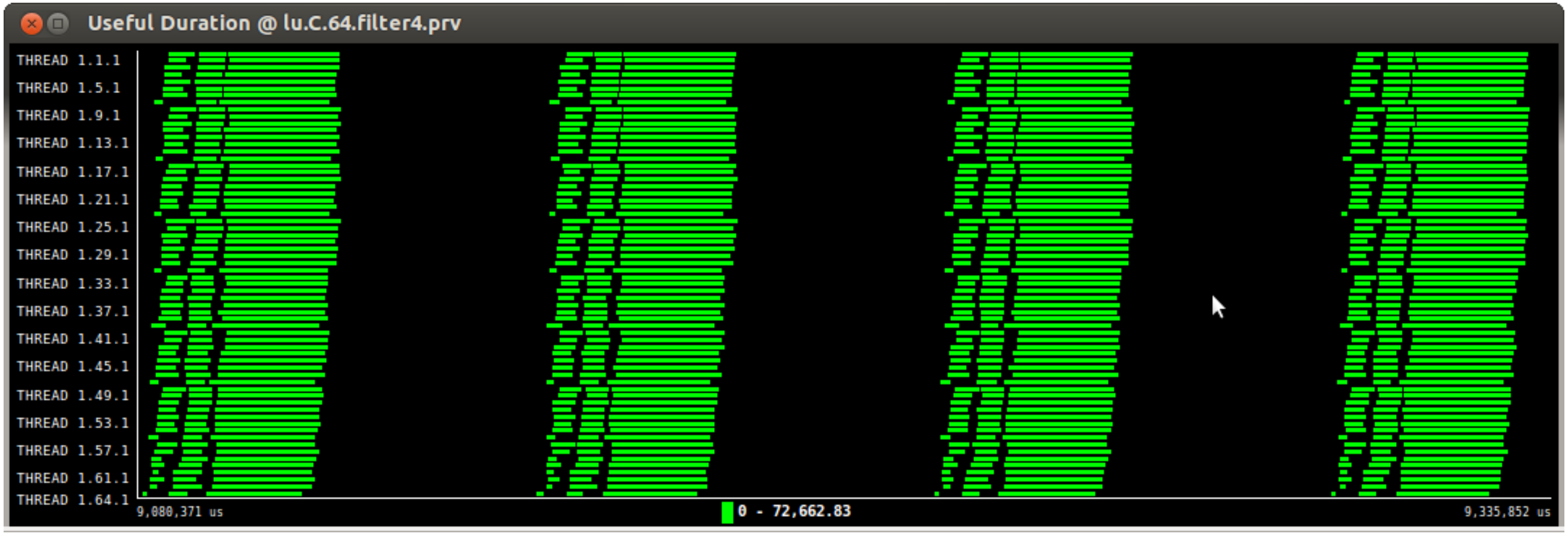
Extrae/Paraver (briefly)

- Instrumentation tool from Barcelona Supercomputing Center
- The main details are defined in an XML file
- For dynamic compilation a wrapper and LD_PRELOAD is enough
- For static compilation, linking is necessary
- Need to compile with at least -g option and -finstrument-functions for functions instrumentation with Intel and GNU compilers
- The trace for LU.C.64 is around to 5 GB
- Paraver is the tool to visualize and handle the traces from Extrae

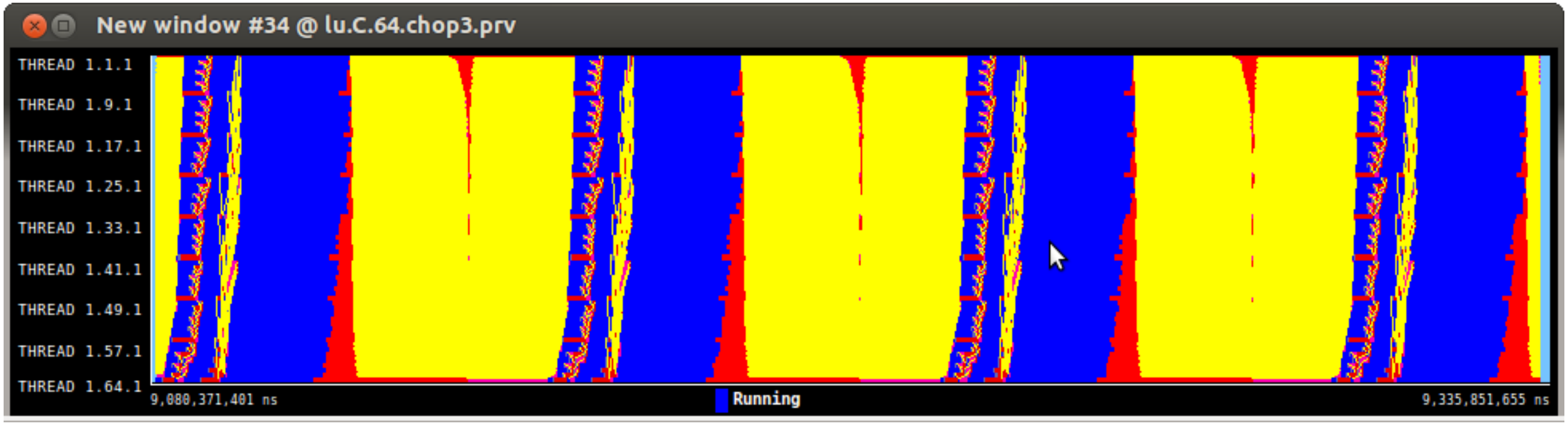
Paraver – Useful duration I



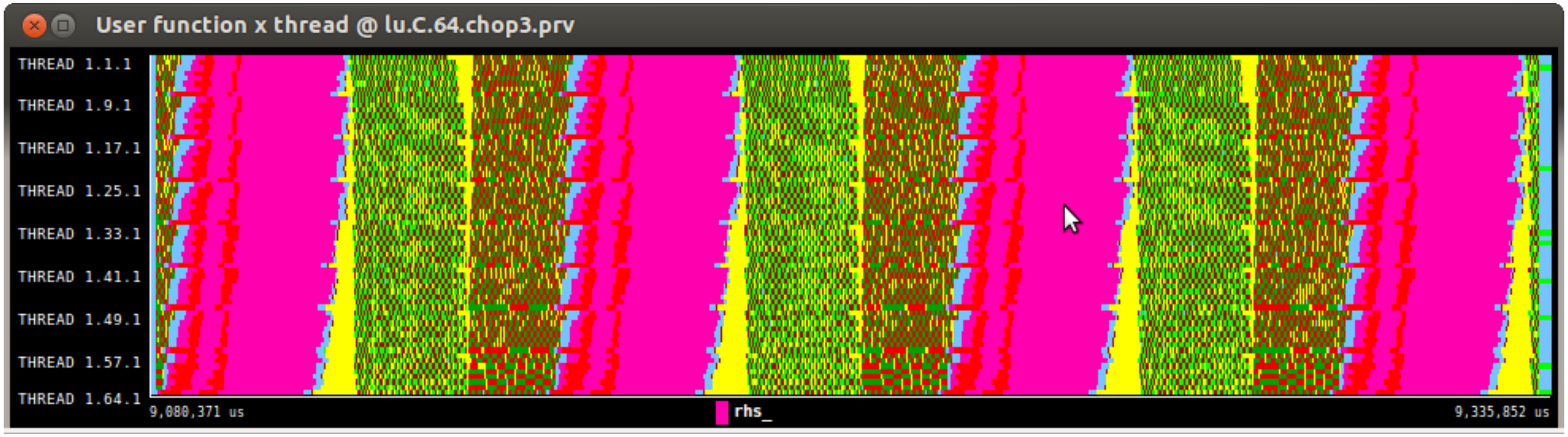
Paraver – Useful duration II - zoom



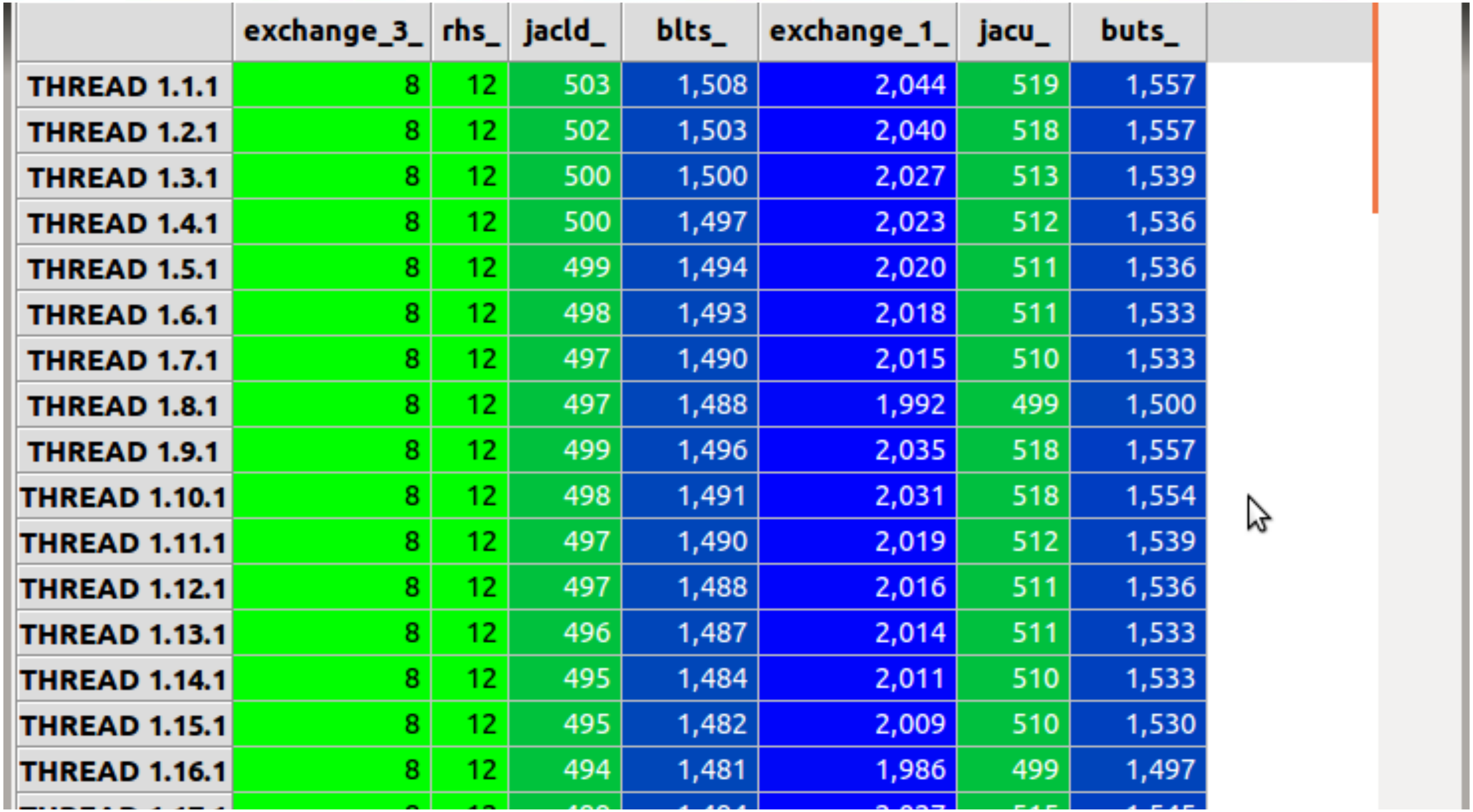
Paraver – Visualize events



Paraver – User functions

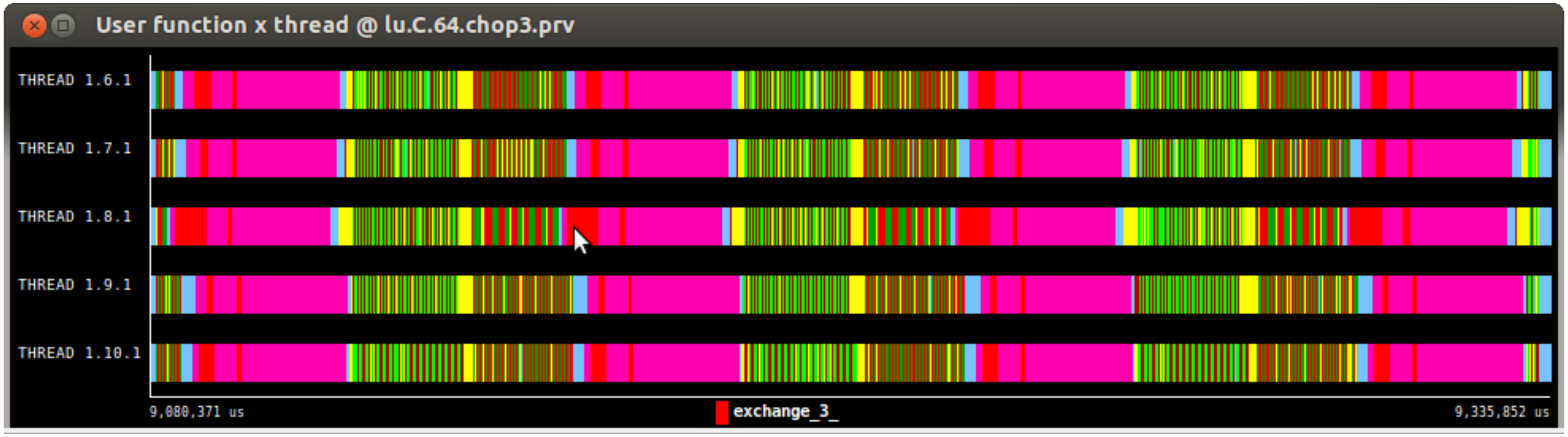


Paraver – User Functions Profile

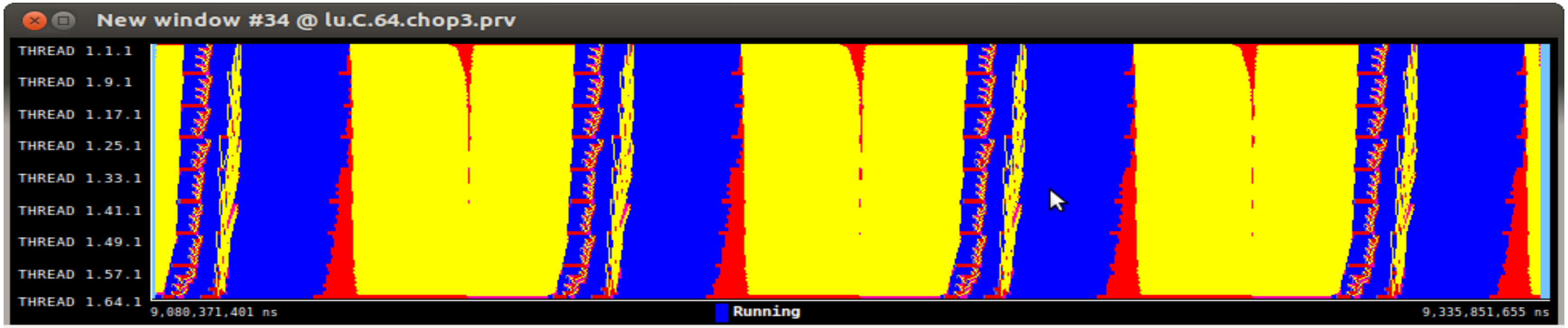


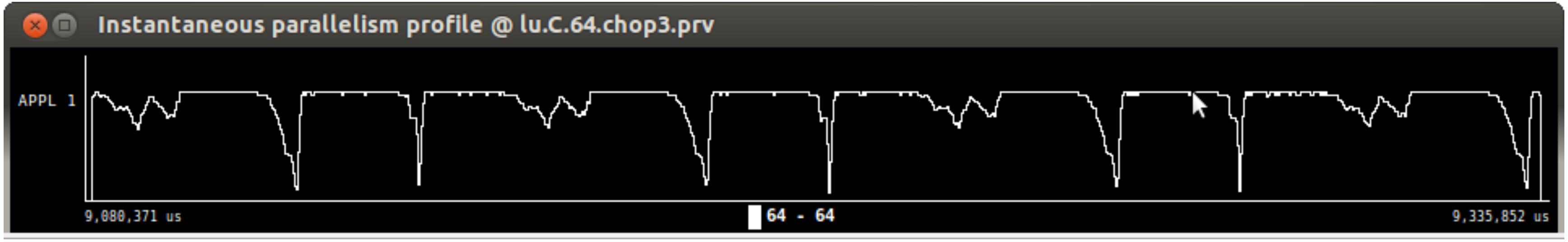
	exchange_3_	rhs_	jacld_	blts_	exchange_1_	jacu_	buts_
THREAD 1.1.1	8	12	503	1,508	2,044	519	1,557
THREAD 1.2.1	8	12	502	1,503	2,040	518	1,557
THREAD 1.3.1	8	12	500	1,500	2,027	513	1,539
THREAD 1.4.1	8	12	500	1,497	2,023	512	1,536
THREAD 1.5.1	8	12	499	1,494	2,020	511	1,536
THREAD 1.6.1	8	12	498	1,493	2,018	511	1,533
THREAD 1.7.1	8	12	497	1,490	2,015	510	1,533
THREAD 1.8.1	8	12	497	1,488	1,992	499	1,500
THREAD 1.9.1	8	12	499	1,496	2,035	518	1,557
THREAD 1.10.1	8	12	498	1,491	2,031	518	1,554
THREAD 1.11.1	8	12	497	1,490	2,019	512	1,539
THREAD 1.12.1	8	12	497	1,488	2,016	511	1,536
THREAD 1.13.1	8	12	496	1,487	2,014	511	1,533
THREAD 1.14.1	8	12	495	1,484	2,011	510	1,533
THREAD 1.15.1	8	12	495	1,482	2,009	510	1,530
THREAD 1.16.1	8	12	494	1,481	1,986	499	1,497

Paraver – Timeline selecting specific MPI processes

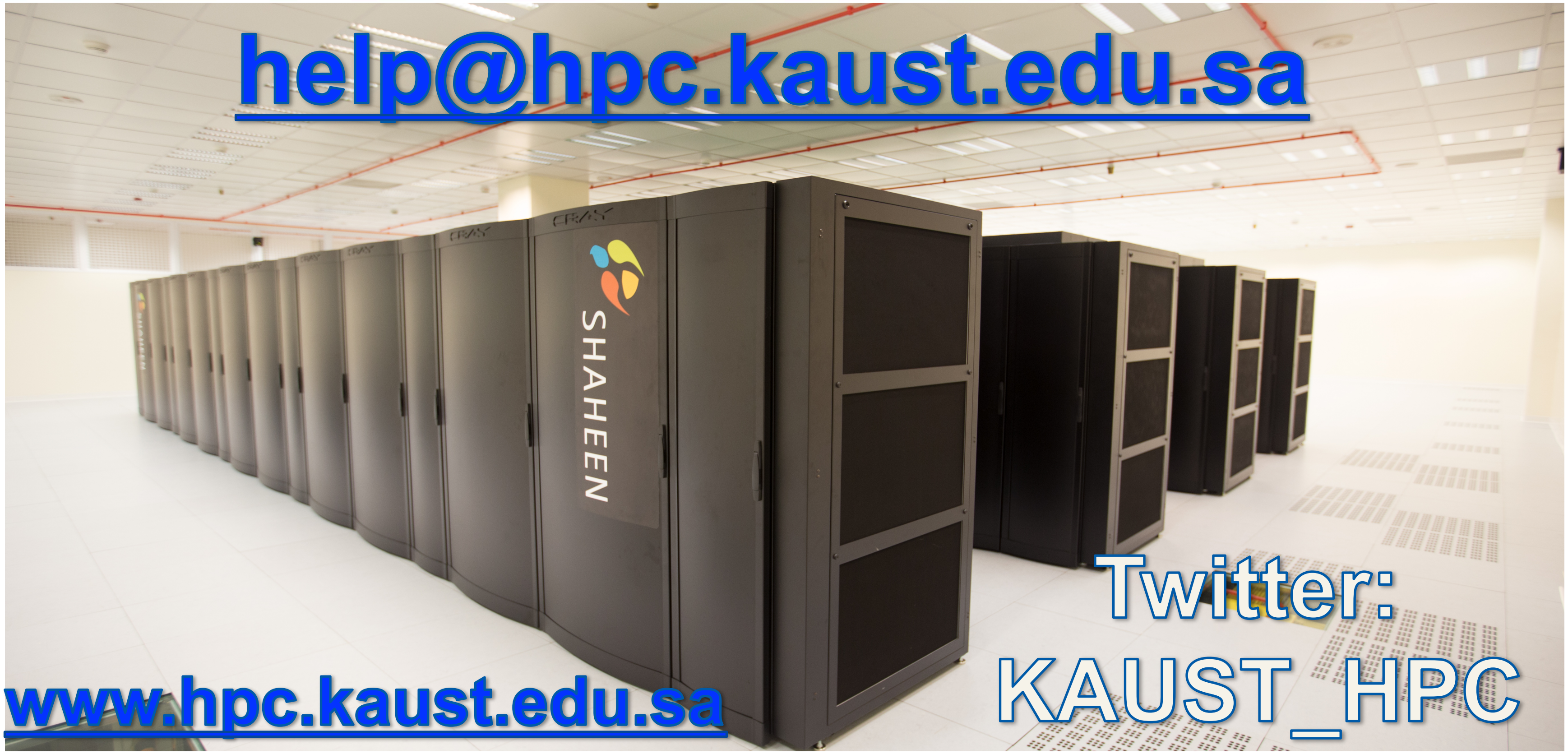


Paraver – Instantaneous parallelism profile





KAUST Supercomputing Laboratory



help@hpc.kaust.edu.sa

www.hpc.kaust.edu.sa

Twitter: KAUST_HPC

Thank You!