# Performance Measurement and Analysis Tools for Cray XE/XK Systems

Heidi Poxon
Cray Inc.

# Topics

- Analysis performed by the Cray performance tools
- Visualization of performance data

# Analysis Performed by the Cray Performance Tools

# Load Imbalance

# Motivation for Load Imbalance Analysis

- **Increasing system software and architecture complexity**
  - Current trend in high end computing is to have systems with tens of thousands of processors
    - This is being accentuated with multi-core processors

- **Applications have to be very well balanced In order to perform at scale on these MPP systems**
  - Efficient application scaling includes a balanced use of requested computing resources

- **Desire to minimize computing resource "waste"**
  - Identify slower paths through code
  - Identify inefficient "stalls" within an application

# MPI Sync Time

- Measure load imbalance in programs instrumented to trace MPI functions to determine if MPI ranks arrive at collectives together
- Separates potential load imbalance from data transfer
- Sync times reported by default if MPI functions traced
- If desired, PAT_RT_MPI_SYNC=0  deactivates this feature

# Imbalance Time

- Metric based on execution time
- It is dependent on the type of activity:
  - User functions

    **Imbalance time = Maximum time – Average time**
  - Synchronization (Collective communication and barriers)

    **Imbalance time = Average time – Minimum time**
- 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

# Imbalance %

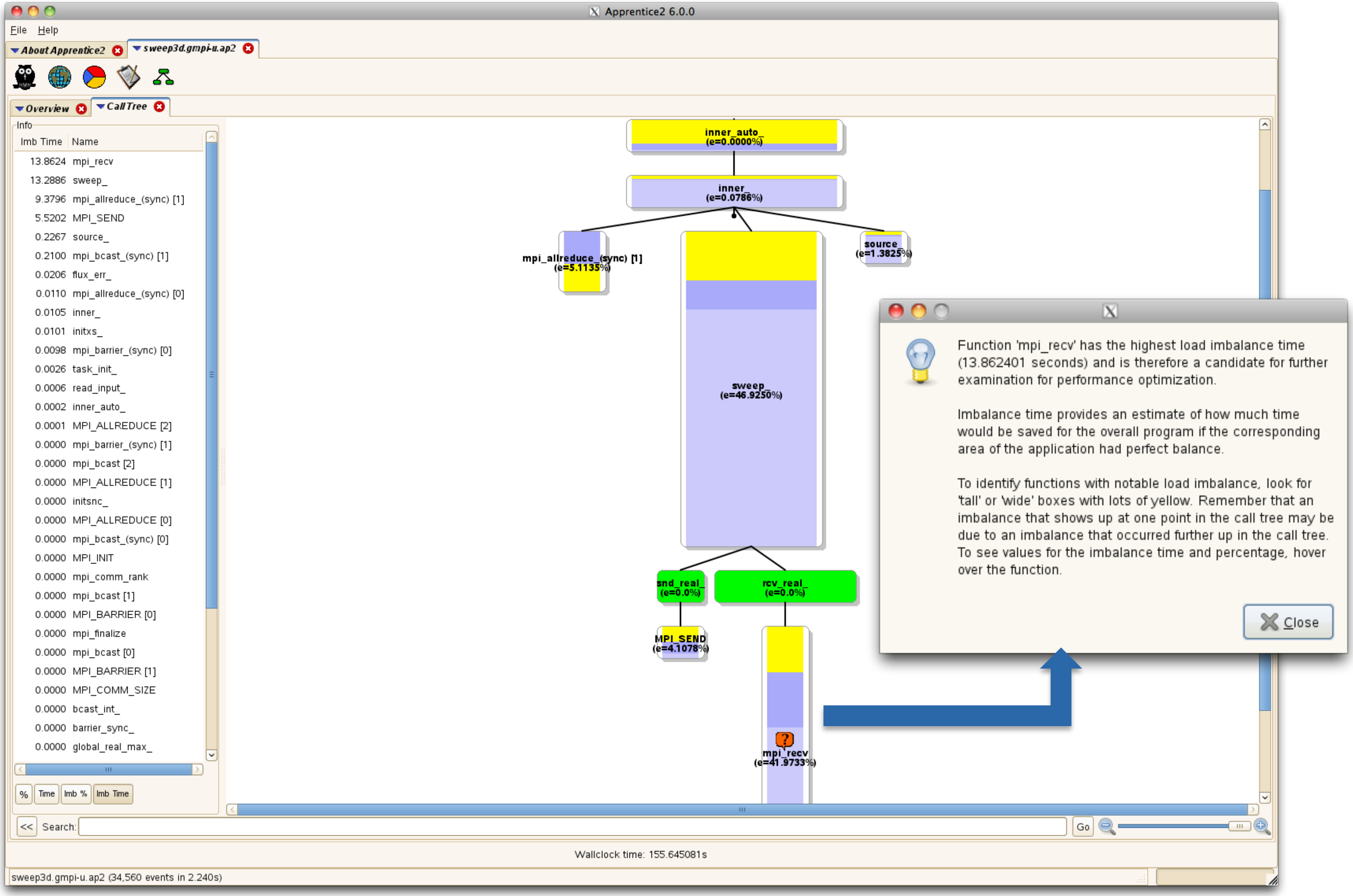
$$\text{Imbalance\%} = 100 \times \frac{\text{Imbalance time}}{\text{Max Time}} \times \frac{N}{N - 1}$$

- Represents % of resources available for parallelism that is “wasted”
- Corresponds to % of time that rest of team is not engaged in useful work on the given function
- Perfectly balanced code segment has imbalance of 0%
- Serial code segment has imbalance of 100%

# Load Imbalance Example in Sampling

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

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

 100.0% | 120.2 |    -- |     -- |Total
|-------------------------------------------------------------------------
|  99.9% | 120.0 |    -- |     -- |USER
||------------------------------------------------------------------------
||  91.2% | 109.6 |    -- |     -- |himenobmtxp_
3|        |       |       |        | himeno/himeno/ACC_CAF/himeno_caf_acc.f08
4|  91.2% | 109.6 |  77.4 |  41.6% |  line.226
||   8.6% |  10.3 |    -- |     -- |jacobi_
3|        |       |       |        | himeno/himeno/ACC_CAF/himeno_caf_acc.f08
4|   4.5% |   5.5 |   5.5 |  50.6% |  line.382
||========================================================================
|   0.1% |   0.2 |    -- |     -- |ETC
|=========================================================================
```

# Call Tree with Discrete Unit of Help



| Imb Time | Name |
|---|---|
| 13.8624 | mpi_recv |
| 13.2886 | sweep_ |
| 9.3796 | mpi_allreduce_(sync) [1] |
| 5.5202 | MPI_SEND |
| 0.2267 | source_ |
| 0.2100 | mpi_bcast_(sync) [1] |
| 0.0206 | flux_err_ |
| 0.0110 | mpi_allreduce_(sync) [0] |
| 0.0105 | inner_ |
| 0.0101 | initxs_ |
| 0.0098 | mpi_barrier_(sync) [0] |
| 0.0026 | task_init_ |
| 0.0006 | read_input_ |
| 0.0002 | inner_auto_ |
| 0.0001 | MPI_ALLREDUCE [2] |
| 0.0000 | mpi_barrier_(sync) [1] |
| 0.0000 | mpi_bcast [2] |
| 0.0000 | MPI_ALLREDUCE [1] |
| 0.0000 | initsnc_ |
| 0.0000 | MPI_ALLREDUCE [0] |
| 0.0000 | mpi_bcast_(sync) [0] |
| 0.0000 | MPI_INIT |
| 0.0000 | mpi_comm_rank |
| 0.0000 | mpi_bcast [1] |
| 0.0000 | MPI_BARRIER [0] |
| 0.0000 | mpi_finalize |
| 0.0000 | mpi_bcast [0] |
| 0.0000 | MPI_BARRIER [1] |
| 0.0000 | MPI_COMM_SIZE |
| 0.0000 | bcast_int_ |
| 0.0000 | barrier_sync_ |
| 0.0000 | global_real_max_ |

Function 'mpi_recv' has the highest load imbalance time (13.862401 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.

Wallclock time: 155.645081s

sweep3d.gmpi-u.ap2 (34,560 events in 2.240s)

# Cache Utilization

# Observations and Suggestions

**The performance tools provide additional automatic HW counter analysis and observations for:**

- **TLB utilization**
  - Measures how well the memory hierarchy is being utilized with regards to TLB
  - Depends on computation being single precision or double precision
  - Poor utilization indicates that not all entries on the page are being utilized between 2 TLB misses
- **cache utilization**
  - Poor utilization indicates that not all entries on the cache line are being utilized between 2 cache misses
- **D1 cache hit (or miss) ratios**
- **D1+D2 cache hit (or miss) ratios**

# Example Cache Threshold Observations

```
================  Observations and suggestions  ========================
D1 cache utilization:
    61.7% of total execution time was spent in 1 functions with D1 cache
    hit ratios below the desirable minimum of 90.0%. Cache utilization
    might be improved by modifying the alignment or stride of references
    to data arrays in these functions.

        D1     Time%   Function
     cache
       hit
     ratio

       74.3%    61.7%  calc3_

D1 + D2 cache utilization:
    61.7% of total execution time was spent in 1 functions with combined
    D1 and D2 cache hit ratios below the desirable minimum of 97.0%.
    Cache utilization might be improved by modifying the alignment or
    stride of references to data arrays in these functions.

     D1+D2     Time%   Function
     cache
       hit
     ratio

       96.6%    61.7%  calc3_

…
```

# Example Cache Threshold Observations (2)

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

TLB utilization:
    82.5% of total execution time was spent in 2 functions with fewer
    than the desirable minimum of 512 data references per TLB miss. TLB
    utilization might be improved by modifying the alignment or stride
    of references to data arrays in these functions.

        LS    Time%   Function
       per
       TLB
        DM

        3.97    61.7%  calc3_
      163.77    20.8%  calc2_
================  End Observations  ====================================
```

# View Profile Data with pat_report

# pat_report: Job Execution Information

```
CrayPat/X:  Version 5.2.3.8078 Revision 8078 (xf 8063)  08/25/11 …

Number of PEs (MPI ranks):   16

Numbers of PEs per Node:     16

Numbers of Threads per PE:    1

Number of Cores per Socket:  12

Execution start time:  Thu Aug 25 14:16:51 2011

System type and speed:  x86_64 2000 MHz

Current path to data file:
  /lus/scratch/heidi/ted_swim/mpi-openmp/run/swim+pat+27472-34t.ap2

Notes for table 1:
…
```

# pat_report: Table Notes

```
Notes for table 1:

  Table option:
    -O profile
  Options implied by table option:
    -d ti%@0.95,ti,imb_ti,imb_ti%,tr -b gr,fu,pe=HIDE
  Other options:
    -T

  Options for related tables:
    -O profile_pe.th              -O profile_th_pe
    -O profile+src                -O load_balance
    -O callers                    -O callers+src
    -O calltree                   -O calltree+src

  The Total value for Time, Calls is the sum for the Group values.
  The Group value for Time, Calls is the sum for the Function values.
  The Function value for Time, Calls is the avg for the PE values.
    (To specify different aggregations, see: pat_help report options s1)

  This table shows only lines with Time% > 0.

  Percentages at each level are of the Total for the program.
    (For percentages relative to next level up, specify:
      -s percent=r[elative])
```

# pat_report: Additional Information

```
...
Instrumented with:
  pat_build -gmpi -u himenoBMTxpr.x

Program invocation:
  ../bin/himenoBMTxpr+pat.x

Exit Status:  0 for 256 PEs

CPU  Family: 15h  Model: 01h  Stepping: 2

Core Performance Boost:  Configured for   0 PEs
                         Capable    for 256 PEs

Memory pagesize:  4096

Accelerator Model: Nvidia X2090 Memory: 6.00 GB Frequency: 1.15 GHz

Programming environment:  CRAY

Runtime environment variables:
  OMP_NUM_THREADS=1
...
```

# Sampling Output (Table 1)

```
Notes for table 1:
...
Table 1:  Profile by Function

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

 100.0% |  775 |    -- |     -- |Total
|-------------------------------------------
|  94.2% |  730 |    -- |     -- |USER
||------------------------------------------
||  43.4% |  336 |  8.75 |   2.6% |mlwxyz_
||  16.1% |  125 |  6.28 |   4.9% |half_
||   8.0% |   62 |  6.25 |   9.5% |full_
||   6.8% |   53 |  1.88 |   3.5% |artv_
||   4.9% |   38 |  1.34 |   3.6% |bnd_
||   3.6% |   28 |  2.00 |   6.9% |currenf_
||   2.2% |   17 |  1.50 |   8.6% |bndsf_
||   1.7% |   13 |  1.97 |  13.5% |model_
||   1.4% |   11 |  1.53 |  12.2% |cfl_
||   1.3% |   10 |  0.75 |   7.0% |currenh_
||   1.0% |    8 |  5.28 |  41.9% |bndbo_
||   1.0% |    8 |  8.28 |  53.4% |bndto_
||==========================================
|   5.4% |   42 |    -- |     -- |MPI
||------------------------------------------
||   1.9% |   15 |  4.62 |  23.9% |mpi_sendrecv_
||   1.8% |   14 | 16.53 |  55.0% |mpi_bcast_
||   1.7% |   13 |  5.66 |  30.7% |mpi_barrier_
|===========================================
```

# pat_report: Flat Profile

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

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

 100.0% | 104.593634 |       -- |     -- | 22649 |Total
|-------------------------------------------------------------
|  71.0% |  74.230520 |       -- |     -- | 10473 |MPI
||------------------------------------------------------------
||  69.7% |  72.905208 | 0.508369 |   0.7% |   125 |mpi_allreduce_
||   1.0% |   1.050931 | 0.030042 |   2.8% |    94 |mpi_alltoall_
||============================================================
|  25.3% |  26.514029 |       -- |     -- |    73 |USER
||------------------------------------------------------------
||  16.7% |  17.461110 | 0.329532 |   1.9% |    23 |selfgravity_
||   7.7% |   8.078474 | 0.114913 |   1.4% |    48 |ffte4_
||============================================================
|   2.5% |   2.659429 |       -- |     -- |   435 |MPI_SYNC
||------------------------------------------------------------
||   2.1% |   2.207467 | 0.768347 |  26.2% |   172 |mpi_barrier_(sync)
||============================================================
|   1.1% |   1.188998 |       -- |     -- | 11608 |HEAP
||------------------------------------------------------------
||   1.1% |   1.166707 | 0.142473 |  11.1% |  5235 |free
|=============================================================
```

# pat_report: Message Stats by Caller

```
Table 4:  MPI Message Stats by Caller

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

 15138076.0 | 4099.4 |   411.6 | 3687.8 |Total
|------------------------------------------------
| 15138028.0 | 4093.4 |   405.6 | 3687.8 |MPI_ISEND
||-----------------------------------------------
||  8080500.0 | 2062.5 |    93.8 | 1968.8 |calc2_
3|            |        |         |        | MAIN_
||||---------------------------------------------
4|||  8216000.0 | 3000.0 | 1000.0 | 2000.0 |pe.0
4|||  8208000.0 | 2000.0 |     -- | 2000.0 |pe.9
4|||  6160000.0 | 2000.0 |  500.0 | 1500.0 |pe.15
||||=============================================
||  6285250.0 | 1656.2 |   125.0 | 1531.2 |calc1_
3|            |        |         |        | MAIN_
||||---------------------------------------------
4|||  8216000.0 | 3000.0 | 1000.0 | 2000.0 |pe.0
4|||  6156000.0 | 1500.0 |     -- | 1500.0 |pe.3
4|||  6156000.0 | 1500.0 |     -- | 1500.0 |pe.5
||||=============================================
. . .
```

# Profile Visualization with Cray Apprentice2

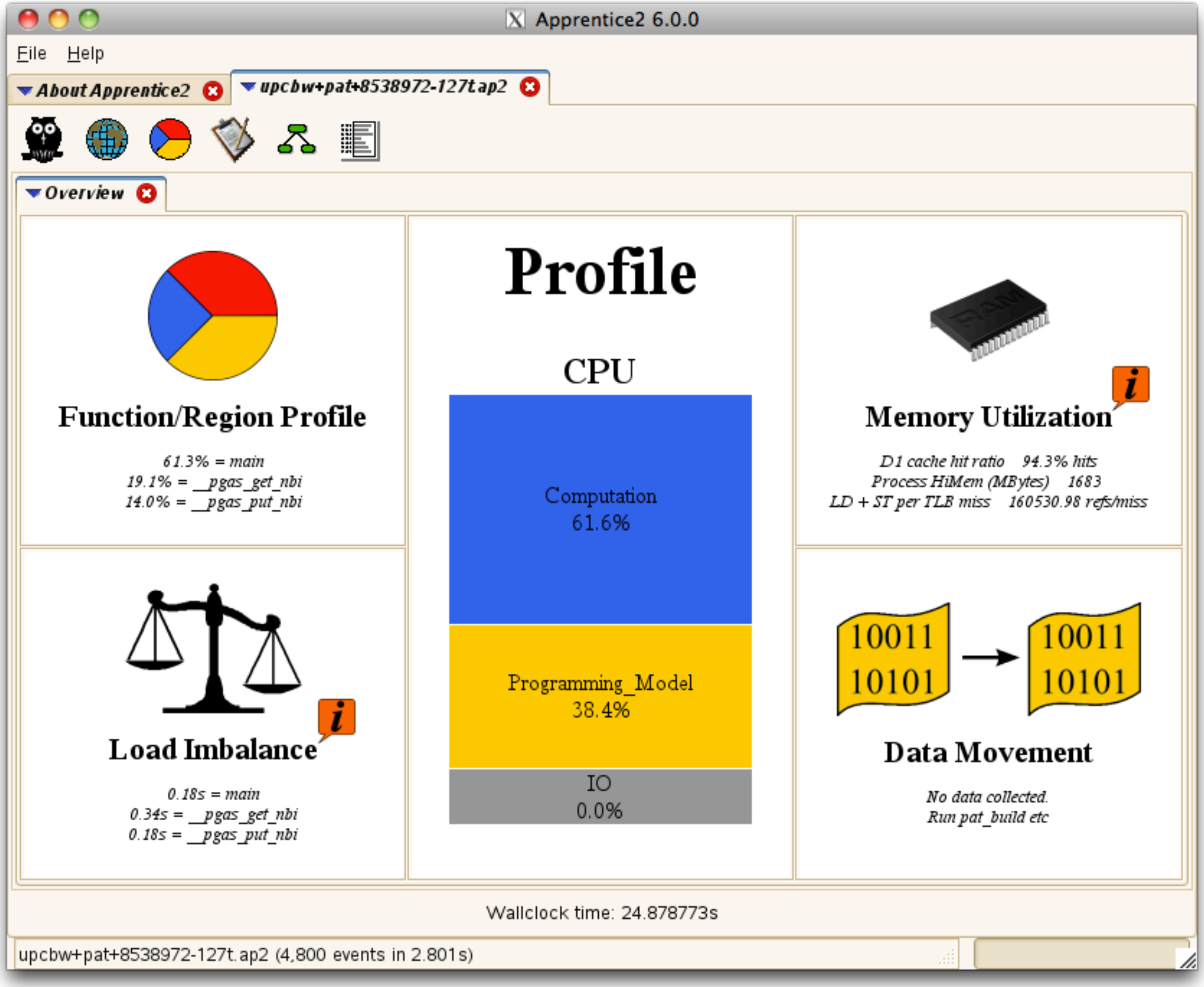
# Cray Apprentice$^2$

- Call graph profile
- Communication statistics
- Time-line view
  - Communication
  - I/O
- Activity view
- Pair-wise communication statistics
- Text reports
- Source code mapping

- Runs on login node
- Supported on Mac OS and Windows also

- Cray Apprentice$^2$ helps identify:
  - Load imbalance
  - Excessive communication
  - Network contention
  - Excessive serialization
  - I/O Problems

# Application Performance Summary



**Function/Region Profile**

61.3% = main
19.1% = __pgas_get_nbi
14.0% = __pgas_put_nbi

**Memory Utilization**

D1 cache hit ratio 94.3% hits
Process HiMem (MBytes) 1683
LD + ST per TLB miss 160530.98 refs/miss

**Load Imbalance**

0.18s = main
0.34s = __pgas_get_nbi
0.18s = __pgas_put_nbi

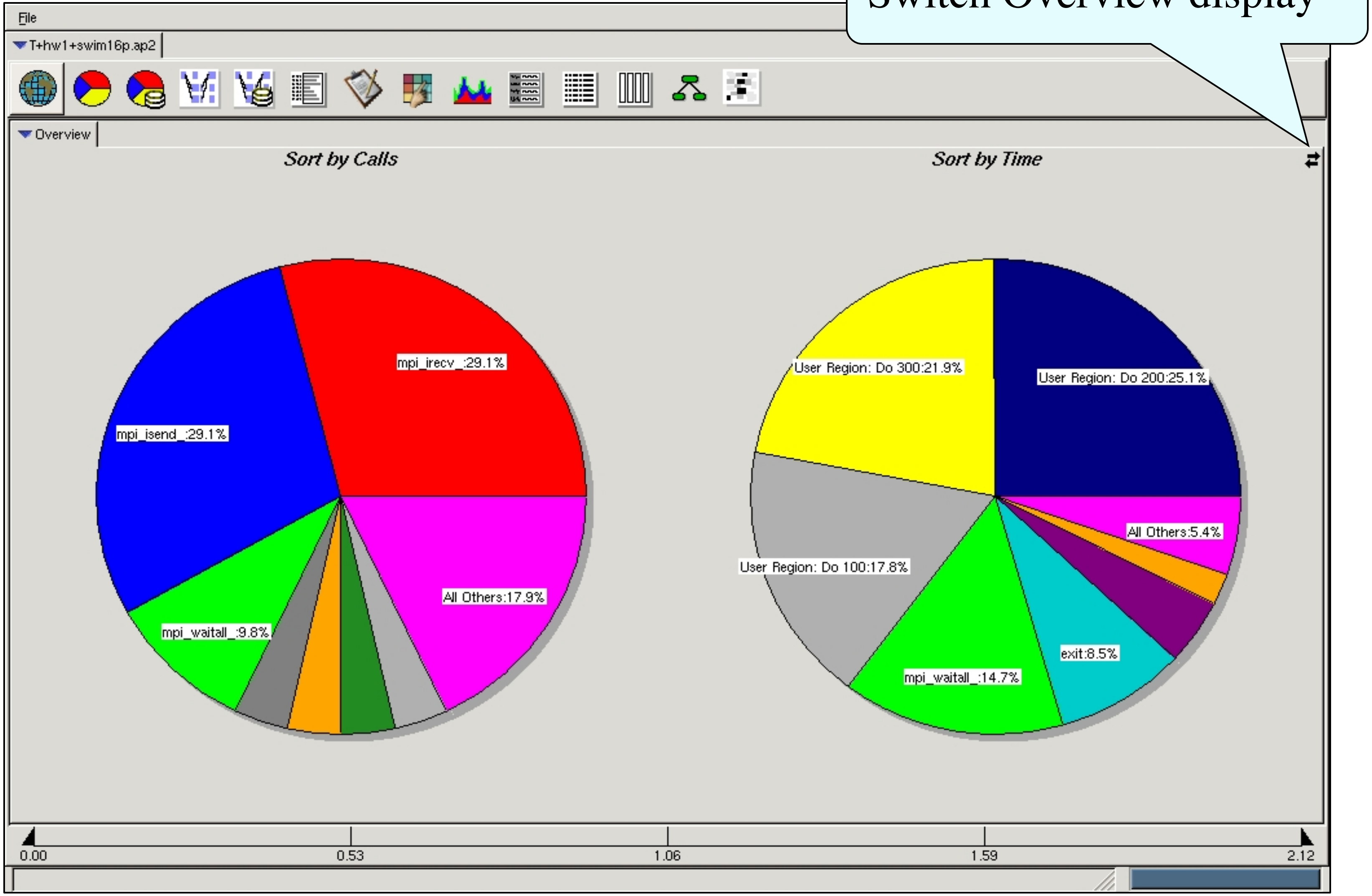
**Data Movement**

No data collected.
Run pat_build etc

Wallclock time: 24.878773s

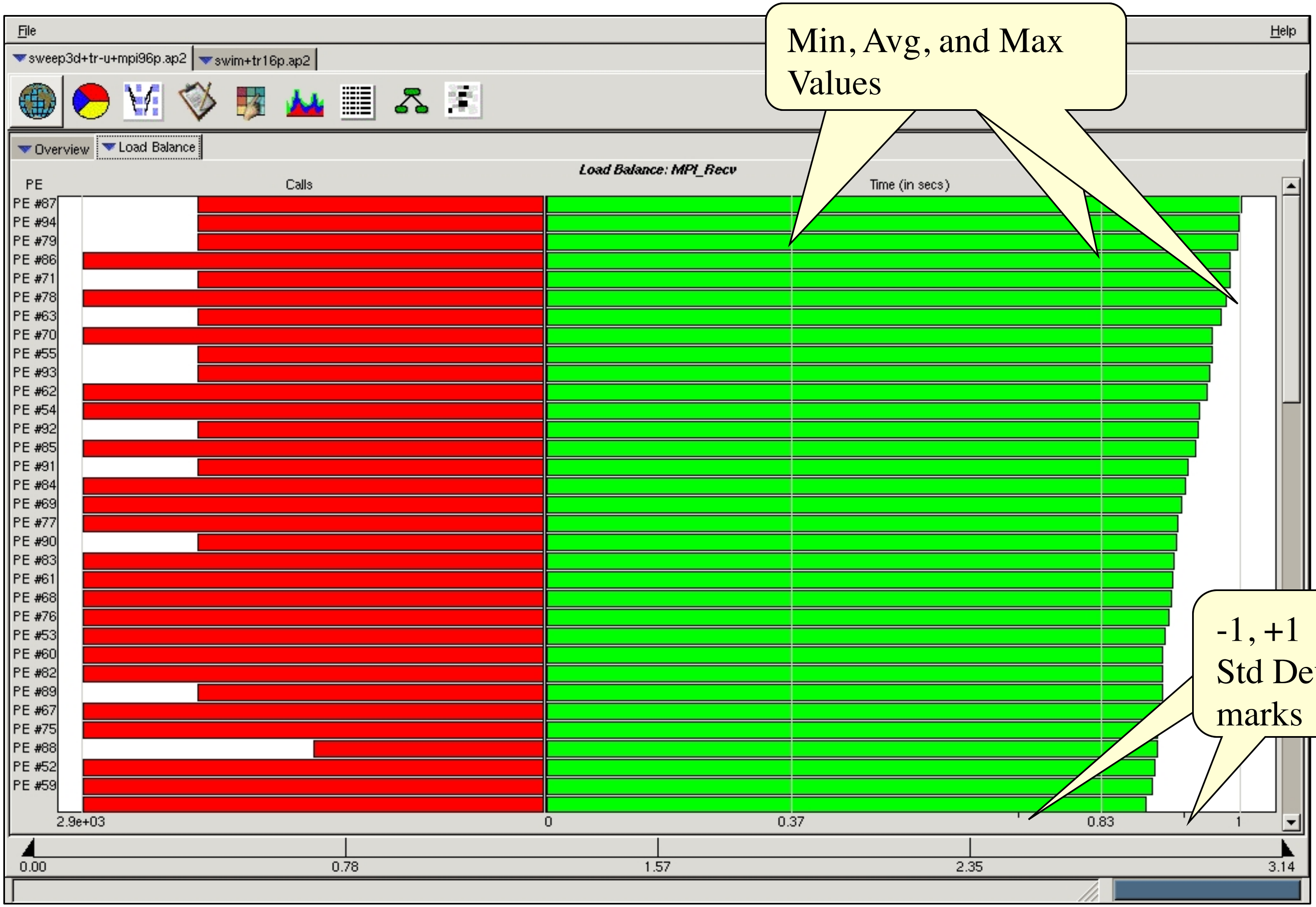
upcbw+pat+8538972-127t.ap2 (4,800 events in 2.801s)

# Statistics Overview



Switch Overview display

# Load Balance View (Aggregated from Overview)



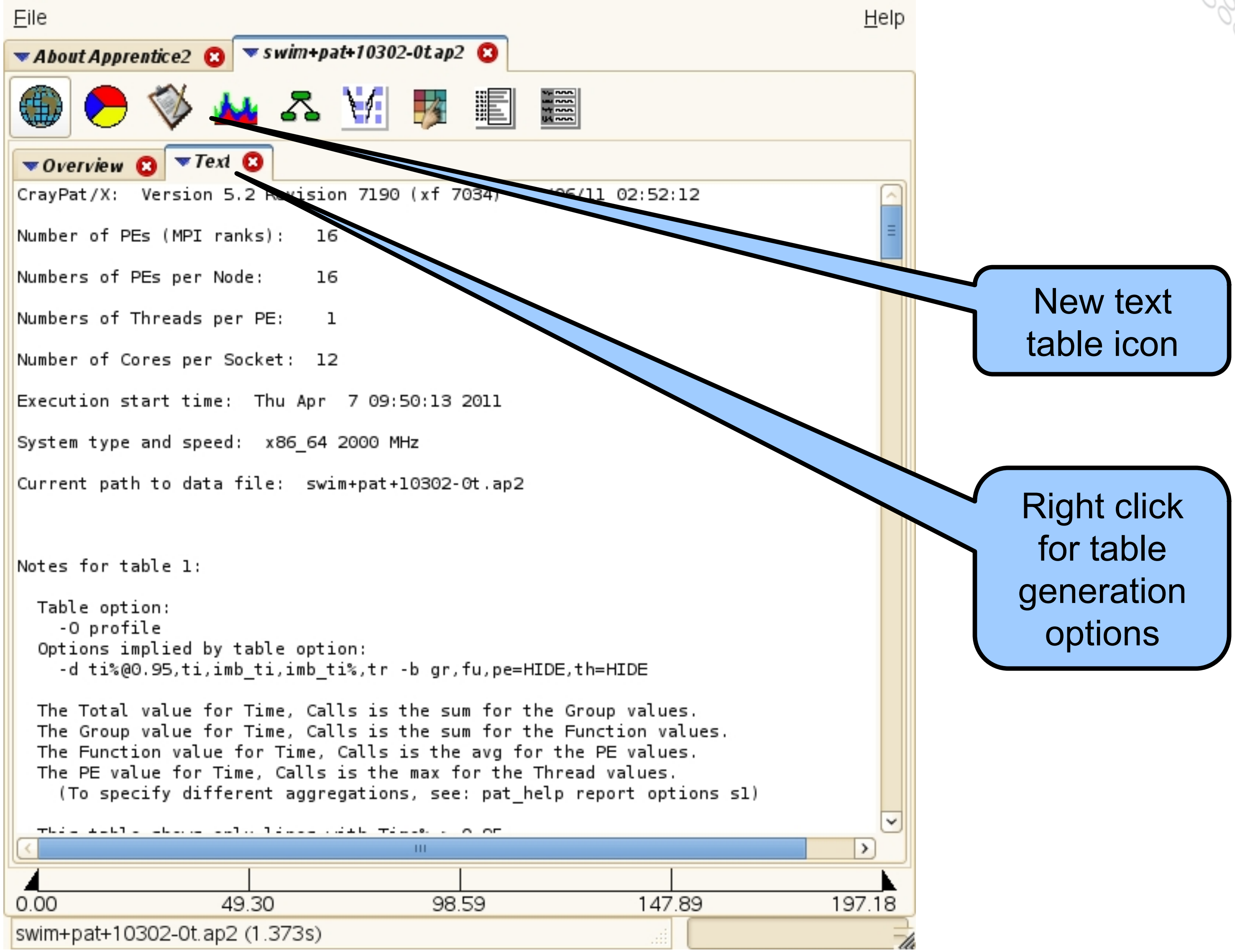
Min, Avg, and Max Values

-1, +1 Std Dev marks

# pat_report Tables in Cray Apprentice2

- Complimentary performance data available in one place
- Drop down menu provides quick access to most common reports
- Ability to easily generate different views of performance data
- Provides mechanism for more in depth explanation of data presented

# Example of pat_report Tables in Cray Apprentice2

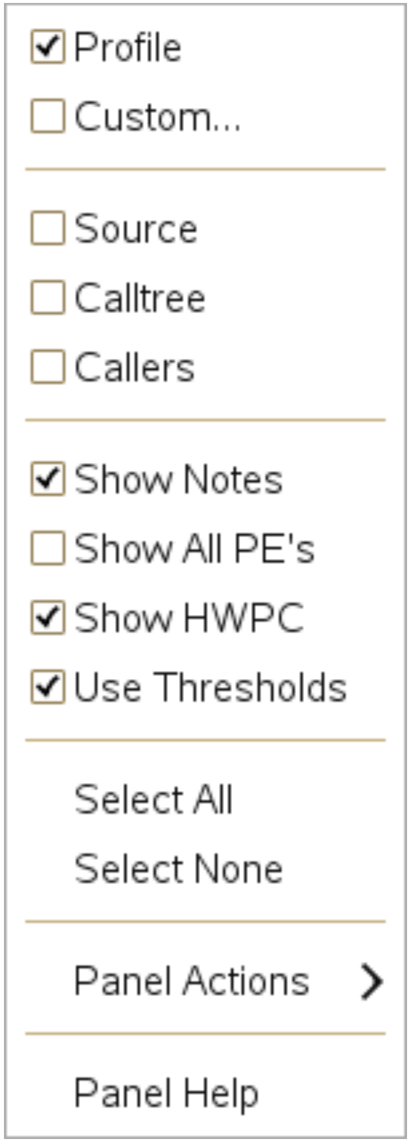


```
CrayPat/X:  Version 5.2 Revision 7190 (xf 7034)  /06/11 02:52:12

Number of PEs (MPI ranks):   16

Numbers of PEs per Node:     16

Numbers of Threads per PE:    1

Number of Cores per Socket:  12

Execution start time:  Thu Apr  7 09:50:13 2011

System type and speed:  x86_64 2000 MHz

Current path to data file:  swim+pat+10302-0t.ap2



Notes for table 1:

  Table option:
    -O profile
  Options implied by table option:
    -d ti%@0.95,ti,imb_ti,imb_ti%,tr -b gr,fu,pe=HIDE,th=HIDE

  The Total value for Time, Calls is the sum for the Group values.
  The Group value for Time, Calls is the sum for the Function values.
  The Function value for Time, Calls is the avg for the PE values.
  The PE value for Time, Calls is the max for the Thread values.
    (To specify different aggregations, see: pat_help report options s1)

  This table shows only lines with Time% > 0.05
```

New text table icon

Right click for table generation options

# Generating New pat_report Tables



☑ Profile
□ Custom...

□ Source
□ Calltree
□ Callers

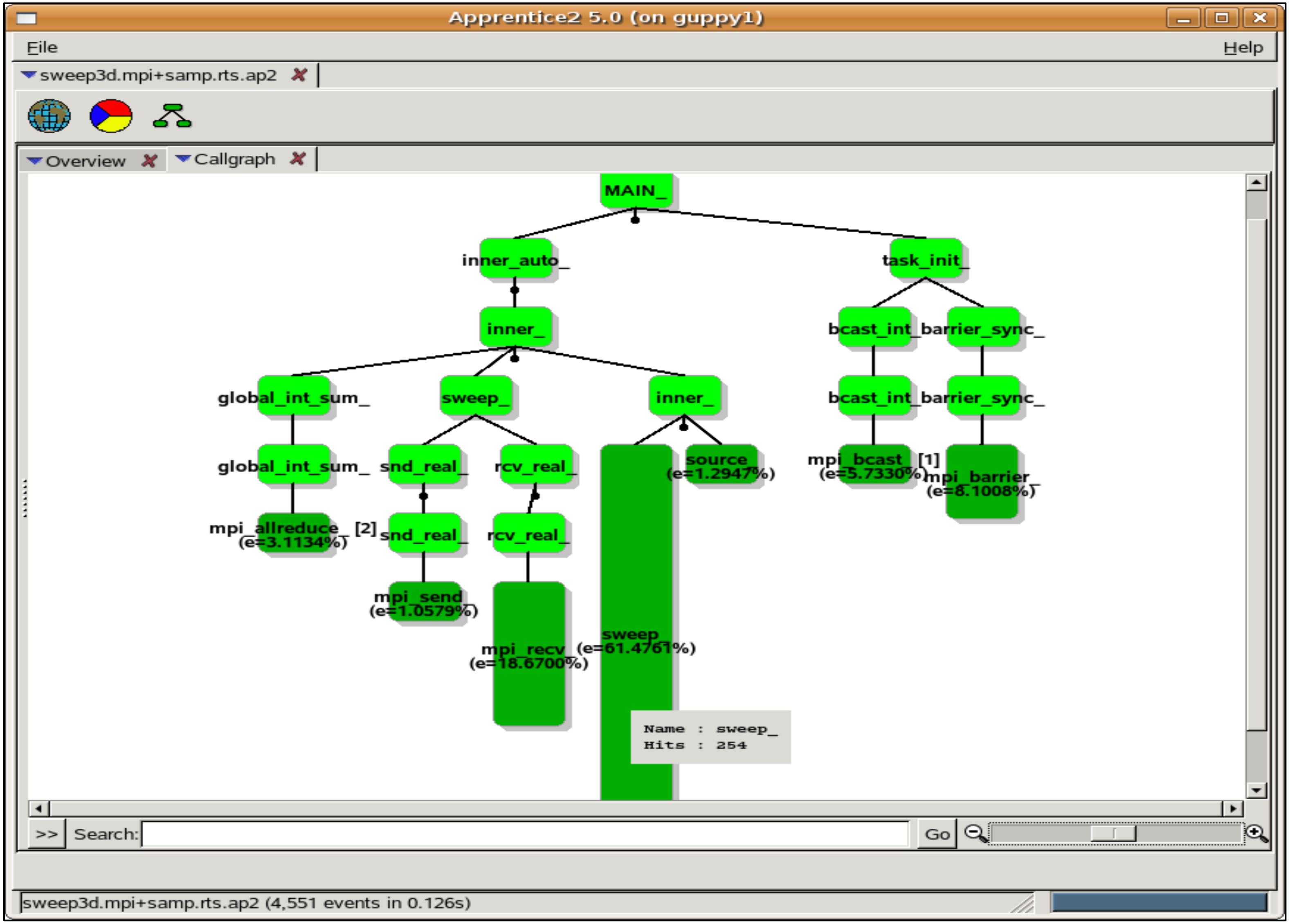
☑ Show Notes
□ Show All PE's
☑ Show HWPC
☑ Use Thresholds

Select All
Select None

Panel Actions >

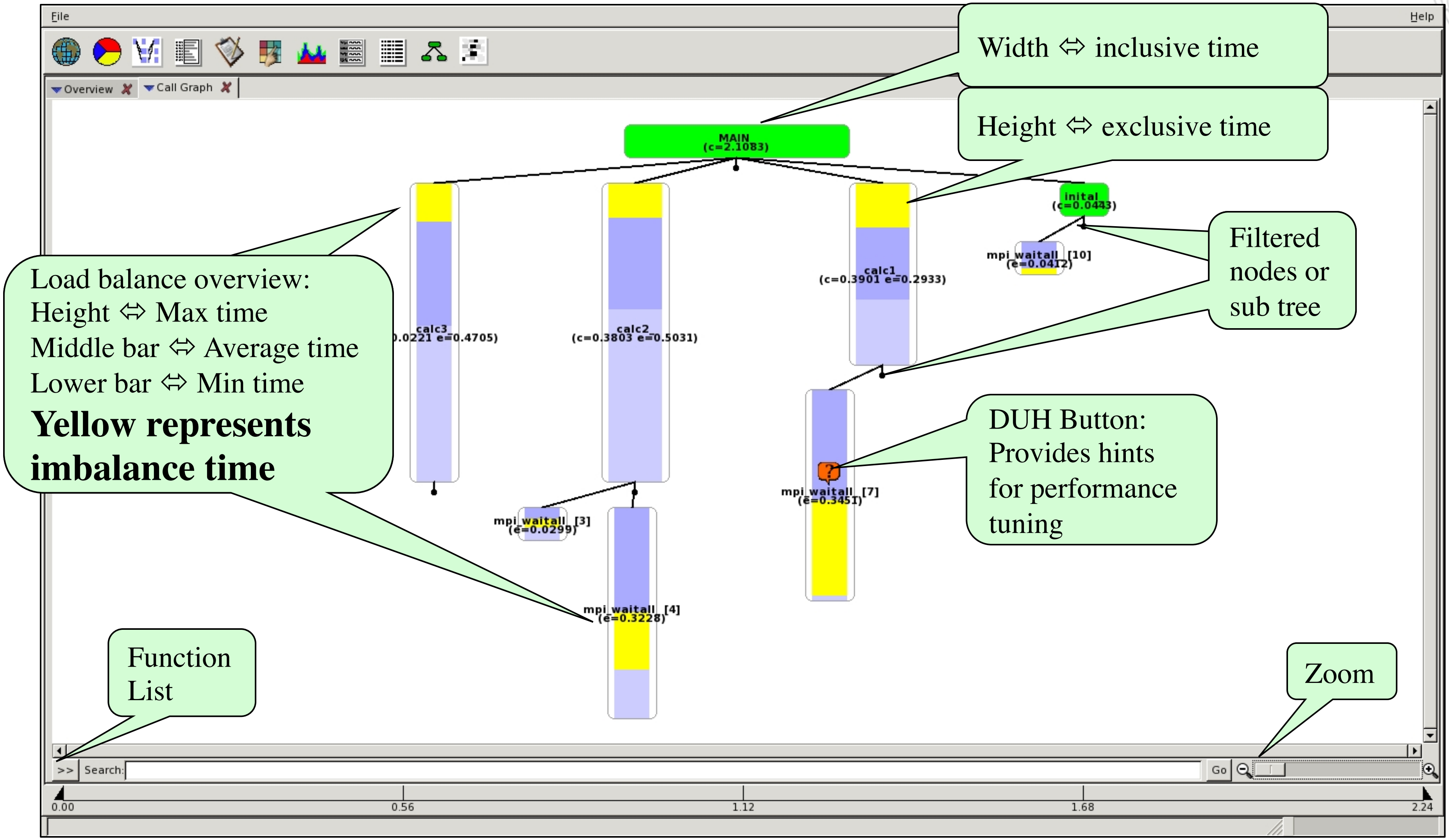
Panel Help

# Apprentice$^2$ Call Tree View of Sampled Data



sweep3d.mpi+samp.rts.ap2 (4,551 events in 0.126s)

# Call Tree View



Width ⇔ inclusive time

Height ⇔ exclusive time

Filtered nodes or sub tree

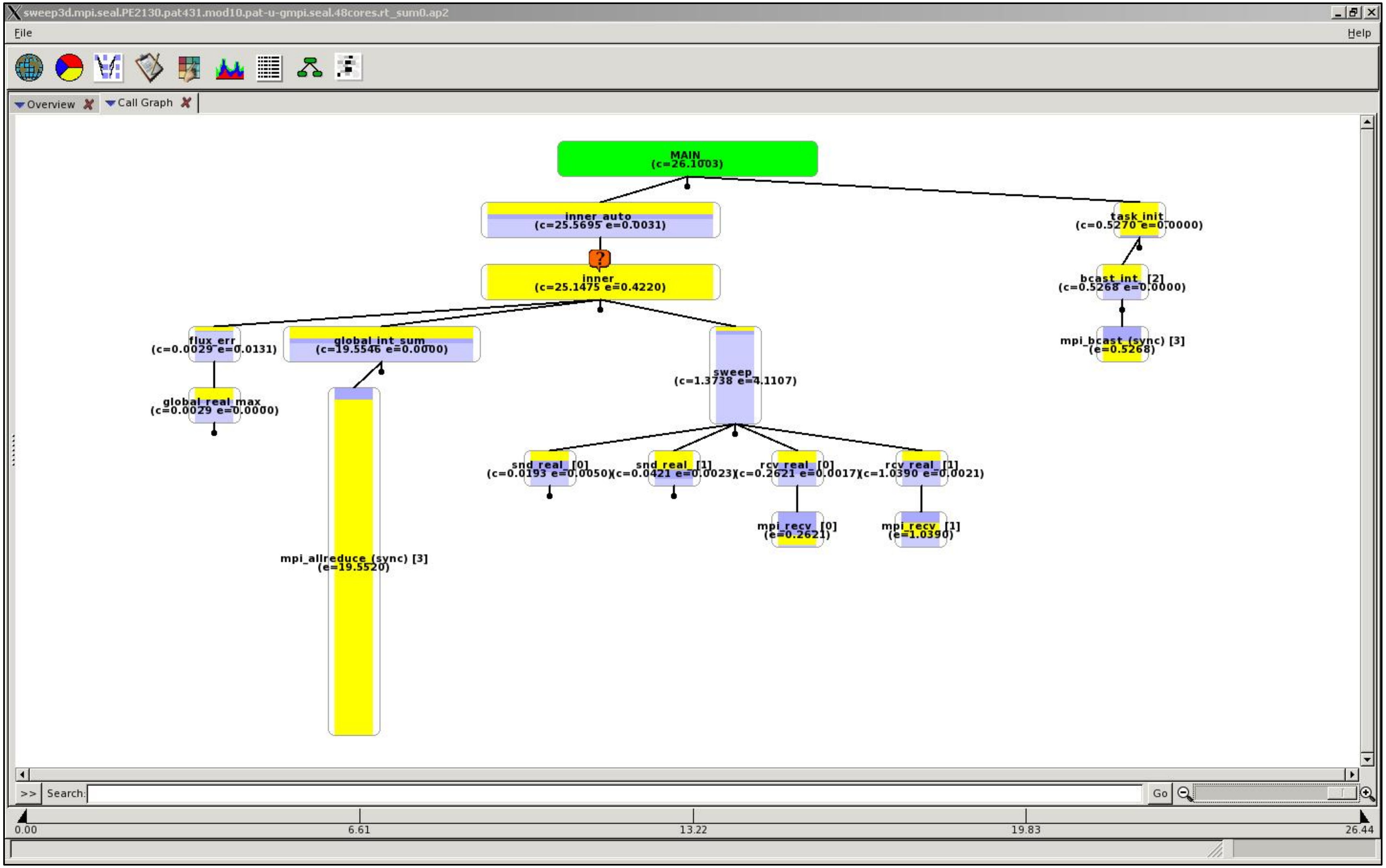
Load balance overview:
Height ⇔ Max time
Middle bar ⇔ Average time
Lower bar ⇔ Min time
**Yellow represents imbalance time**

DUH Button: Provides hints for performance tuning

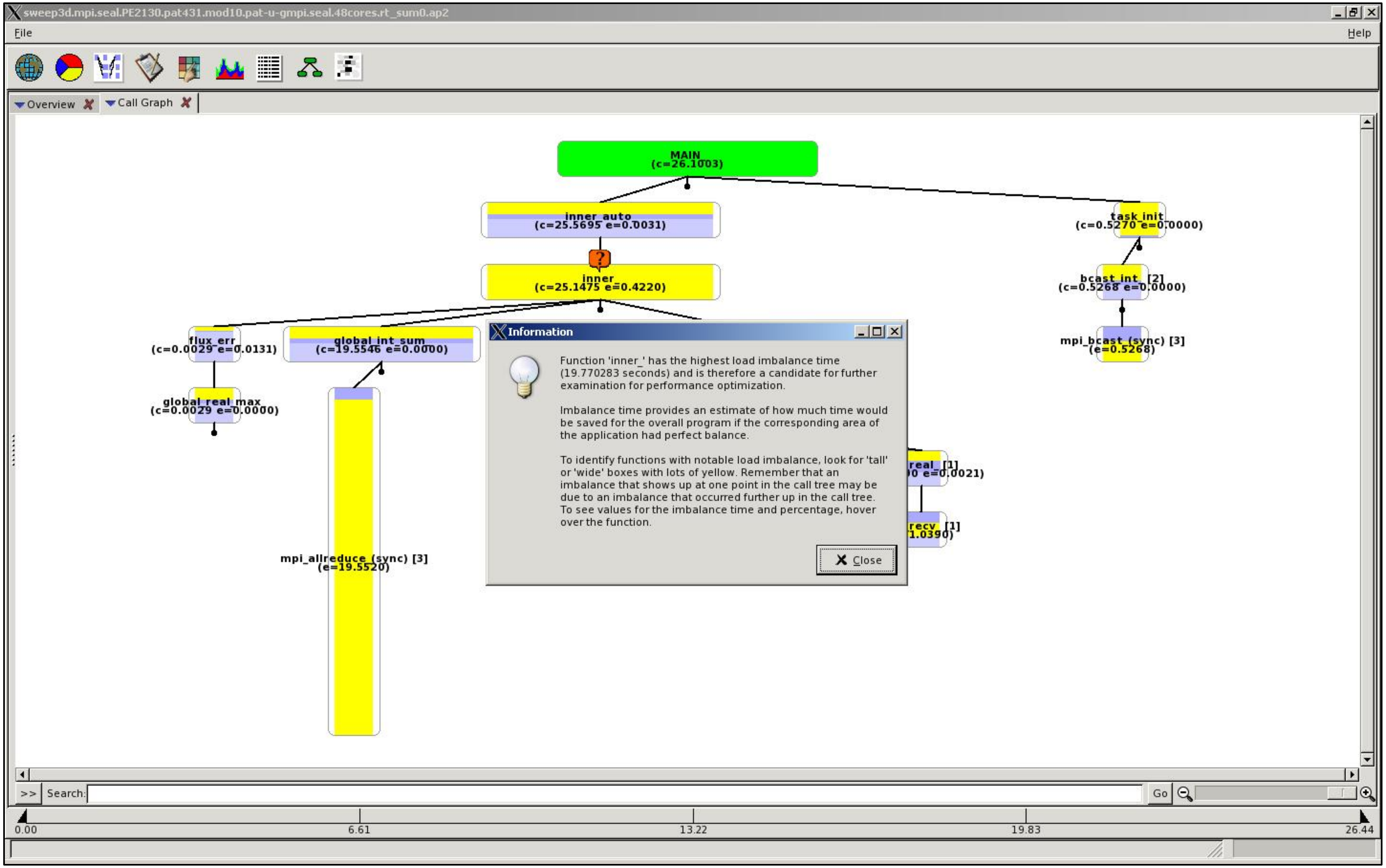
Function List

Zoom

# Call Tree Visualization



# Discrete Unit of Help (DUH Button)

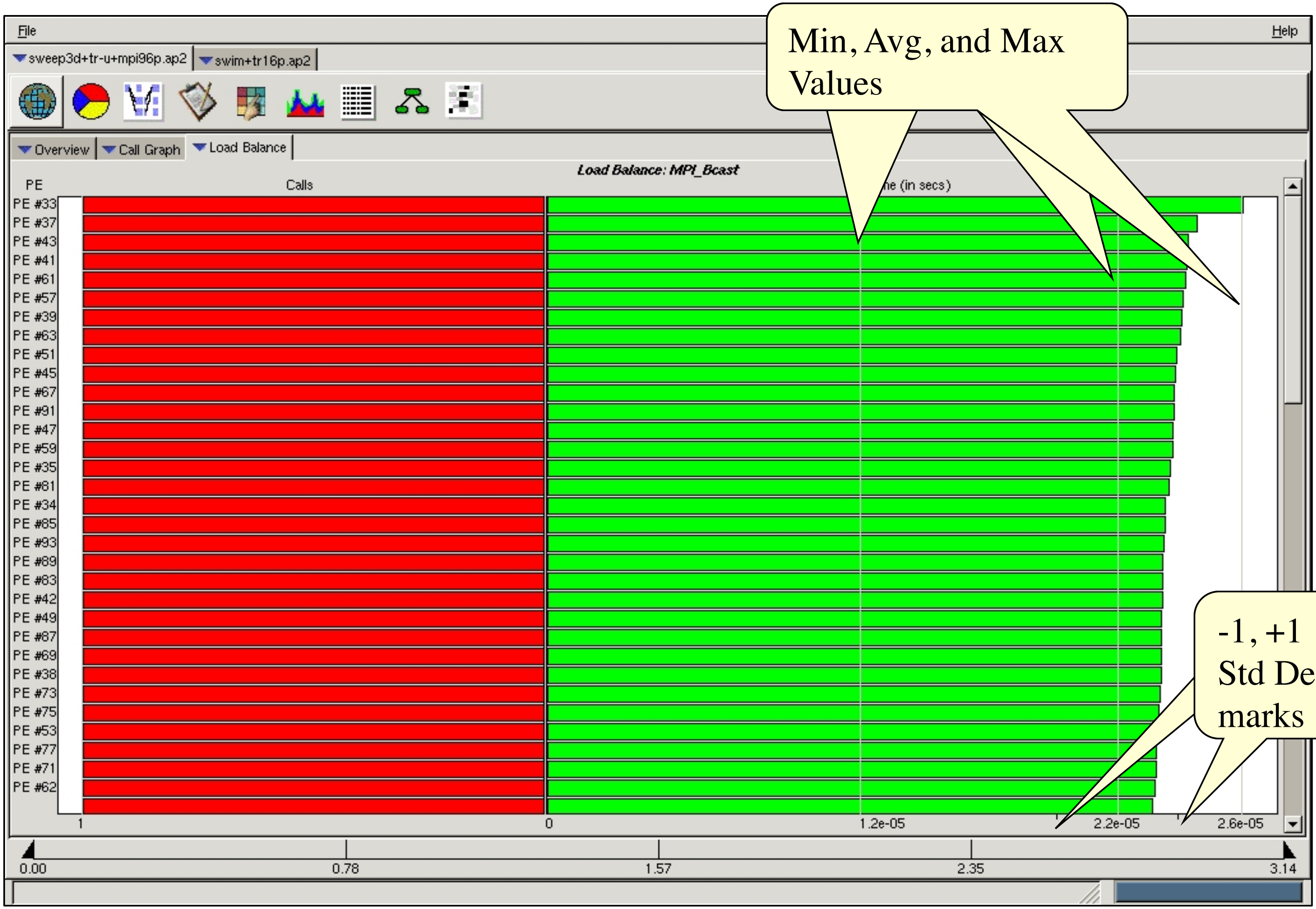


Function 'inner_' has the highest load imbalance time (19.770283 seconds) and is therefore a candidate for further examination for performance optimization.

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.

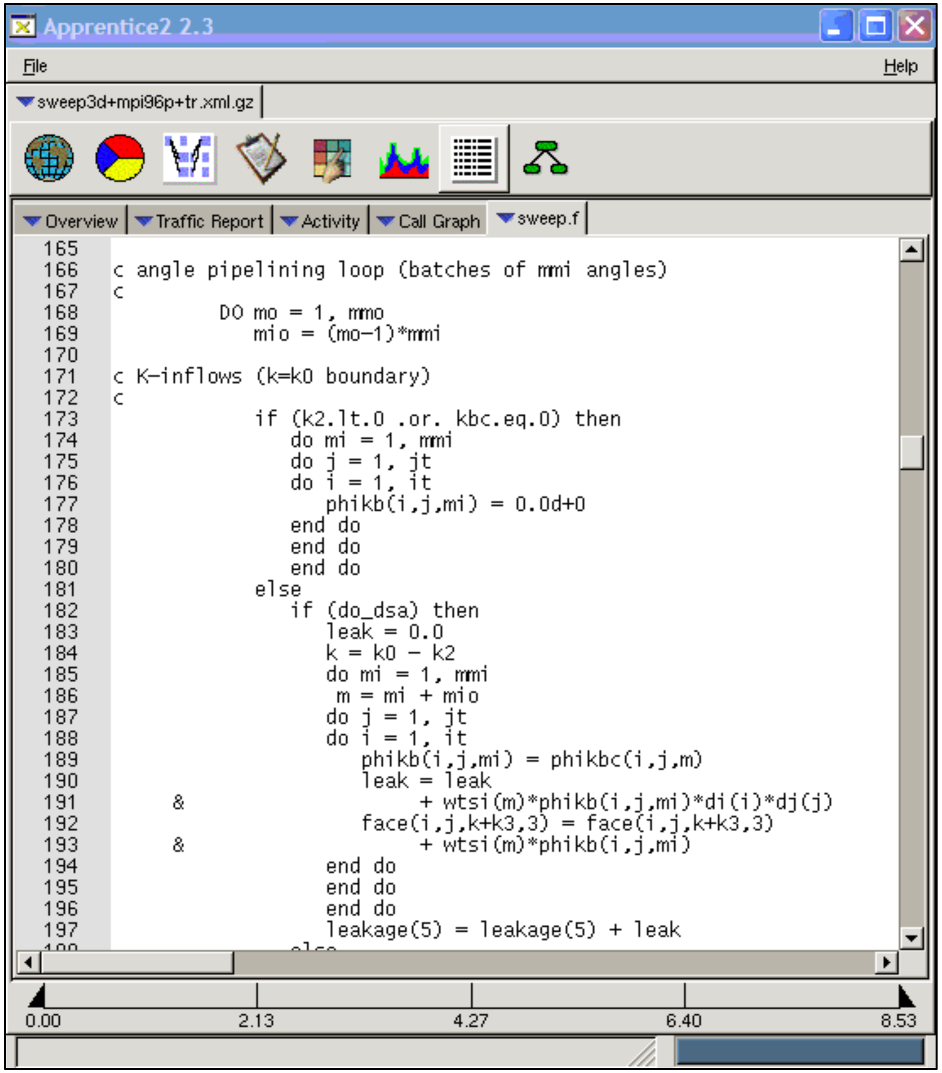
# Load Balance View (from Call Tree)



Min, Avg, and Max Values

-1, +1 Std Dev marks

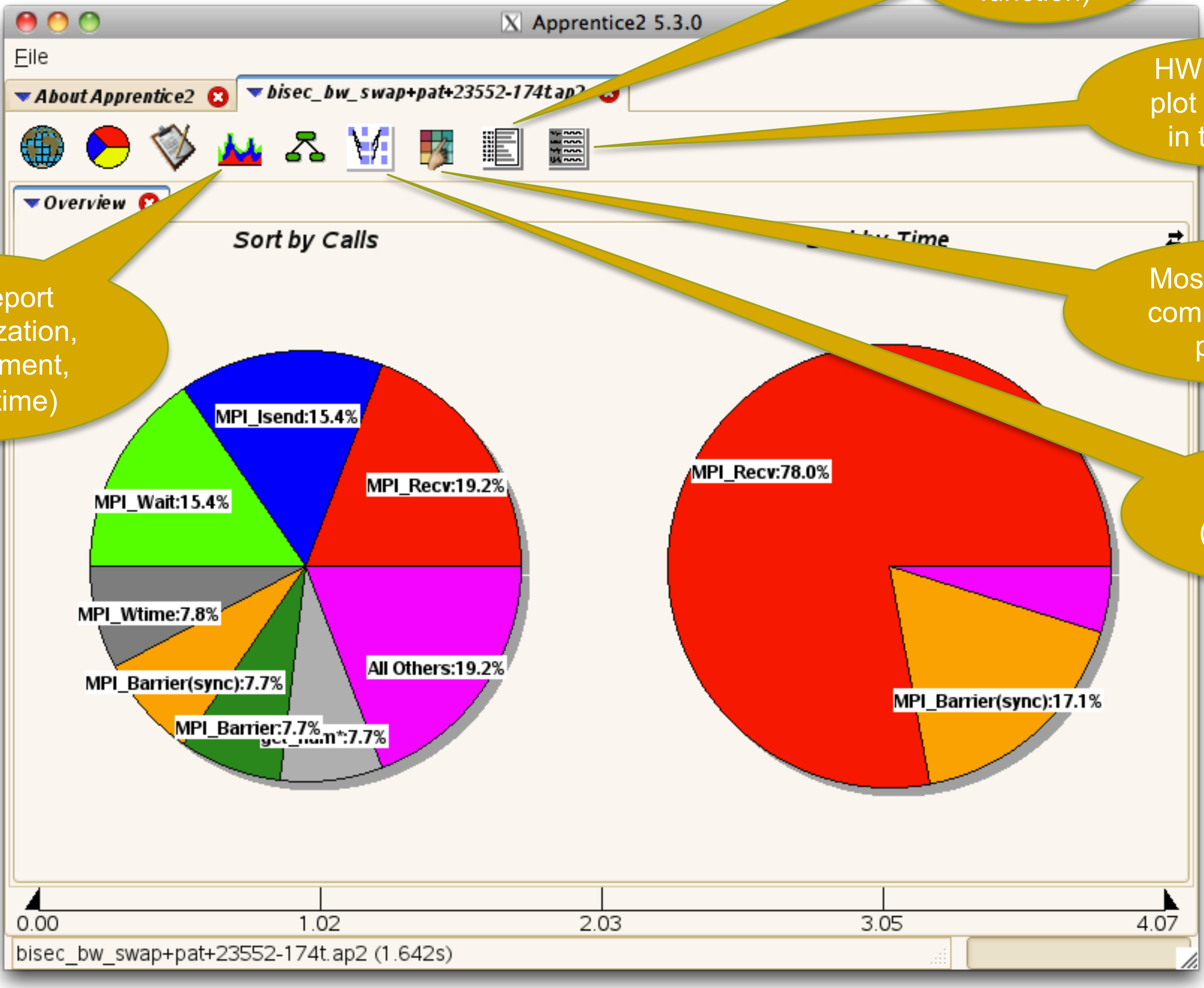
# Source Mapping from Call Tree



```

c angle pipelining loop (batches of mmi angles)
c
         DO mo = 1, mmo
            mio = (mo-1)*mmi

c K-inflows (k=k0 boundary)
c
            if (k2.lt.0 .or. kbc.eq.0) then
               do mi = 1, mmi
               do j = 1, jt
               do i = 1, it
                  phikb(i,j,mi) = 0.0d+0
               end do
               end do
               end do
            else
               if (do_dsa) then
                  leak = 0.0
                  k = k0 - k2
                  do mi = 1, mmi
                   m = mi + mio
                  do j = 1, jt
                  do i = 1, it
                     phikb(i,j,mi) = phikbc(i,j,m)
                     leak = leak
     &                    + wtsi(m)*phikb(i,j,mi)*di(i)*dj(j)
                     face(i,j,k+k3,3) = face(i,j,k+k3,3)
     &                    + wtsi(m)*phikb(i,j,mi)
                  end do
                  end do
                  end do
                  leakage(5) = leakage(5) + leak
               else
```

# Full Trace Visualization with Cray Apprentice2

# Trace Overview – Additional Views



HW counters overview (counter histogram by function)

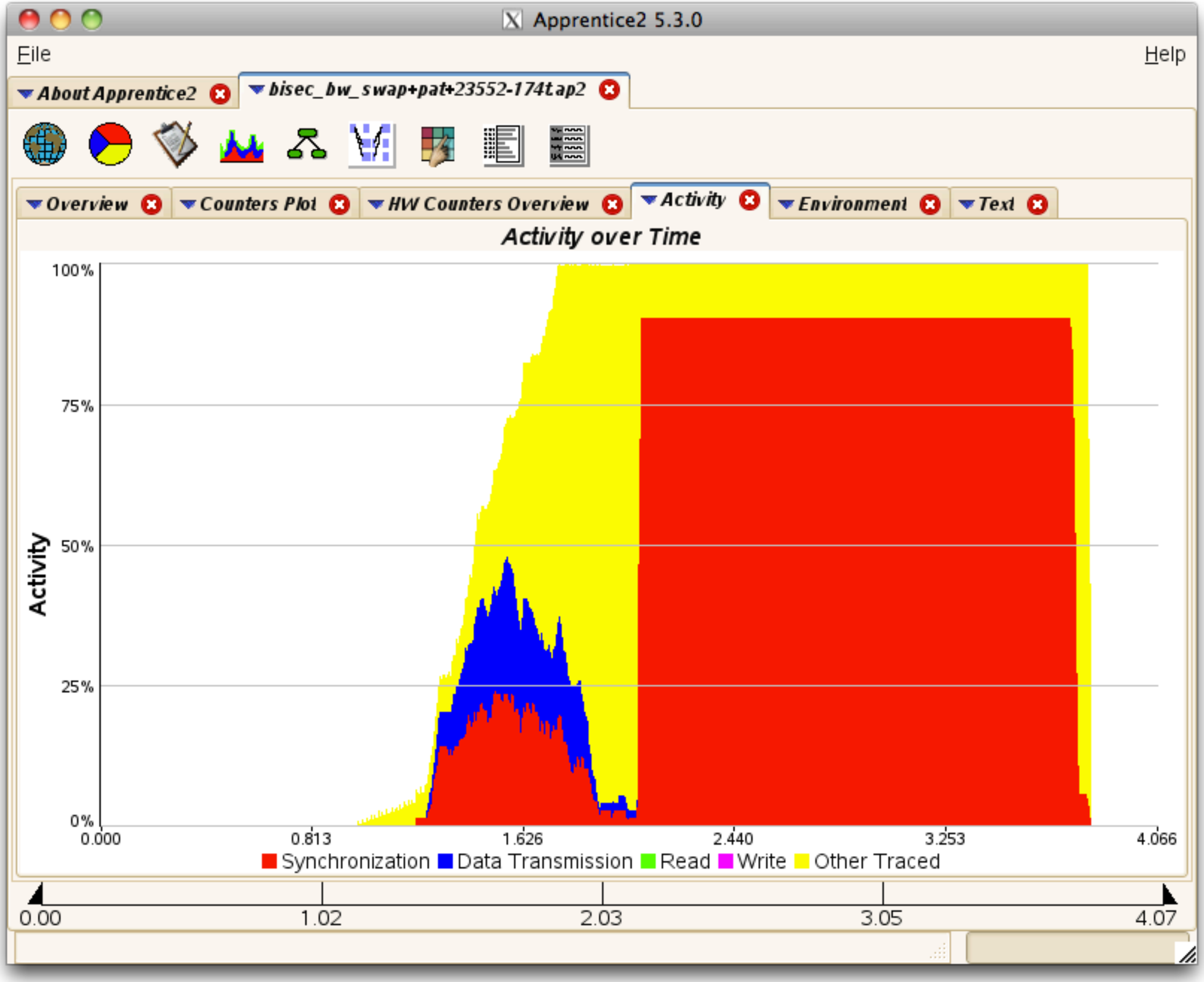
HW counters plot (counters in timeline)

Activity report (Synchronization, data movement, etc. over time)

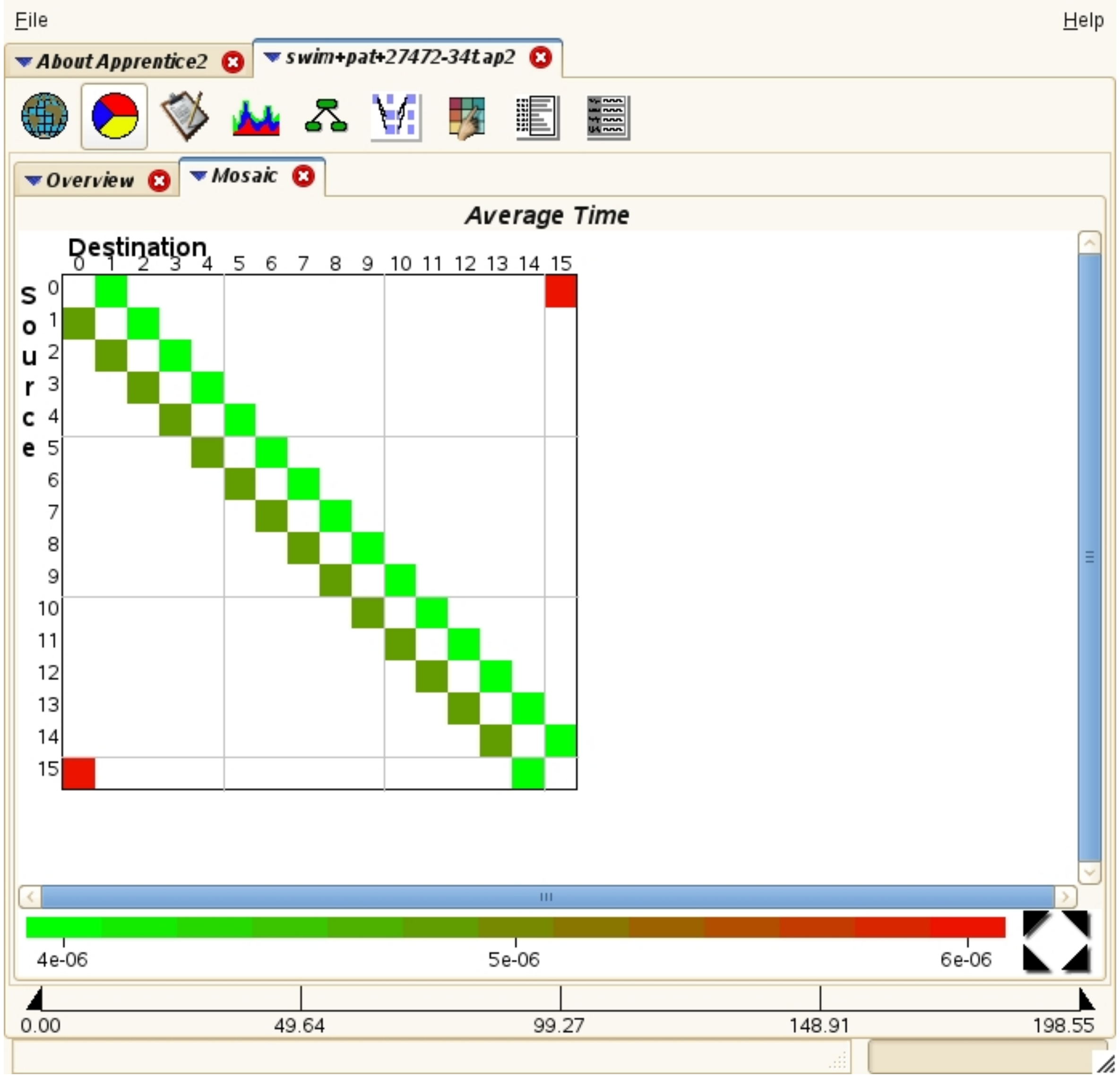
Mosaic (shows communication pattern)

Traffic report (MPI timeline)

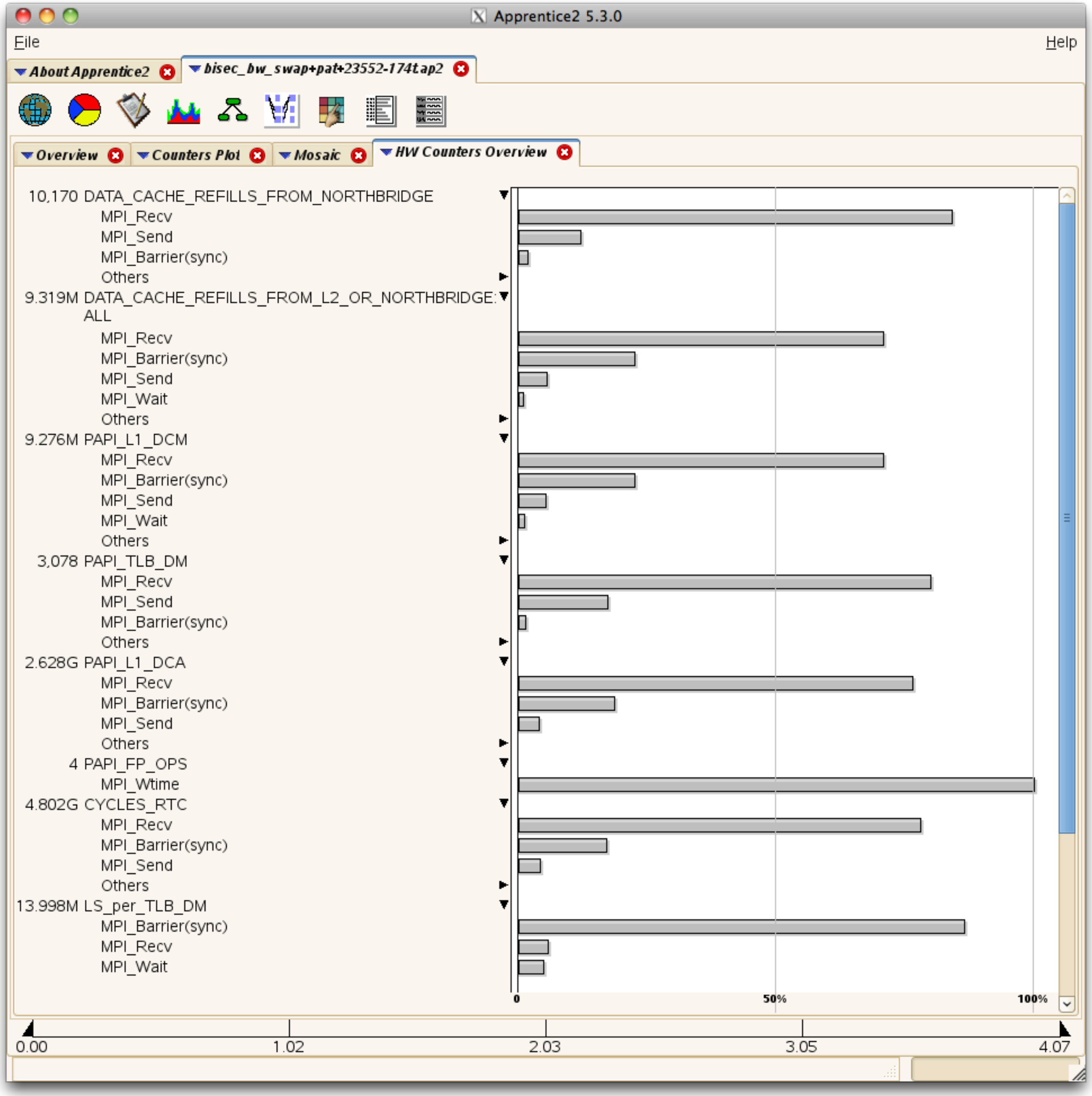
# Activity Report



# Mosaic View – Shows Communication Pattern

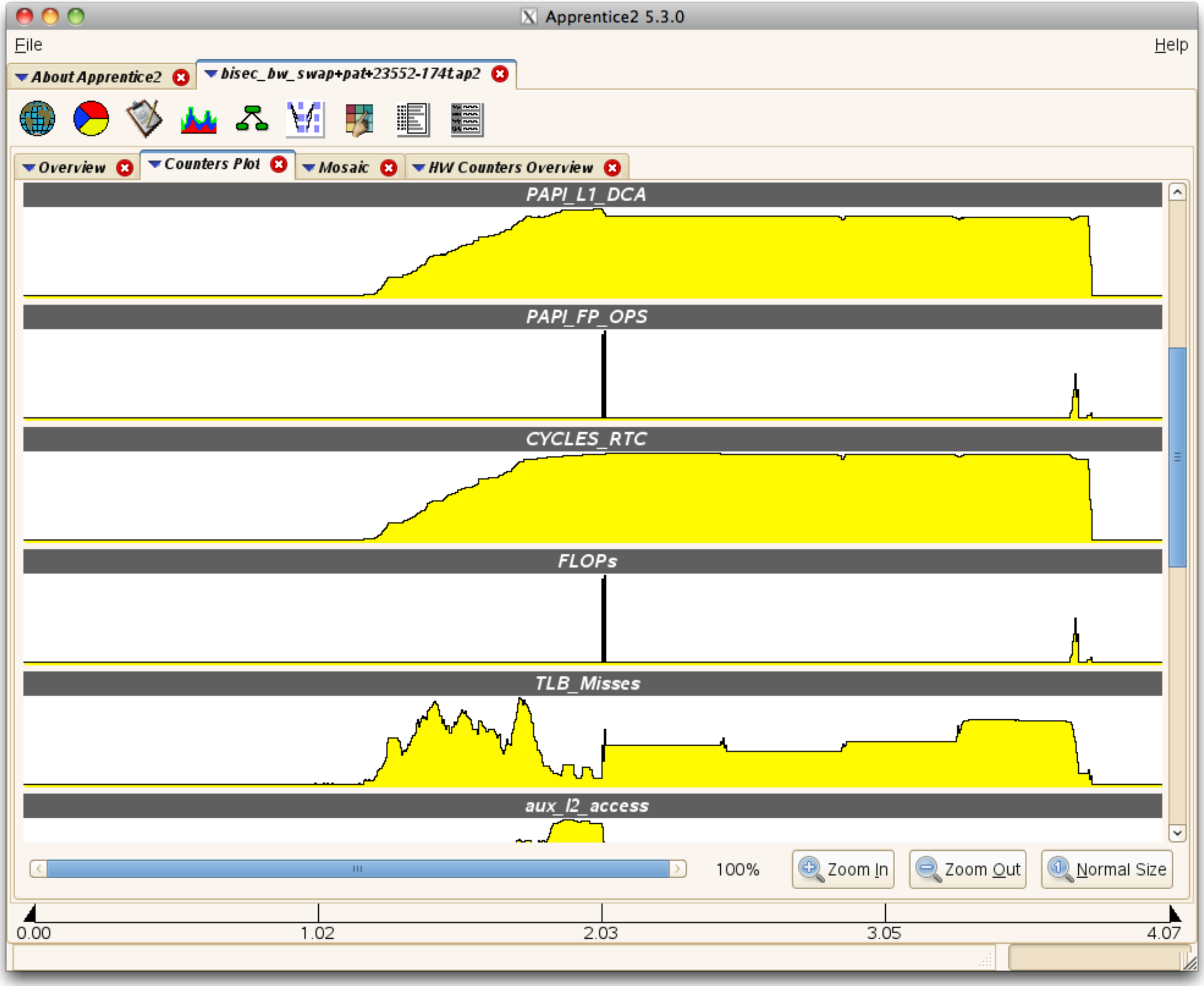


# HW Counters Overview

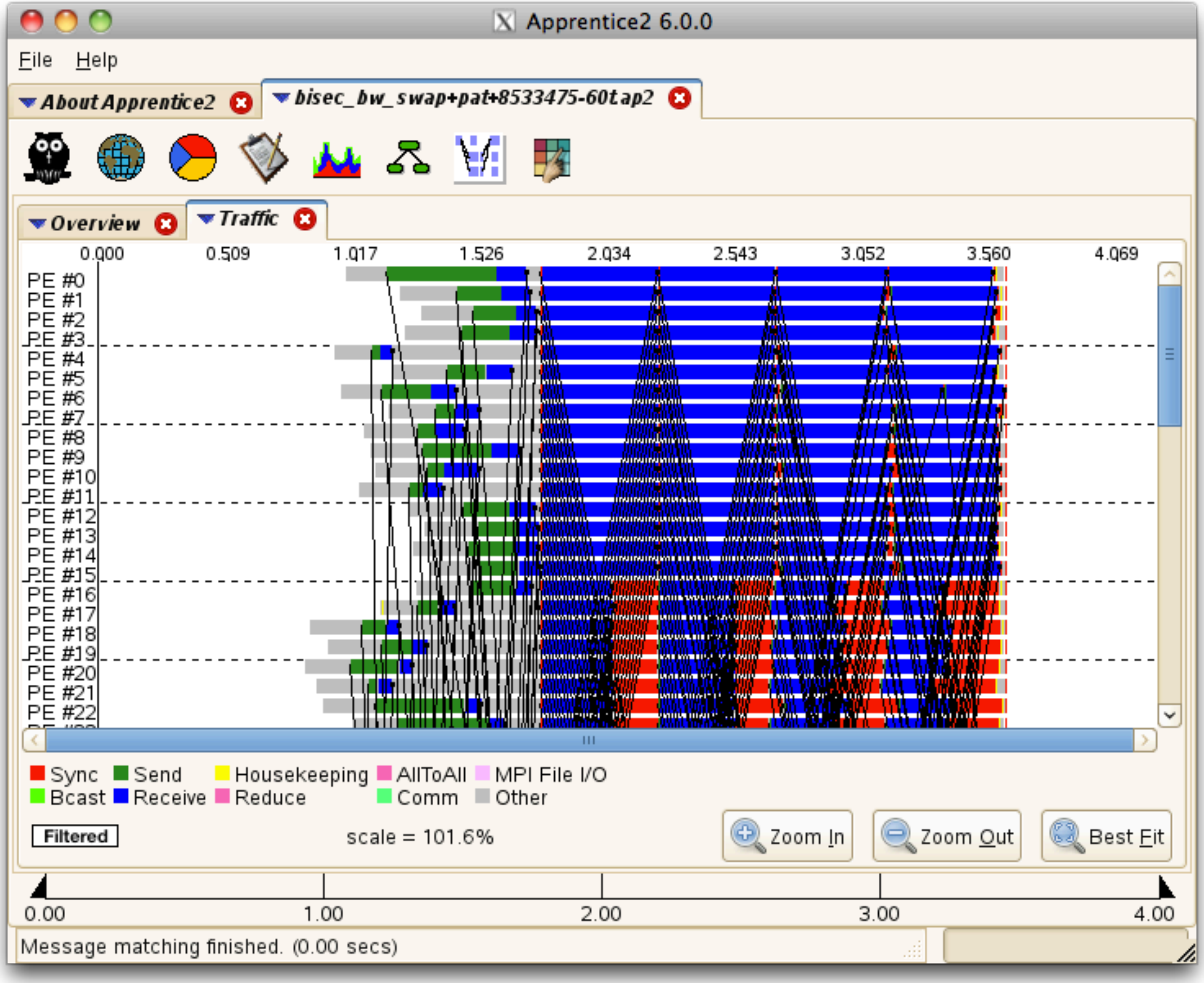


```
10,170 DATA_CACHE_REFILLS_FROM_NORTHBRIDGE
            MPI_Recv
            MPI_Send
            MPI_Barrier(sync)
            Others
9.319M DATA_CACHE_REFILLS_FROM_L2_OR_NORTHBRIDGE:
       ALL
            MPI_Recv
            MPI_Barrier(sync)
            MPI_Send
            MPI_Wait
            Others
9.276M PAPI_L1_DCM
            MPI_Recv
            MPI_Barrier(sync)
            MPI_Send
            MPI_Wait
            Others
 3,078 PAPI_TLB_DM
            MPI_Recv
            MPI_Send
            MPI_Barrier(sync)
            Others
2.628G PAPI_L1_DCA
            MPI_Recv
            MPI_Barrier(sync)
            MPI_Send
            Others
     4 PAPI_FP_OPS
            MPI_Wtime
4.802G CYCLES_RTC
            MPI_Recv
            MPI_Barrier(sync)
            MPI_Send
            Others
13.998M LS_per_TLB_DM
            MPI_Barrier(sync)
            MPI_Recv
            MPI_Wait
```

# HW Counters Plot



# Traffic Report – MPI Communication Timeline



Message matching finished. (0.00 secs)

# Man pages

- **intro_craypat(1)**
  - Introduces the craypat performance tool
- **pat_build(1)**
  - Instrument a program for performance analysis
- **pat_help(1)**
  - Interactive online help utility
- **pat_report(1)**
  - Generate performance report in both text and for use with GUI
- **app2 (1)**
  - Describes how to launch Cray Apprentice2 to visualize performance data

# Man pages (2)

- **hwpc(5)**
  - describes predefined hardware performance counter groups
- **nwpc(5)**
  - Describes predefined network performance counter groups
- **accpc(5) / accpc_k20(5)**
  - Describes predefined GPU performance counter groups
- **intro_papi(3)**
  - Lists PAPI event counters
  - Use papi_avail or papi_native_avail utilities to get list of events when running on a specific architecture

# Questions ?