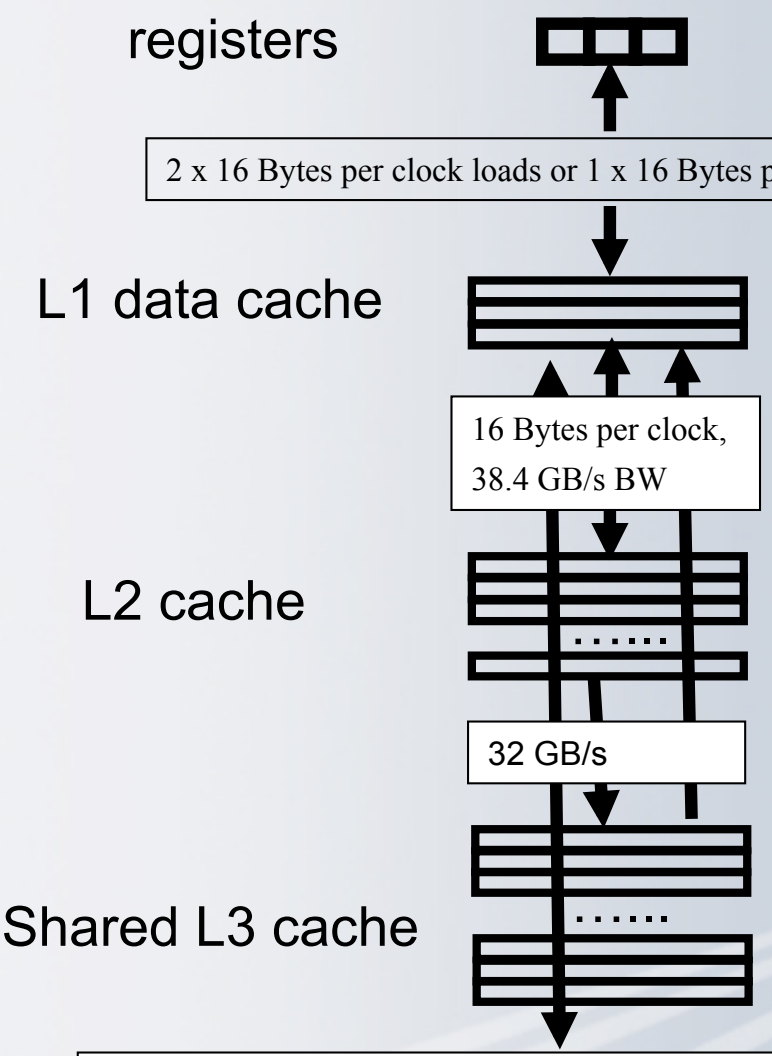
Using Hardware Performance Counters on the Cray XT

Luiz DeRose
Programming Environment Director
Cray Inc.
ldr@cray.com

CSCS
July 13-15, 2009

CRAY
THE SUPERCOMPUTER COMPANY

Simplified memory hierachy on the Quad Core AMD Opteron



16 SSE2 128-bit registers
16 64 bit registers

- 64 Byte cache line, 8 banks
- complete data cache lines are loaded from main memory, if not in L2 or L3 cache
- if L1 data cache needs to be refilled, then storing back to L2 cache, if L2 needs to be refilled, storing back to L3
- Items in L1 and L2 are exclusive, L3 is “sharing aware”

- 64 Byte cache line
- write back cache: data offloaded from L1 data cache are stored here first until they are flushed out to main memory

16 Bytes wide => 10.6 GB/s for DDR2-667, 73ns

Main memory

8GB/s over coherent Hyper Transport, 115ns

Remote memory

Hardware Performance Counters

- AMD Opteron Hardware Performance Counters
  - Four 48-bit performance counters.
    - Each counter can monitor a single event
      - Count specific processor events
        - » the processor increments the counter when it detects an occurrence of the event
        - » (e.g., cache misses)
      - Duration of events
        - » the processor counts the number of processor clocks it takes to complete an event
        - » (e.g., the number of clocks it takes to return data from memory after a cache miss)
  - Time Stamp Counters (TSC)
    - Cycles (user time)

PAPI Predefined Events

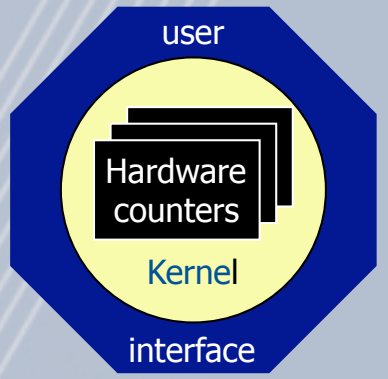
- Common set of events deemed relevant and useful for application performance tuning
  - Accesses to the memory hierarchy, cycle and instruction counts, functional units, pipeline status, etc.
  - The “papi_avail” utility shows which predefined events are available on the system – execute on compute node
- PAPI also provides access to native events
  - The “papi_native_avail” utility lists all AMD native events available on the system – execute on compute node
- Information on PAPI and AMD native events
  - pat_help counters
  - man papi_counters
  - For more information on AMD counters:
    - http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/26049.PDF

Hardware Counters Selection

- PAT_RT_HWPC <set number> | <event list>
  - Specifies hardware counter events to be monitored
    - A set number can be used to select a group of predefined hardware counters events (recommended)
      - CrayPat provides 19 groups on the Cray XT systems
    - Alternatively a list of hardware performance counter event names can be used
      - Maximum of 4 events
    - Both formats can be specified at the same time, with later definitions overriding previous definitions
    - Hardware counter events are not collected by default
    - Hardware counters collection is not supported with sampling on systems running Catamount on the compute nodes

Accuracy Issues

- Granularity of the measured code
  - If not sufficiently large enough, overhead of the counter interfaces may dominate
- Pay attention to what is not measured:
  - Out-of-order processors
  - Speculation
  - Lack of standard on what is counted
    - Microbenchmarks can help determine accuracy of the hardware counters
- For more information on AMD counters:
  - architecture manuals:
    - http://www.amd.com/us-en/assets/content_type/white_papers_and_tech_docs/26049.PDF



Hardware Performance Counters

```
Hardware performance counter events:
   PAPI_TOT_INS         Instructions completed
   PAPI_L1_DCA          Level 1 data cache accesses
   PAPI_FP_OPS          Floating point operations
   DATA_CACHE_MISSES    Data Cache Misses
   CYCLES_USER          User Cycles (approx, from clock ticks)

Estimated minimum overhead per call of a traced function,
   which was subtracted from the data shown in this report
   (for raw data, use the option:  -s overhead=include):
     PAPI_TOT_INS          2021.905  instr
     PAPI_L1_DCA           1275.739  refs
     PAPI_FP_OPS              0.000  ops
     DATA_CACHE_MISSES        7.702  misses
     CYCLES_USER              0.000  cycles
     Time                     2.054  microseconds
```

PAT_RT_HWPC=1 (Summary with TLB)

```
  PAPI_TLB_DM  Data translation lookaside buffer misses
  PAPI_L1_DCA  Level 1 data cache accesses
  PAPI_FP_OPS  Floating point operations
  DC_MISS      Data Cache Miss
  User_Cycles  Virtual Cycles
==========================================================================
USER
--------------------------------------------------------------------------
  Time%                                        68.0%
  Time                                     1.336838 secs
  Imb.Time                                      -- secs
  Imb.Time%                                     --
  Calls                     0.001M/sec       1500.0 calls
  PAPI_L1_DCM              14.365M/sec     18539562 misses
  PAPI_TLB_DM               2.114M/sec      2727811 misses
  PAPI_L1_DCA             276.056M/sec    356285406 refs
  PAPI_FP_OPS             382.697M/sec    493918940 ops
  User time (approx)        1.291 secs   2839375000 cycles  96.5%Time
  Average Time per Call                    0.000891 sec
  CrayPat Overhead : Time     0.2%
  HW FP Ops / User time   382.697M/sec    493918940 ops  4.3%peak(DP)
  HW FP Ops / WCT         369.468M/sec
  Computational intensity   0.17 ops/cycle   1.39 ops/ref
  MFLOPS (aggregate)      6123.16M/sec
  TLB utilization          130.61 refs/miss  0.255 avg uses
  D1 cache hit,miss ratios  94.8% hits        5.2% misses
  D1 cache utilization (M)  19.22 refs/miss  2.402 avg uses
==========================================================================
```

PAT_RT_HWPC=1
Flat profile data
Hard counts
Derived metrics

PAT_RT_HWPC=2 (L1 and L2 Metrics)

```
========================================================================
USER
------------------------------------------------------------------------
  Time%                                              57.5%
  Time                                            1.282322 secs
  Imb.Time                                             -- secs
  Imb.Time%                                            --
  Calls                        0.001M/sec           1500.0 calls
  REQUESTS_TO_L2:DATA         92.608M/sec        117033567 req
  DATA_CACHE_REFILLS:
    L2_MODIFIED:L2_OWNED:
    L2_EXCLUSIVE:L2_SHARED     9.691M/sec         12247253 fills
  DATA_CACHE_REFILLS_FROM_SYSTEM:
    ALL                       23.312M/sec         29461089 fills
  PAPI_L1_DCA                285.477M/sec        360771229 refs
  User time (approx)           1.264 secs       2780250000 cycles  98.6%Time
  Average Time per Call                           0.000855 sec
  CrayPat Overhead : Time       0.1%
  D1 cache hit,miss ratio (R)  88.4% hits           11.6% misses
  D1 cache utilization          8.65 refs/refill    1.081 avg uses
  D2 cache hit,miss ratio      74.8% hits           25.2% misses
  D1+D2 cache hit,miss ratio   91.8% hits            8.2% misses
  D1+D2 cache utilization      12.25 refs/miss      1.531 avg uses
  System to D1 refill         23.312M/sec         29461089 lines
  System to D1 bandwidth    1422.878MB/sec       1885509672 bytes
  L2 to Dcache bandwidth     591.504MB/sec        783824176 bytes
========================================================================
```

PAT_RT_HWPC=3 (Bandwidth)

```
========================================================================
USER / mlwxyz_
------------------------------------------------------------------------
  Time%                                                  44.0%
  Time                                                5.393606
  Imb.Time                                            0.054000
  Imb.Time%                                              1.0%
  Calls                                                    10
  QUADWORDS_WRITTEN_TO_SYSTEM:
    ALL                                76.516M/sec     410363958 ops
  DATA_CACHE_REFILLS:
    L2_MODIFIED:L2_OWNED:
    L2_EXCLUSIVE:L2_SHARED             14.494M/sec      77731399 fills
  DATA_CACHE_REFILLS_FROM_SYSTEM:
    ALL                                18.999M/sec     101891701 fills
  DATA_CACHE_LINES_EVICTED:ALL         52.589M/sec     282042348 ops
  User time (approx)                    5.363 secs   11262496875 cycles
  Average Time per Call                                0.539361 sec/call
  Cycles                                5.363 secs   11262496875 cycles
  User time (approx)                    5.363 secs   11262496875 cycles
  Utilization rate                                        99.4%
  D2 cache hit ratio                                      43.3%
  System to D1 refill                  18.999M/sec     101891701 lines
  System to D1 bandwidth             1159.587MB/sec    6521068888 bytes
  L2 to Dcache bandwidth              884.629MB/sec    4974809544 bytes
  L2 to System BW per core            583.773MB/sec    3282911662 bytes
========================================================================
```

PAT_RT_HWPC=5 (Floating point mix)

```
================================================================================
USER
--------------------------------------------------------------------------------
  Time%                                        58.5%
  Time                                     1.166749 secs
  Imb.Time                                      -- secs
  Imb.Time%                                     --
  Calls                    0.001M/sec       1500.0 calls
  RETIRED_MMX_AND_FP_INSTRUCTIONS:
    PACKED_SSE_AND_SSE2  481.704M/sec    544927850 instr
  PAPI_FML_INS           153.030M/sec    173115267 ops
  PAPI_FAD_INS           283.583M/sec    320803673 ops
  PAPI_FDV_INS             7.258M/sec      8210206 ops
  User time (approx)       1.131 secs   2601875000 cycles  97.0%Time
  Average Time per Call                    0.000778 sec
  CrayPat Overhead : Time   0.2%
  HW FP Ops / Cycles                          0.19 ops/cycle
  HW FP Ops / User time  436.613M/sec    493918940 ops  4.7%peak(DP)
  HW FP Ops / WCT        423.329M/sec
  FP Multiply / FP Ops                       35.0%
  FP Add / FP Ops                            65.0%
  MFLOPS (aggregate)     6985.81M/sec
================================================================================
```

PAT_RT_HWPC=12 (QC Vectorization)

```
================================================================================
USER
--------------------------------------------------------------------------------
  Time%                                          62.6%
  Time                                       1.251600 secs
  Imb.Time                                        -- secs
  Imb.Time%                                       --
  Calls                      0.001M/sec        1500.0 calls
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS                                 0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS           199.842M/sec   248803518 ops
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS:OP_TYPE                         0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS:OP_TYPE   396.722M/sec   493918940 ops
  User time (approx)           1.245 secs  2863500000 cycles  99.5%Time
  Average Time per Call                       0.000834 sec
  CrayPat Overhead : Time       0.2%
================================================================================
```

Vectorization Example

```
================================================================================
USER / calc2_
--------------------------------------------------------------------------------
  Time%                                          28.2%
  Time                                       0.600875 secs
  Imb.Time                                   0.069872 secs
  Imb.Time%                                     11.9%
  Calls                      864.9 /sec          500.0 calls
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS                                 0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS          369.139M/sec     213408500 ops
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS:OP_TYPE                         0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS:OP_TYPE  369.139M/sec     213408500 ops
  User time (approx)           0.578 secs   1271875000 cycles  96.2%Time
```

When compiled with fastsse:

```
================================================================================
USER / calc2_
--------------------------------------------------------------------------------
  Time%                                          24.3%
  Time                                       0.485654 secs
  Imb.Time                                   0.146551 secs
  Imb.Time%                                     26.4%
  Calls                        0.001M/sec          500.0 calls
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS                                 0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS          208.641M/sec     103016531 ops
  RETIRED_SSE_OPERATIONS:
    SINGLE_ADD_SUB_OPS:
    SINGLE_MUL_OPS:OP_TYPE                         0 ops
  RETIRED_SSE_OPERATIONS:
    DOUBLE_ADD_SUB_OPS:
    DOUBLE_MUL_OPS:OP_TYPE  415.628M/sec     205216531 ops
  User time (approx)           0.494 secs   1135625000 cycles  100.0%Time
```

How do I interpret these derived metrics?

- The following thresholds are guidelines to identify if optimization is needed:
  - **Computational Intensity: < 0.5 ops/ref**
    - This is the ratio of FLOPS by L&S
    - Measures how well the floating point unit is being used
  - **FP Multiply / FP Ops** or **FP Add / FP Ops**: **< 25%**
  - **Vectorization: < 1.5**

Memory Hierarchy Thresholds

- **TLB utilization: < 90.0%**
  - Measures how well the memory hierarchy is being utilized with regards to TLB
  - This metric depends on the computation being single precision or double precision
    - A page has 4 Kbytes. So, one page fits 512 double precision words or 1024 single precision words
  - TLB utilization < 1 indicates that not all entries on the page are being utilized between two TLB misses
- **D1 cache utilization: < 1 (D1+D2 cache utilization: < 1)**
  - A cache line has 64 bytes (8 double precision words or 16 single precision words)
  - D1 cache utilization < 1 indicates that not all entries on the cache line are being utilized between two cache misses
- **D1 cache hit (or miss) ratios: < 90% (> 10%)**
- **D2 (L2) cache hit (or miss) ratios: < 95% (> 5%)**
- **D1 + D2 cache hit (or miss) ratios: < 92% (> 8%)**
  - D1 and D2 caches on the Opteron are complementary
  - This metric provides a view of the Total Cache hit (miss) ratio

Detecting Load Imbalance on the Cray XT

Luiz DeRose
Programming Environment Director
Cray Inc.
ldr@cray.com

CSCS
July 13-15, 2009

CRAY
THE SUPERCOMPUTER COMPANY

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

Cray Tools Load Imbalance Support

- Very few performance tools focus on load imbalance
  - Need standard metrics
  - Need intuitive way of presentation
- CrayPat support:
  - Imbalance time and %
  - MPI sync time
  - OpenMP Performance Metrics
  - MPI rank placement suggestions
- Cray Apprentice^2 support:
  - Load imbalance visualization

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

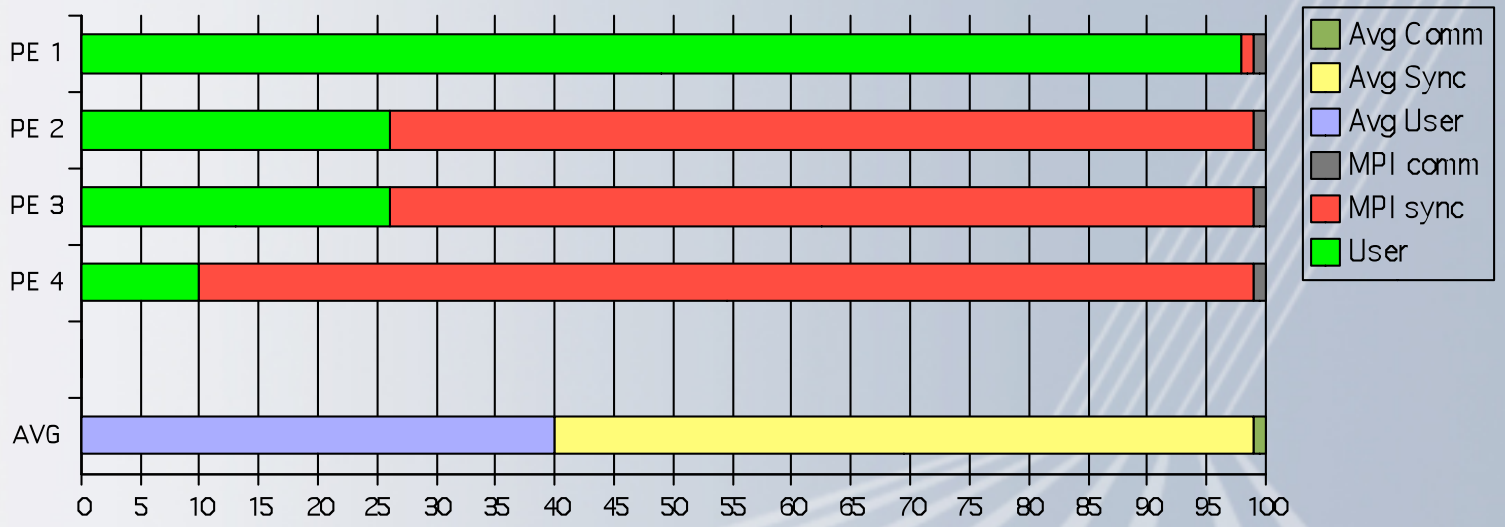
Load balance metric - rationale

Between two barriers

User: Imb = Max-Avg = 99-40 = 59

MPI Sync: Avg = 59

MPI Sync+Comm: Avg-Min = 60-1 = 59

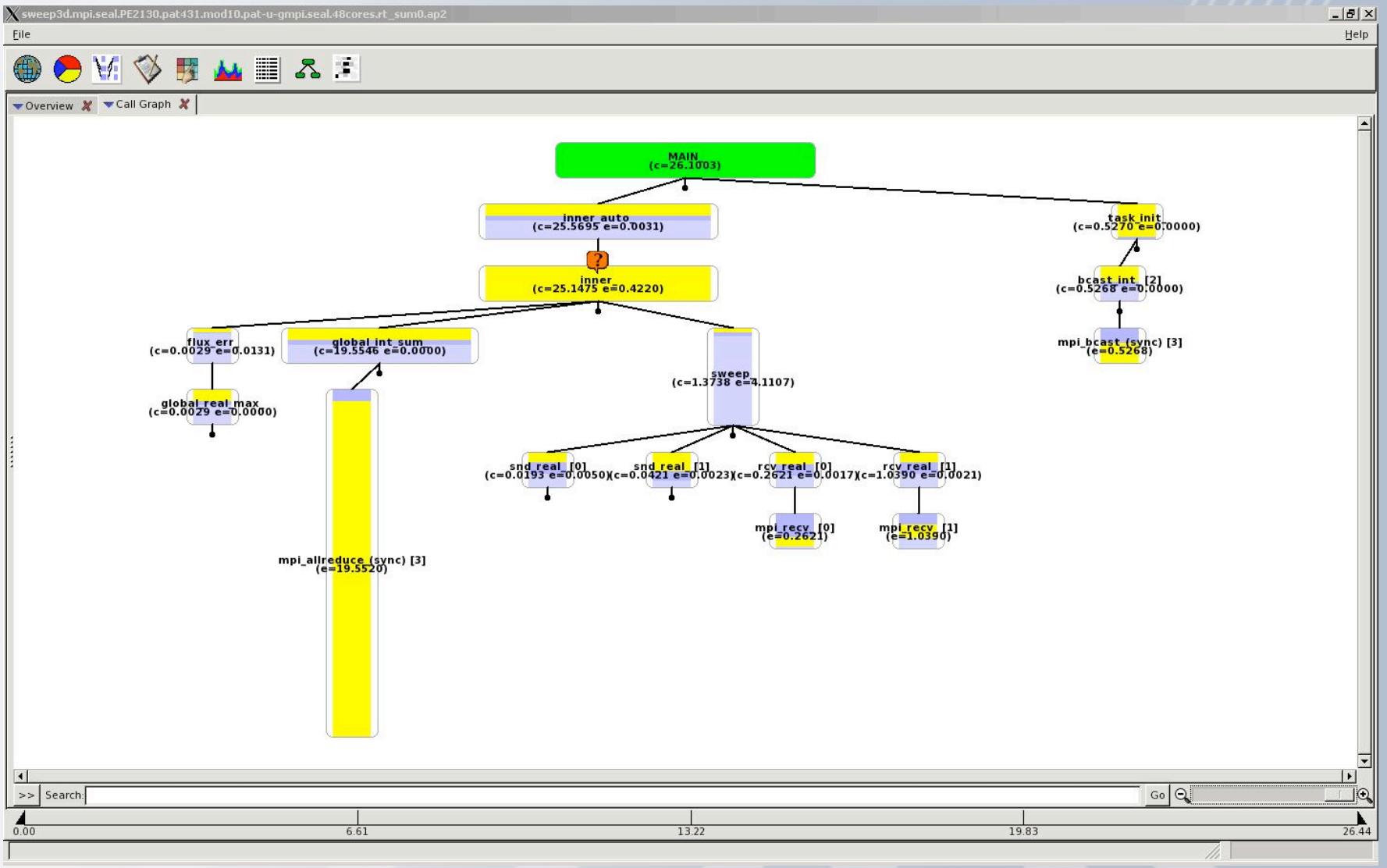


Imbalance %

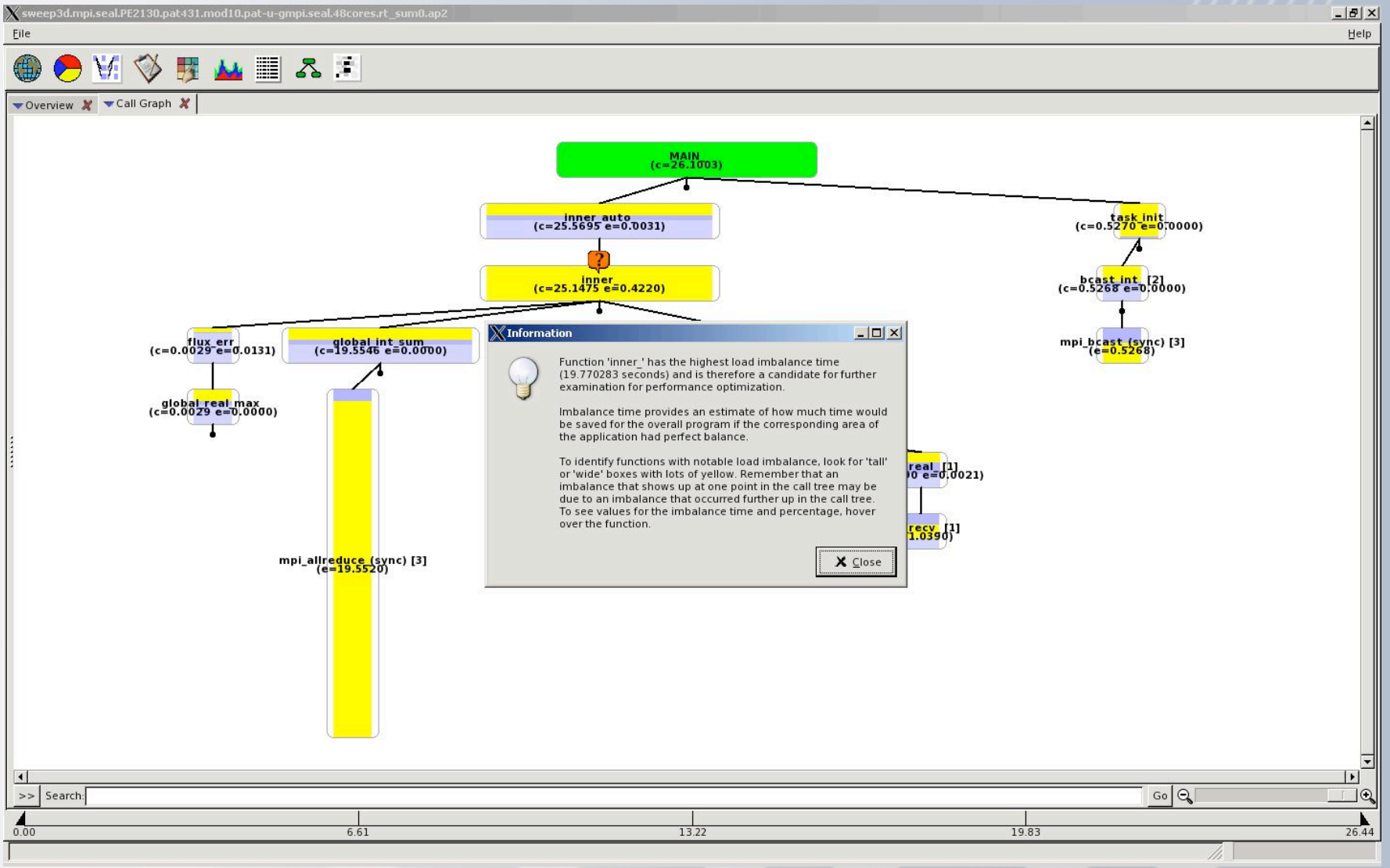
$$\text{Imbalance\%} = 100 \times \frac{\text{Imbalance time}}{\text{Max Time}} \times \frac{N}{N - 1}$$

- 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%

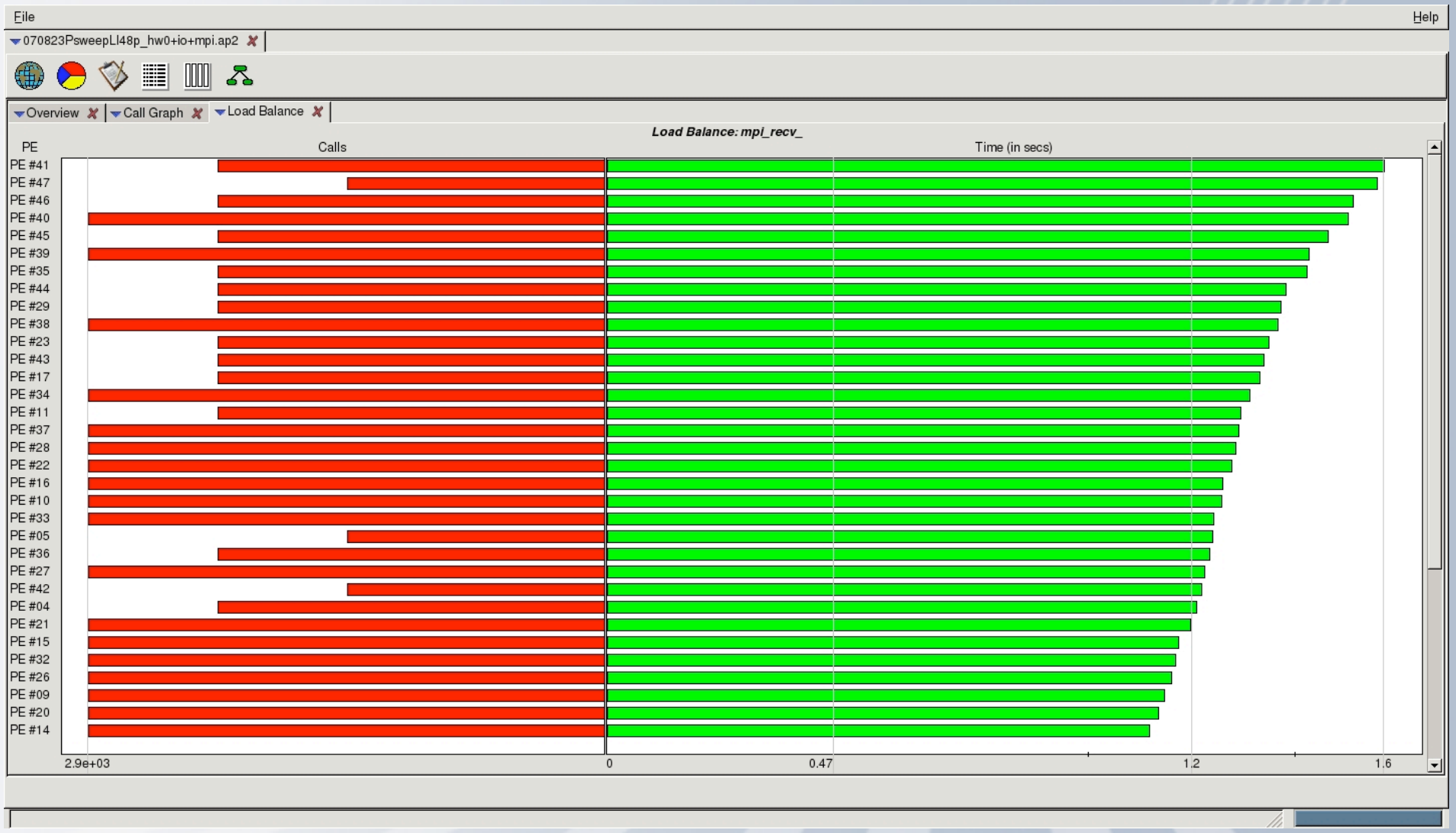
Call Tree Visualization (Swim3d)



Discrete Unit of Help (DUH Button)



Load Distribution



Profile with Load Distribution by Groups

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

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

 100.0% | 0.482144 |       -- |     -- |  2530 |Total
|-----------------------------------------------------------
|  83.7% | 0.403314 |       -- |     -- |   303 |USER
||----------------------------------------------------------
||  32.4% | 0.156028 | 0.009882 |   6.8% |    98 |calc3_
||  27.7% | 0.133643 | 0.007400 |   6.0% |   100 |calc2_
||  21.0% | 0.101406 | 0.002552 |   2.8% |   100 |calc1_
||   2.0% | 0.009696 | 0.000287 |   3.3% |     1 |inital_
||==========================================================
|  16.3% | 0.078830 |       -- |     -- |  2227 |MPI
||----------------------------------------------------------
||  12.7% | 0.061266 | 0.078133 |  64.1% |   351 |mpi_waitall_
||   2.2% | 0.010607 | 0.011582 |  59.7% |   936 |mpi_isend_
||   1.4% | 0.006945 | 0.004463 |  44.7% |   936 |mpi_irecv_
|===========================================================
```

MPI Sync Time

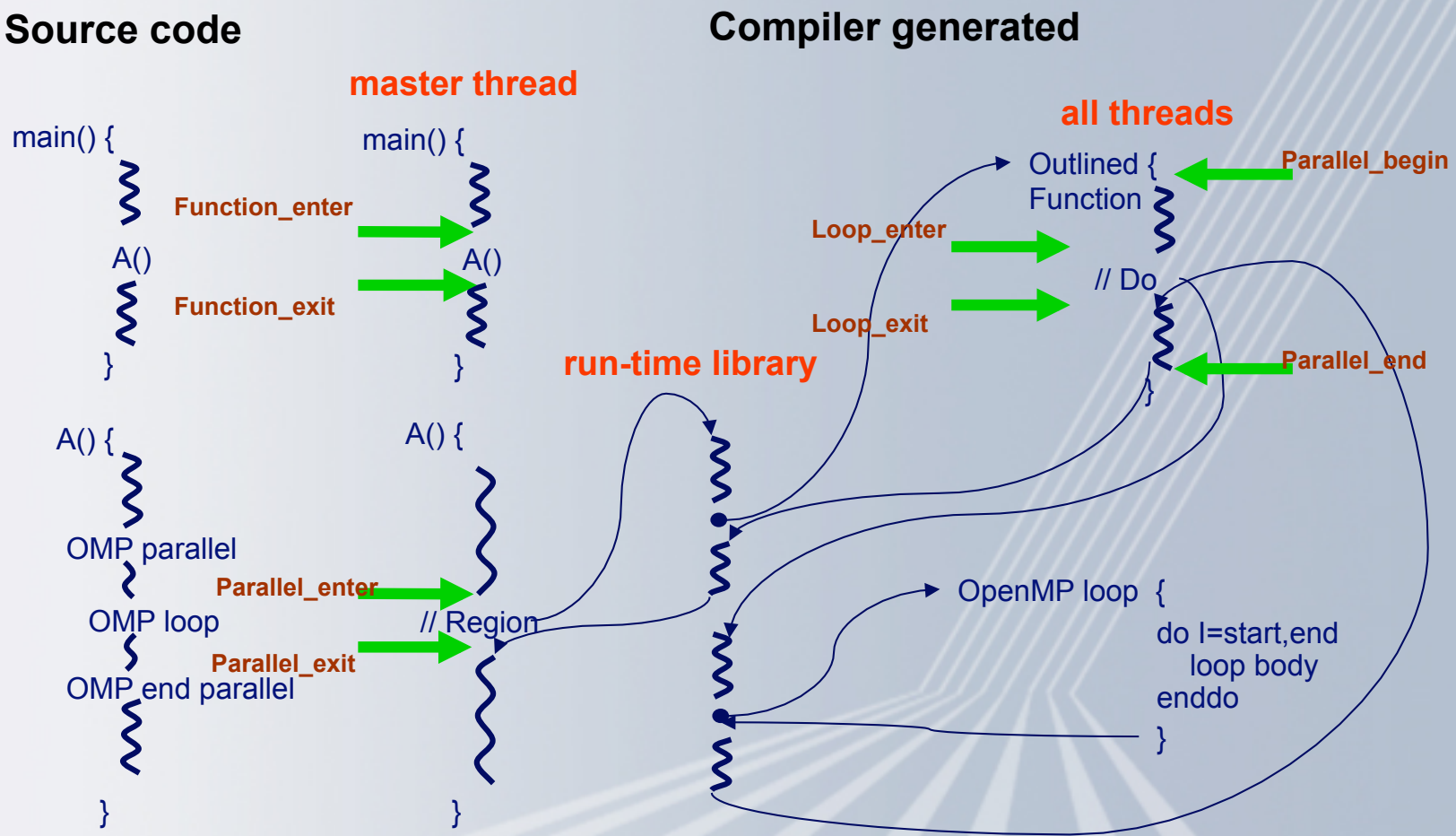
- Measure load imbalance in programs instrumented to trace MPI functions to determine if MPI ranks arrive at collectives together
- Separates potential load imbalance from data transfer
- Sync times reported by default if MPI functions traced
- If desired, PAT_RT_MPI_SYNC=0 deactivated this feature

MPI Sync Time Statistics

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

 100.0% | 7.193714 |       -- |     -- | 17604 |Total
|-----------------------------------------------------------
|  76.5% | 5.500078 |       -- |     -- |  4752 |USER
||----------------------------------------------------------
||  96.0% | 5.277791 | 0.171848 |   3.3% |    12 |sweep_
||   3.2% | 0.177352 | 0.005482 |   3.1% |    12 |source_
||   0.3% | 0.018588 | 0.000527 |   2.9% |    12 |flux_err_
||   0.2% | 0.010866 | 0.003033 |  22.8% |  2280 |snd_real_
||   0.1% | 0.005032 | 0.000144 |   2.9% |     1 |initialize_
||   0.1% | 0.004933 | 0.000154 |   3.2% |     1 |initxs_
||   0.1% | 0.002819 | 0.001773 |  40.3% |  2280 |rcv_real_
||==========================================================
|  16.6% | 1.197321 |       -- |     -- |  4603 |MPI
||----------------------------------------------------------
||  93.9% | 1.124227 | 0.277878 |  20.7% |  2280 |mpi_recv_
||   5.9% | 0.070481 | 0.014437 |  17.7% |  2280 |mpi_send_
||   0.2% | 0.002210 | 0.001088 |  34.4% |    32 |mpi_allreduce_
||==========================================================
|   6.3% | 0.453091 |       -- |     -- |    39 |MPI_SYNC
||----------------------------------------------------------
||  61.1% | 0.277012 | 0.215608 |  45.7% |     4 |mpi_bcast_(sync)
||  38.7% | 0.175564 | 0.270049 |  63.2% |    32 |mpi_allreduce_(sync)
||   0.1% | 0.000515 | 0.000265 |  35.5% |     3 |mpi_barrier_(sync)
|==========================================================
```

OpenMP (Ideal) Instrumentation



CrayPat OpenMP Performance Metrics

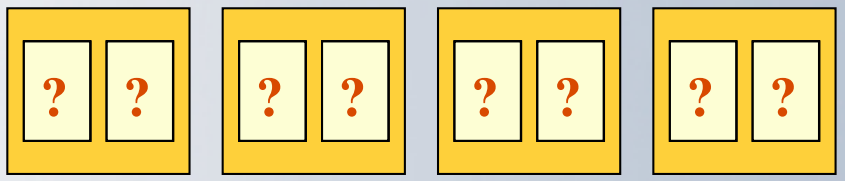
- Per-thread timings
- Overhead incurred at enter/exit of
  - Parallel regions
  - Worksharing constructs within parallel regions
- Load balance information across threads
- Sampling performance data without API
- Separate metrics for OpenMP runtime and OpenMP API calls

OpenMP Data from pat_report

- Default view (no options needed to pat_report)
  - Focus on where program is spending its time
  - Shows imbalance across all threads
  - Assumes all requested resources should be used
  - Highlights non-uniform imbalance across threads

MPI Rank Reorder

- MPI rank placement with environment variable



  - Distributed placement
  - SMP style placement
  - Folded rank placement
  - User provided rank file

Rank Reorder Example - hycom

```
pat_report -O load_balance

Table 2:  Load Balance across PE's by FunctionGroup

  Time % |    Cum. |        Time |       Calls |Group
         | Time % |             |             | PE[mmm]

  100.0% | 100.0% | 482.705844 | 7446623155 |Total
|-----------------------------------------------
|  57.7% |  57.7% | 278.657370 | 7329740077 |USER
||----------------------------------------------
||   0.5% |   0.5% | 361.310805 |   33130409 |pe.201
||   0.4% |  58.2% | 311.898417 |   34020074 |pe.45
||   0.0% | 100.0% |  23.780267 |     320096 |pe.184
||==============================================
|  42.3% | 100.0% | 204.048383 |  116783478 |MPI
||----------------------------------------------
||   0.9% |   0.9% | 476.662251 |     399087 |pe.184
||   0.3% |  61.9% | 167.921814 |     422197 |pe.37
||   0.2% | 100.0% | 119.123503 |     514637 |pe.201
|==============================================
```

Rank Reorder Example - hycom

```
pat_report -O load_balance -s pe=ALL

Table 2:  Load Balance across PE's by FunctionGroup

  Time % |    Cum. |        Time |       Calls |Group
         | Time % |             |             | PE

  100.0% | 100.0% | 482.705844 | 7446623155 |Total
|-----------------------------------------------
|  57.7% |  57.7% | 278.657370 | 7329740077 |USER
||----------------------------------------------
||   0.5% |   0.5% | 361.310805 |   33130409 |pe.201
||   0.5% |   1.0% | 349.849557 |   30460022 |pe.182
||   0.5% |   1.5% | 346.919713 |   33685730 |pe.200
||   0.5% |   2.0% | 342.844256 |   34879988 |pe.188
||   0.5% |   2.5% | 342.308415 |   34913960 |pe.172
. . .
||   0.1% |  99.8% |  45.464691 |    3000260 |pe.248
||   0.1% |  99.9% |  35.970972 |     399401 |pe.213
||   0.0% |  99.9% |  27.431543 |     340673 |pe.232
||   0.0% | 100.0% |  25.142167 |     117620 |pe.240
||   0.0% | 100.0% |  23.780267 |     320096 |pe.184
||==============================================
```

Rank Reorder Example - hycom

After custom placement (10% performance improvement):

```
Table 2:  Load Balance with MPI Sent Message Stats

  Time % |       Time | Sent Msg |     Sent Msg | Avg Sent |Group
         |            |    Count |  Total Bytes | Msg Size | PE[mmm]

  100.0% | 437.418783 | 17161829 | 289328285840 | 16858.83 |Total
|-------------------------------------------------------------------
|  60.2% | 263.211966 |       -- |           -- |       -- |USER
||------------------------------------------------------------------
||   0.5% | 322.019049 |       -- |           -- |       -- |pe.158
||   0.4% | 286.179471 |       -- |           -- |       -- |pe.126
||   0.0% |  23.318648 |       -- |           -- |       -- |pe.184
||==================================================================
|  39.8% | 174.206510 | 17161829 | 289328285840 | 16858.83 |MPI
||------------------------------------------------------------------
||   1.0% | 414.091071 |    62224 |    635942368 | 10220.21 |pe.184
||   0.3% | 151.242560 |    68002 |   1039329136 | 15283.80 |pe.126
||   0.3% | 115.396258 |    68002 |   1039329136 | 15283.80 |pe.158
|==================================================================
```

MPI Rank Placement Suggestions

- When to use?
  - Point-to-point communication consumes significant fraction of the program time and have a significant imbalance
    - pat_report -O mpi_sm_rank_order ...
  - When there seems to be a load imbalance of another type
    - Can get a suggested rank order file based on user time
      - pat_report -O mpi_rank_order ...
    - Can have a different metric for load balance
      - pat_report -O mpi_rank_order -s mro_metric=DATA_CACHE_MISSES ...
- Information in resulting report
  - Available if MPI functions traced (-g mpi)
- Custom placement files automatically generated

MPI Rank Placement Suggestions (cont'd)

- See table notes in resulting report from pat_report
- Report provides quad core and dual core suggestions
- Set MPICH_RANK_REORDER_METHOD environment variable
  - Set to numerical value or MPICH_RANK_ORDER file from pat_report

Example: -O mpi_rank_order (asura)

```
Notes for table 1:

    To maximize the locality of point to point communication, choose
    and specify a Rank Order with small Max and Avg Sent Msg Total Bytes
    per node for the target number of cores per node.

    To specify a Rank Order with a numerical value, set the environment
    variable MPICH_RANK_REORDER_METHOD to the given value.

    To specify a Rank Order with a letter value 'x', set the environment
    variable MPICH_RANK_REORDER_METHOD to 3, and copy or link the file
    MPICH_RANK_ORDER.x to MPICH_RANK_ORDER.

Table 1:  Sent Message Stats and Suggested MPI Rank Order


                    Sent Msg Total Bytes per MPI rank

              Max           Avg           Min  Max   Min
      Total Bytes   Total Bytes   Total Bytes  Rank  Rank

        378638104     271474542     169280552  56    109

-------------------------------------------------------------
       Quad core:  Sent Msg Total Bytes per node

 Rank           Max           Avg           Min  Max Node       Min Node
Order   Total Bytes   Total Bytes   Total Bytes  Ranks          Ranks

    d    1093188824    1085898170    1071670808  92,124,35,91   86,27,108,63
    u    1093188824    1085898170    1071670808  92,124,35,91   86,27,108,63
    1    1249207480    1085898170     930426320  56,57,58,59    108,109,110,111
    2    1297029256    1085898170     936841176  70,57,71,56    74,53,75,52
    0    1300686504    1085898170     923754472  6,70,7,71      52,116,53,117
```

Example: File MPICH_RANK_ORDER.u (asura)

```
# Suggested custom rank placement:
#
#     pat_report -O mpi_sm_rank_order \
#       /home/crayadm/ldr/ASURA/asura10it.x+apa+4442-824tdt.ap2
#
#   Targets multi-core processors, based on Sent Msg Total Bytes.
#
#   Program:     /work/crayadm/ldr/ASURA/run/asura10it.x
#   Number PEs: 128
#   Cores/Node:  4
#
#   Heuristic: u
#
86,27,108,63,13,67,23,39,70,3,113,17,21,46,40,89
28,36,34,10,7,127,41,105,94,25,12,38,6,75,57,60
56,109,106,68,42,66,43,79,72,45,85,80,33,111,49,107
14,103,114,9,126,52,78,2,55,88,87,118,119,64,15,16
90,102,122,31,37,123,29,59,71,53,98,82,92,124,35,91
5,125,115,11,97,95,30,54,19,4,69,0,62,110,51,112
26,32,121,77,65,100,76,24,58,74,1,18,101,116,84,50
44,96,93,20,83,61,104,47,99,81,120,73,8,117,22,48
```

MPI + OpenMP? (some ideas)

- When does it pay to add OpenMP to my MPI code?
  - Only add OpenMP when code is network bound
  - Adding OpenMP to memory bound codes will most likely hurt performance rather than help it
  - Look at collective time, excluding sync time: this goes up as network becomes a problem
  - Look at point-to-point wait times: if these go up, network may be a problem

Parallel Performance Analysis and Visualization on the Cray XT

Luiz DeRose
Programming Environment Director
Cray Inc.
ldr@cray.com

CSCS
July 13-15, 2009

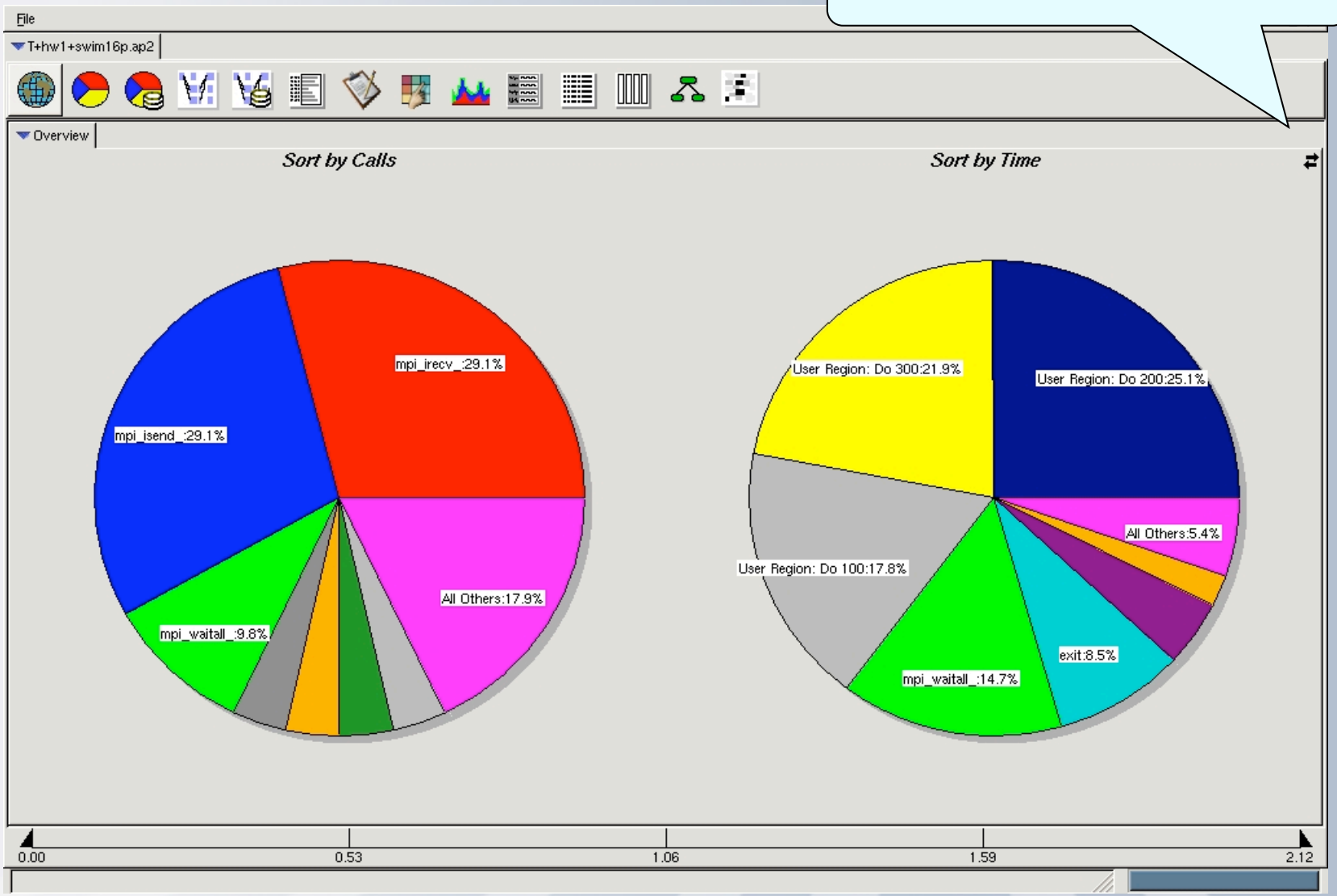
CRAY
THE SUPERCOMPUTER COMPANY

Cray Apprentice^2

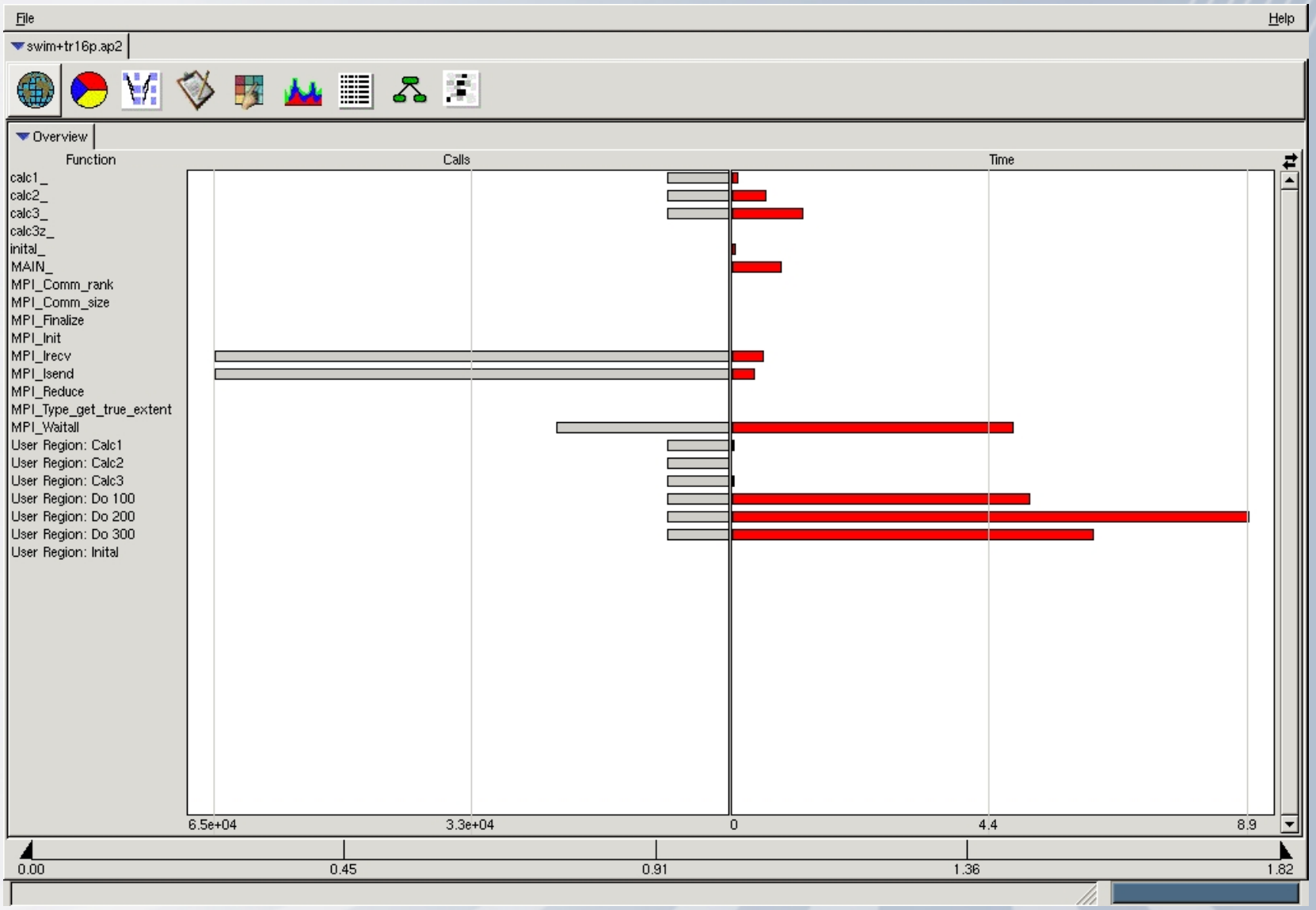
- Call graph profile
- Communication statistics
- Time-line view
  - Communication
  - I/O
- Activity view
- Pair-wise communication statistics
- Text reports
- Source code mapping

- Cray Apprentice^2
- is target to help and correct:
  - Load imbalance
  - Excessive communication
  - Network contention
  - Excessive serialization
  - I/O Problems

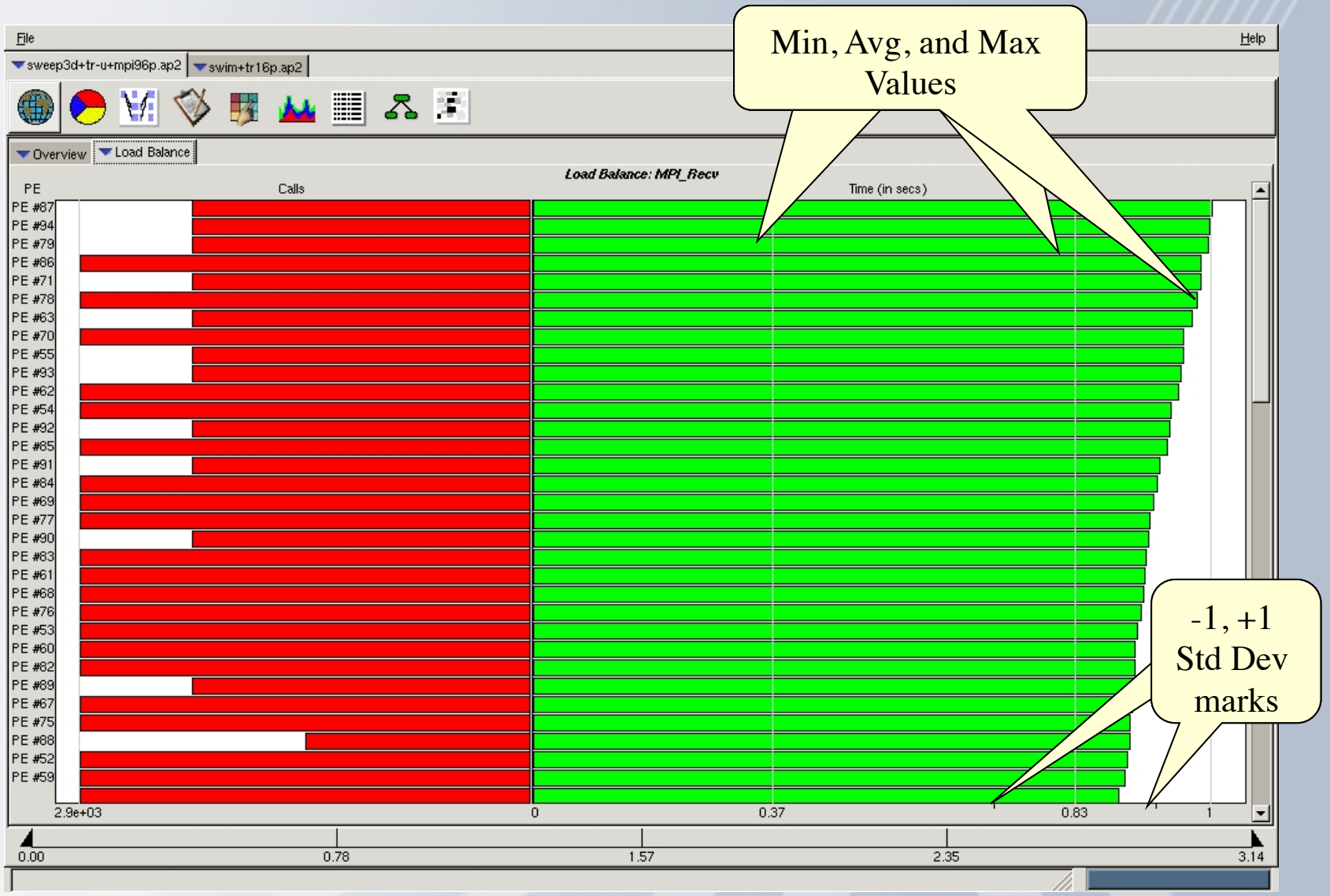
Statistics Overview



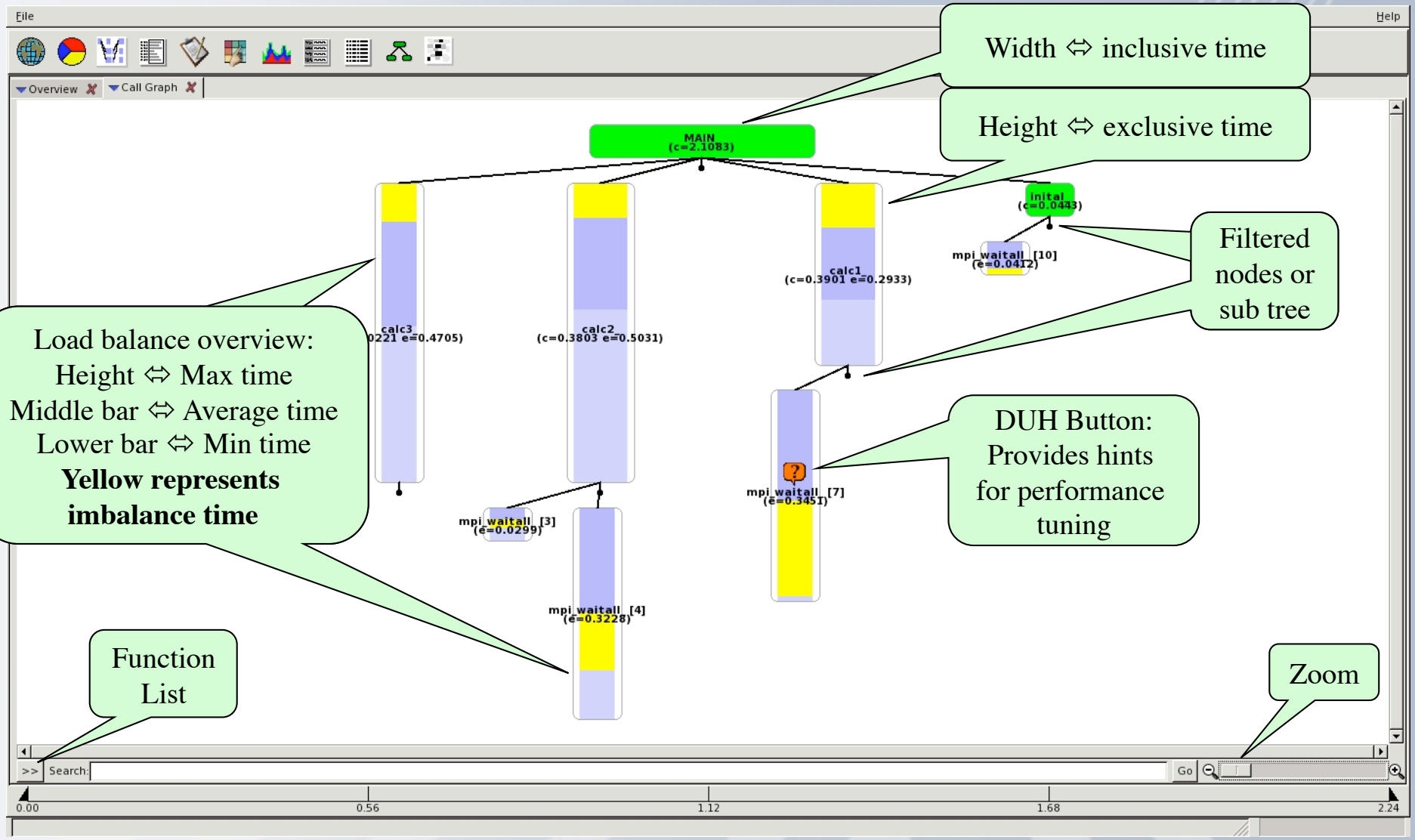
Function Profile



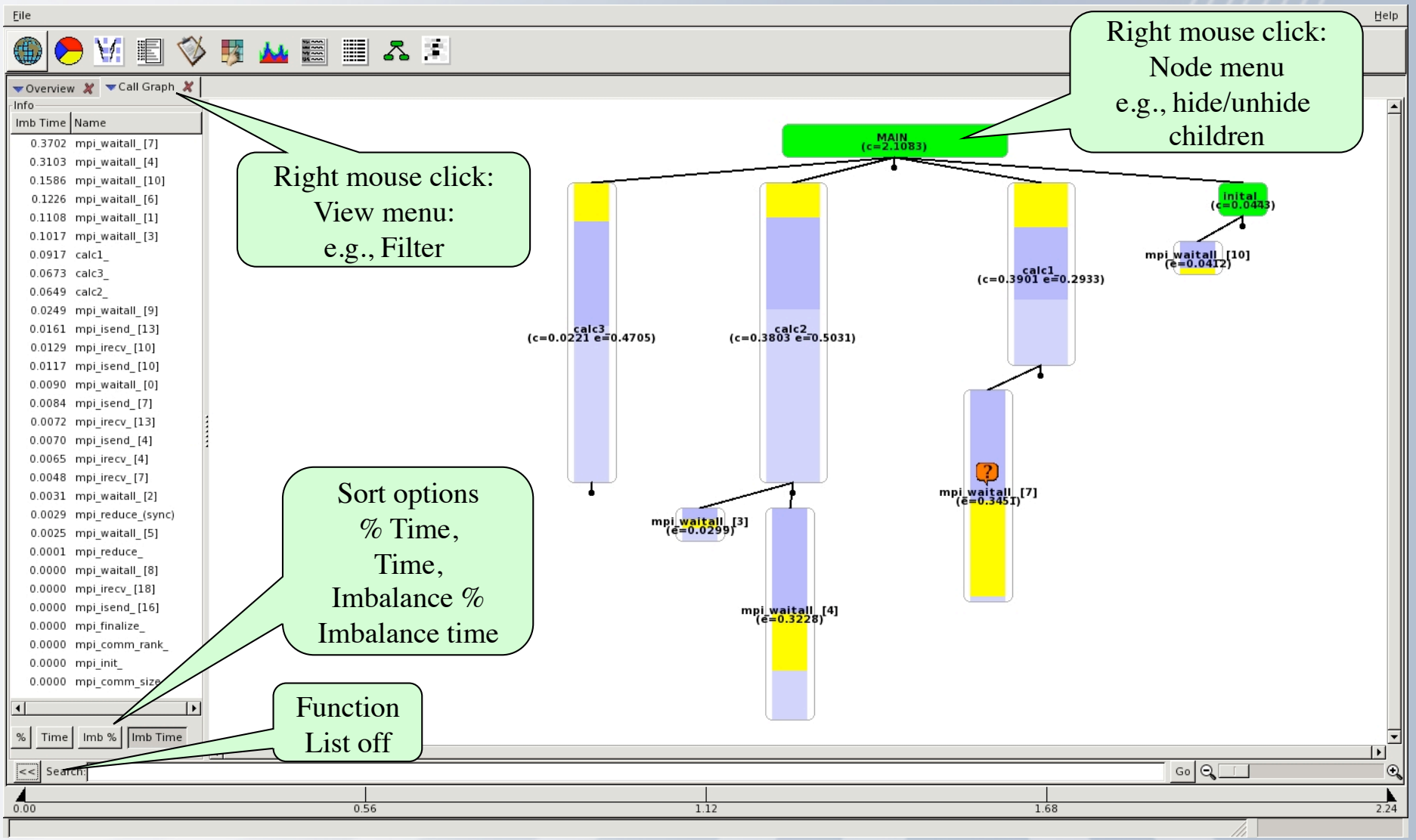
Load Balance View (Aggregated)



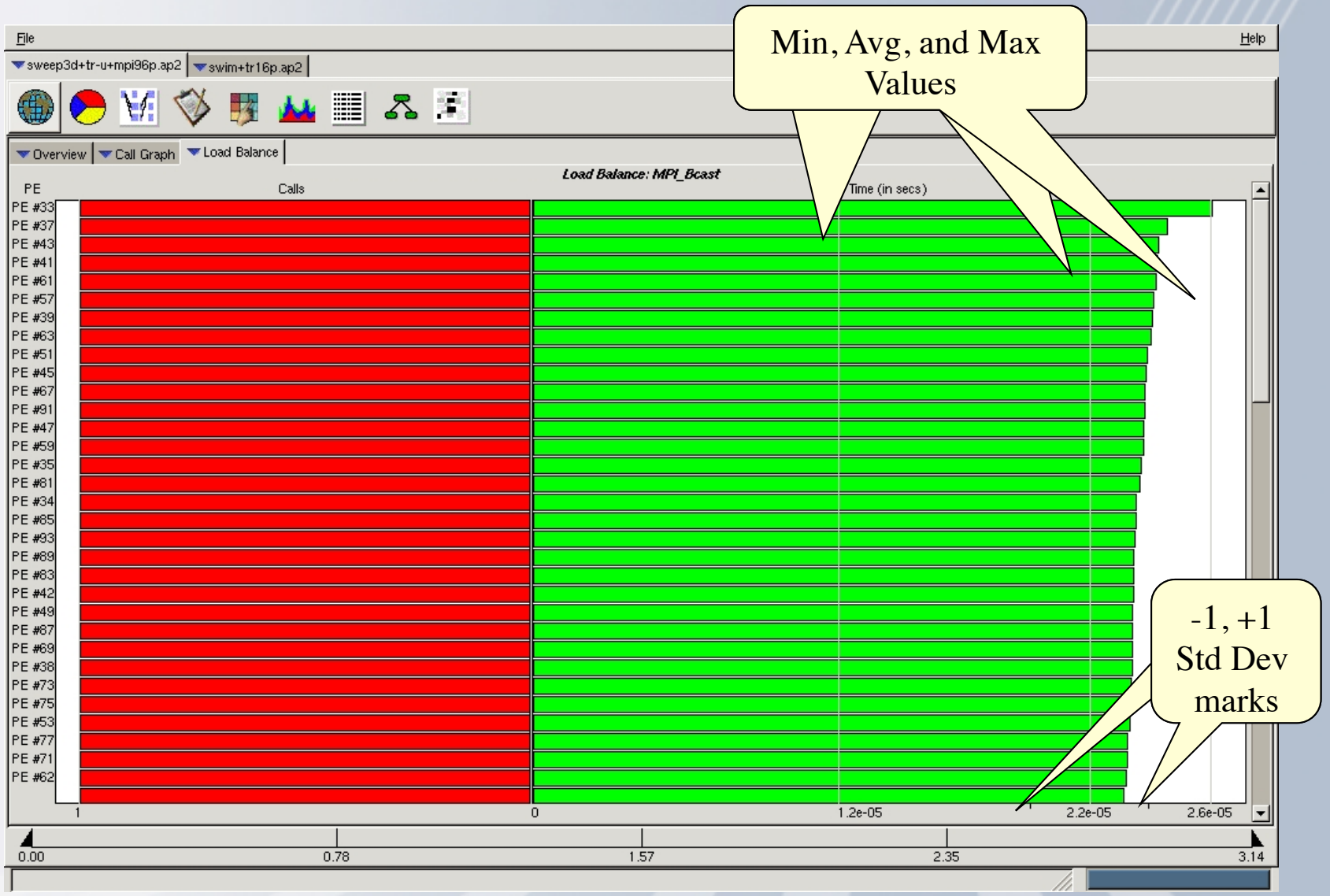
Call Tree View



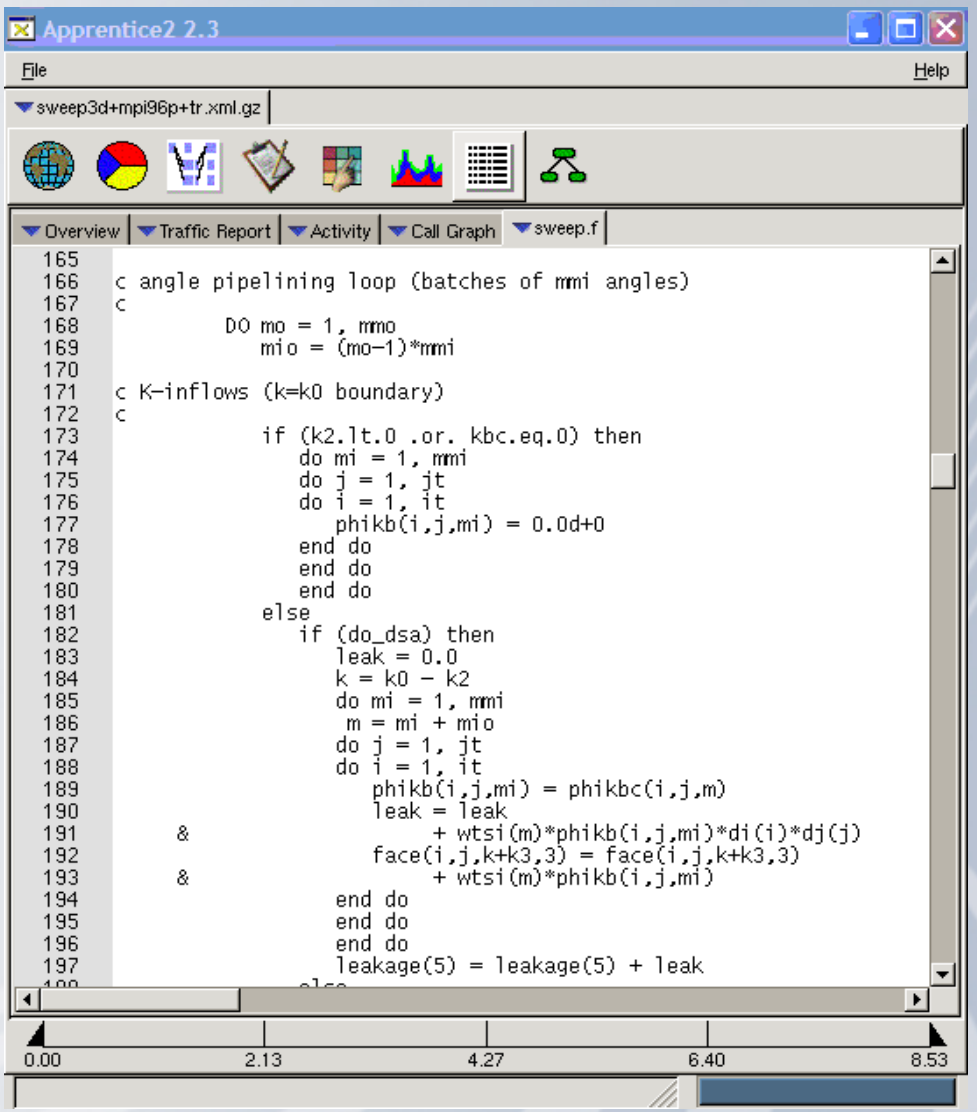
Call Tree View – Function List



Load Balance View (from Call Tree)



Source Mapping from Call Tree



Function Profile

File Help

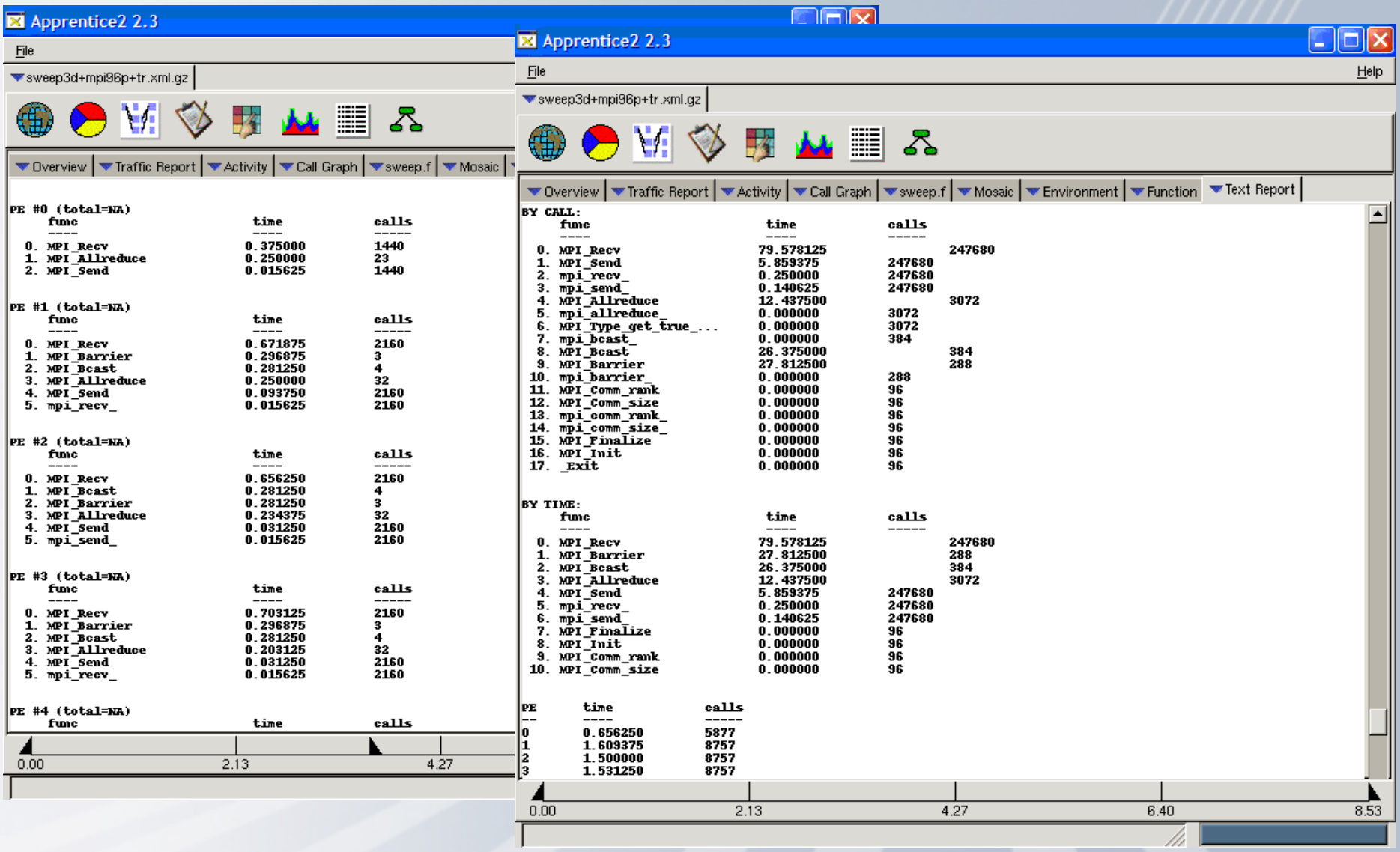
swim+iompi+1566td.ap2 | T+hw1+swp+io+mpi+48p.ap2

Overview | Function

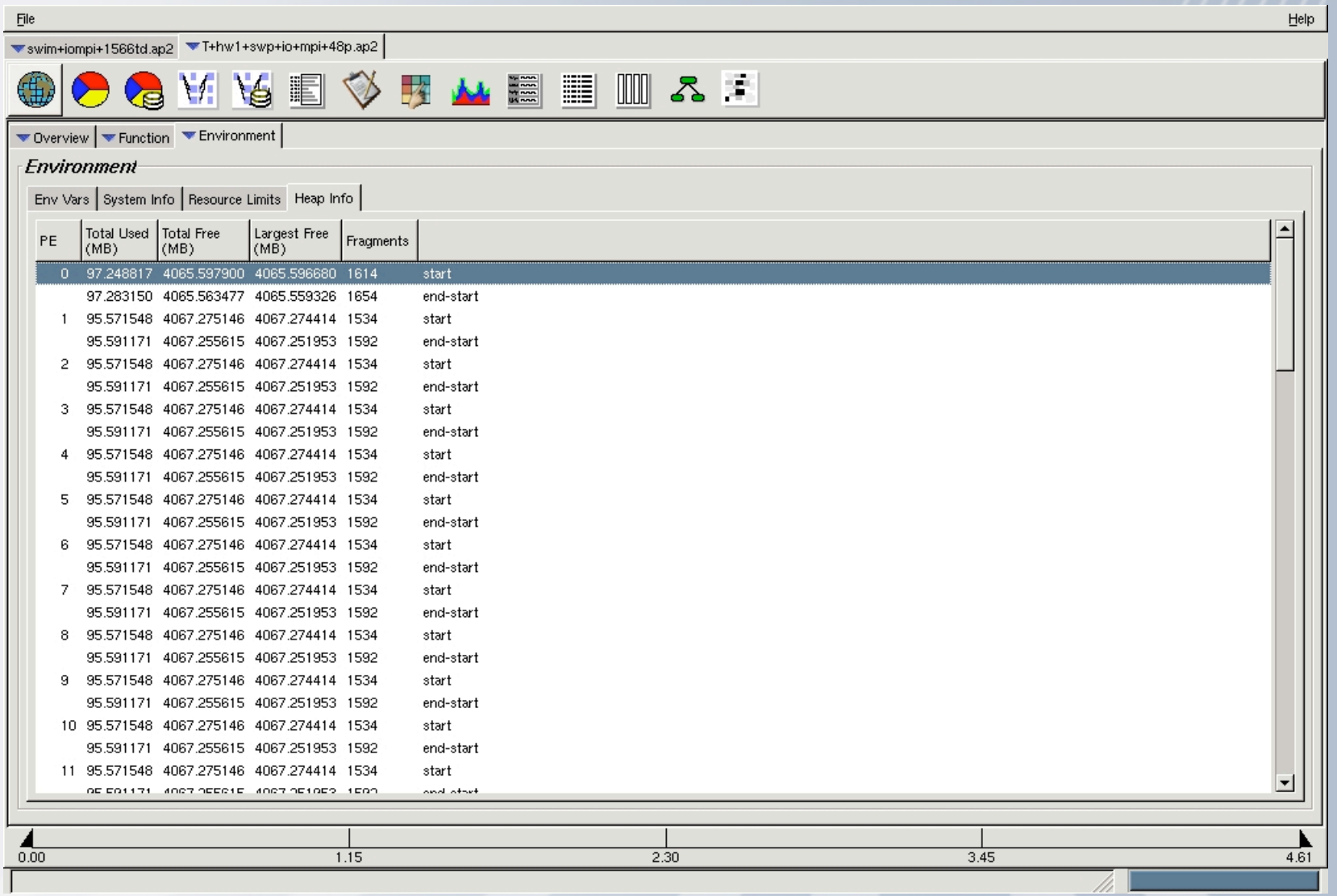
Time	Percent	Hits	Callsites	Imbalance %	Potential Savings	Function	Line	File
124.175511	63.29	576	1	5.63	0.15	sweep_	116	/lus/nid00036/ldr/Apps/sweep3d/sweep.f
40.211774	20.50	118080	1	23.40	0.25	mpi_recv_	0	==NA==
16.319527	8.32	48	1	48.26	0.30	exit	35	/notbackedup/users/rsrel/rs64.REL_1_4_33.060914.Thu/pe/computelibs/glibc/stdlib/exit.c
6.173236	3.15	1536	3	50.00	0.12	mpi_allreduce_	0	==NA==
2.760376	1.41	118080	1	17.58	0.01	mpi_send_	0	==NA==
2.250029	1.15	576	1	2.62	0.00	source_	18	/lus/nid00036/ldr/Apps/sweep3d/source.f
1.984620	1.01	144	1	2.59	0.00	mpi_barrier_	0	==NA==
0.867359	0.44	192	2	2.47	0.00	mpi_bcast_	0	==NA==
0.416231	0.21	576	1	2.60	0.00	flux_err_	17	/lus/nid00036/ldr/Apps/sweep3d/flux_err.f
0.382130	0.19	118080	2	10.98	0.00	snd_real_	135	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.237772	0.12	309	1	95.76	0.07	fwrite	36	/notbackedup/users/rsrel/rs64.REL_1_4_33.060914.Thu/pe/computelibs/glibc/libio/iofwrite.c
0.185067	0.09	118080	2	17.30	0.00	rcv_real_	164	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.067832	0.03	48	1	4.56	0.00	initialize_	42	/lus/nid00036/ldr/Apps/sweep3d/initialize.f
0.059407	0.03	48	1	4.99	0.00	initxs_	77	/lus/nid00036/ldr/Apps/sweep3d/initialize.f
0.041371	0.02	48	1	23.84	0.00	inner_	72	/lus/nid00036/ldr/Apps/sweep3d/inner.f
0.023948	0.01	8	1		0.00	fputc	35	/notbackedup/users/rsrel/rs64.REL_1_4_33.060914.Thu/pe/computelibs/glibc/libio/fputc.c
0.016902	0.01	68	1		0.00	getc	36	/notbackedup/users/rsrel/rs64.REL_1_4_33.060914.Thu/pe/computelibs/glibc/libio/getc.c
0.008104	0.00	4992	2	28.14	0.00	octant_	17	/lus/nid00036/ldr/Apps/sweep3d/octant.f
0.002457	0.00	576	1	18.79	0.00	global_real_max_	321	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.002083	0.00	48	1	69.55	0.00	MAIN_	72	/lus/nid00036/ldr/Apps/sweep3d/driver.f
0.001588	0.00	576	1	39.10	0.00	global_int_sum_	373	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.001393	0.00	48	1	10.23	0.00	inner_auto_	69	/lus/nid00036/ldr/Apps/sweep3d/inner_auto.f
0.000982	0.00	48	1	97.74	0.00	task_init_	24	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.000739	0.00	384	2	27.87	0.00	global_real_sum_	347	/lus/nid00036/ldr/Apps/sweep3d/mpi_stuff.f
0.000662	0.00	2	1		0.00	fopen	106	/notbackedup/users/rsrel/rs64.REL_1_4_33.060914.Thu/pe/computelibs/glibc/libio/iofopen.c
0.000499	0.00	48	1	7.54	0.00	initsnc_	175	/lus/nid00036/ldr/Apps/sweep3d/initialize.f

0.00 1.15 2.30 3.45 4.61

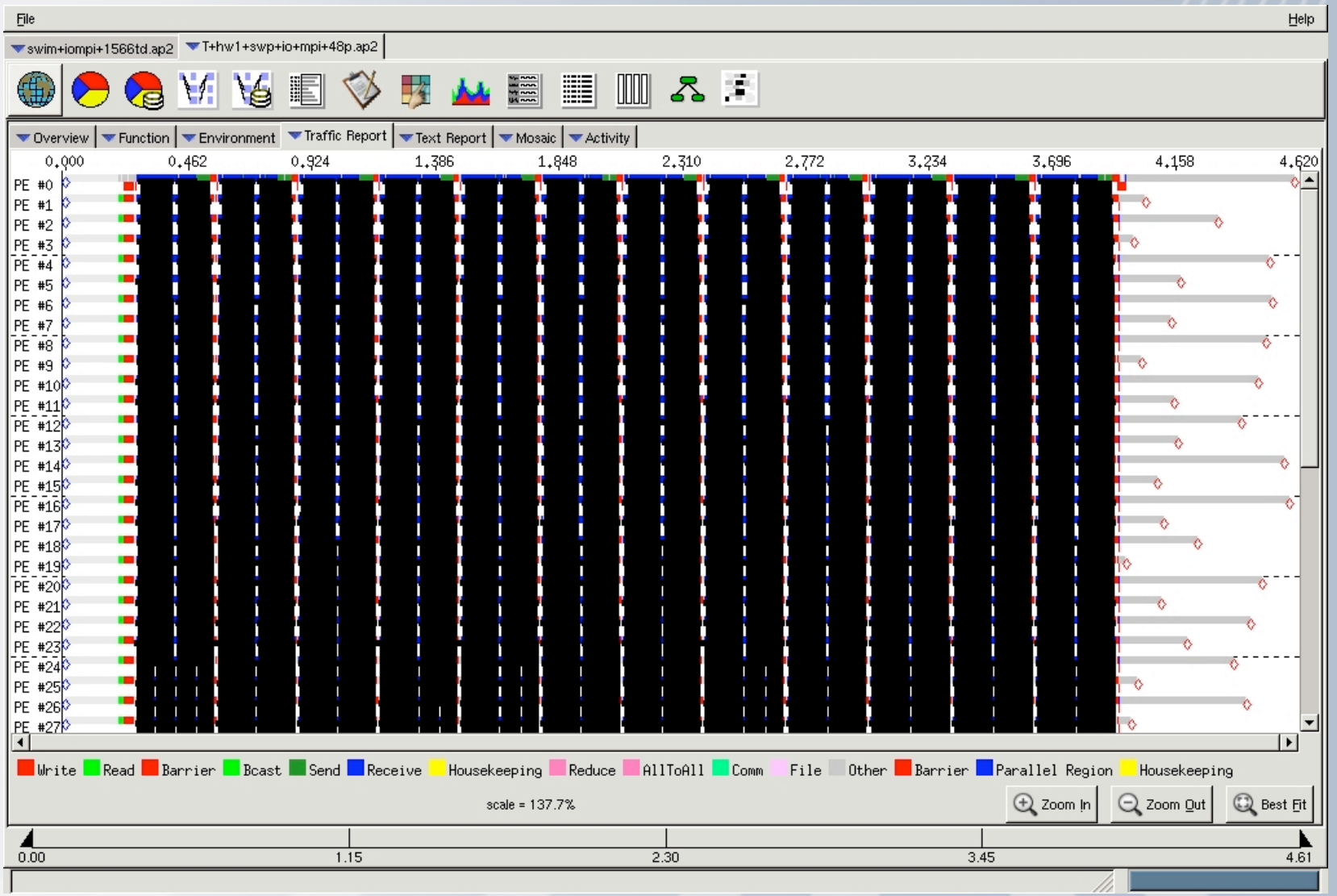
Distribution by PE, by Call, & by Time



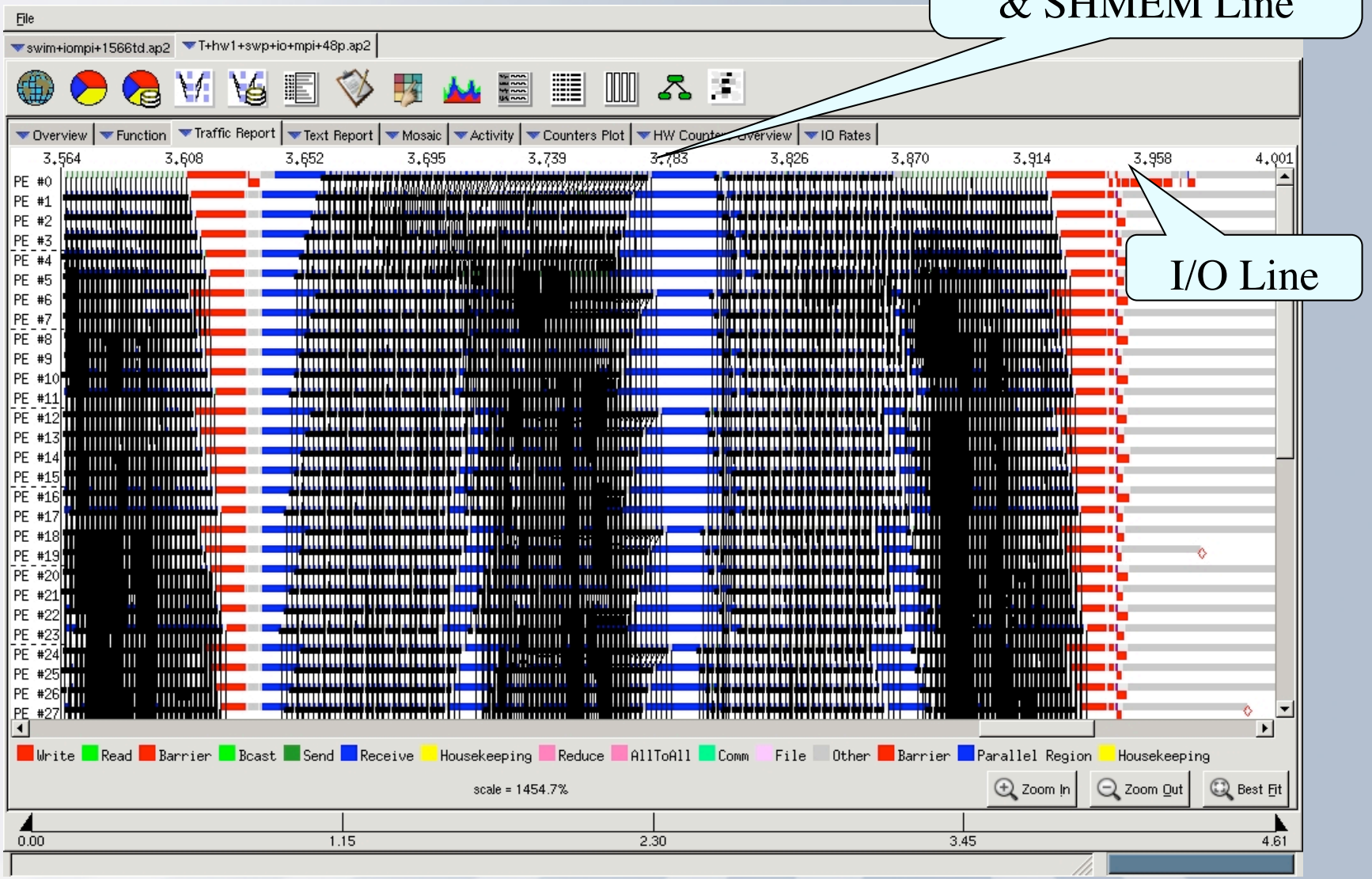
Environment & Execution Details



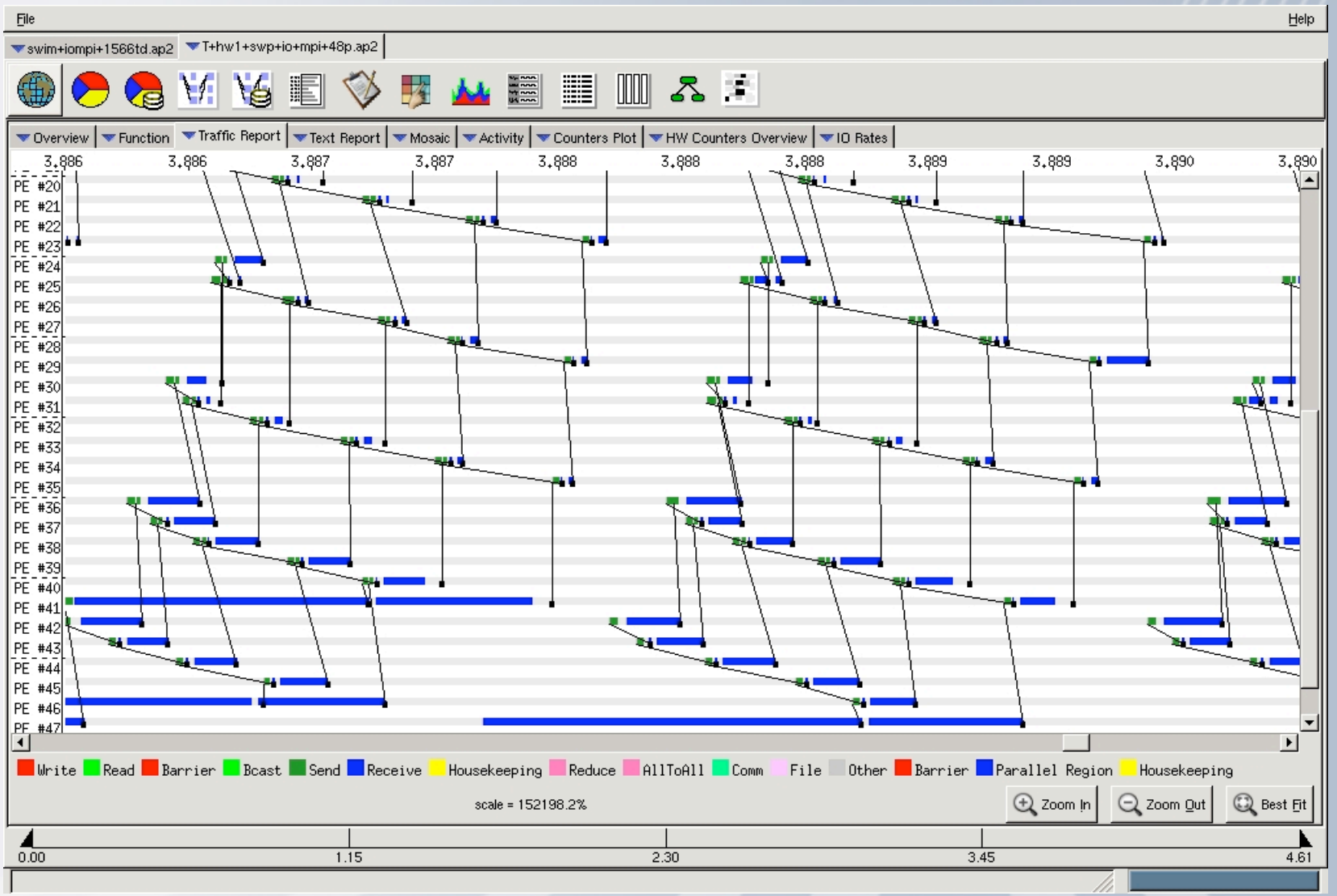
Time Line View (Sweep3D)



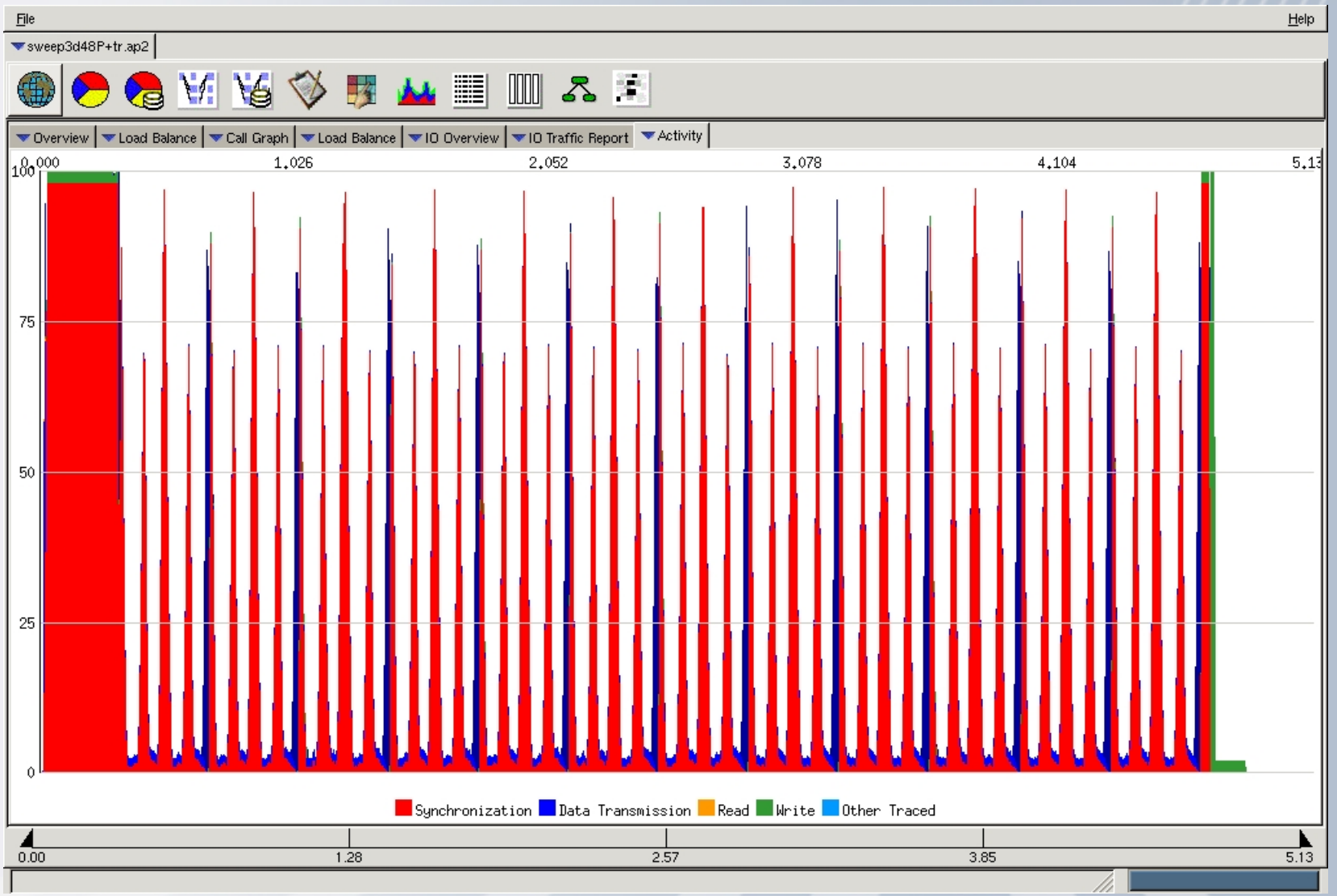
Time Line View (Zoom)



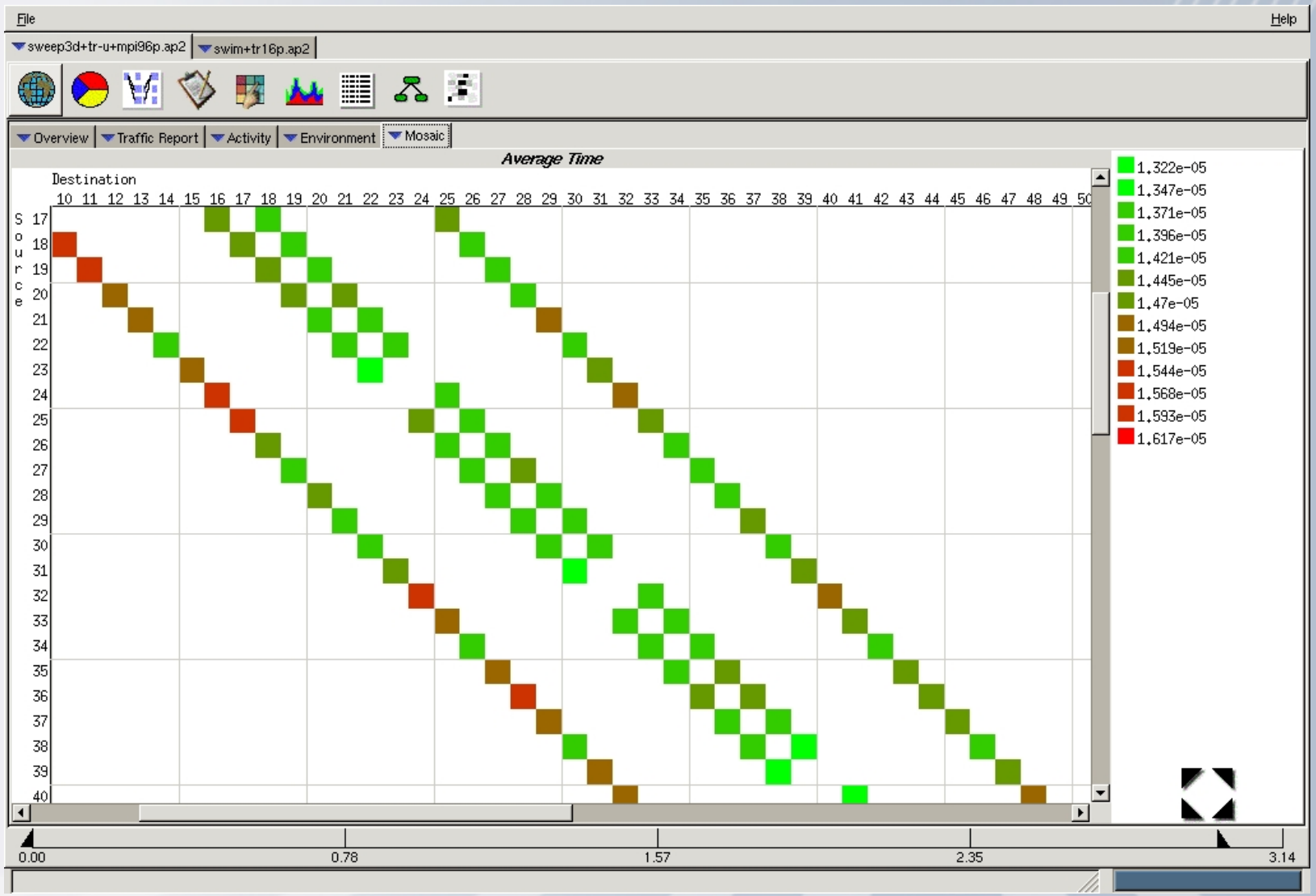
Time Line View (Fine Grain Zoom)



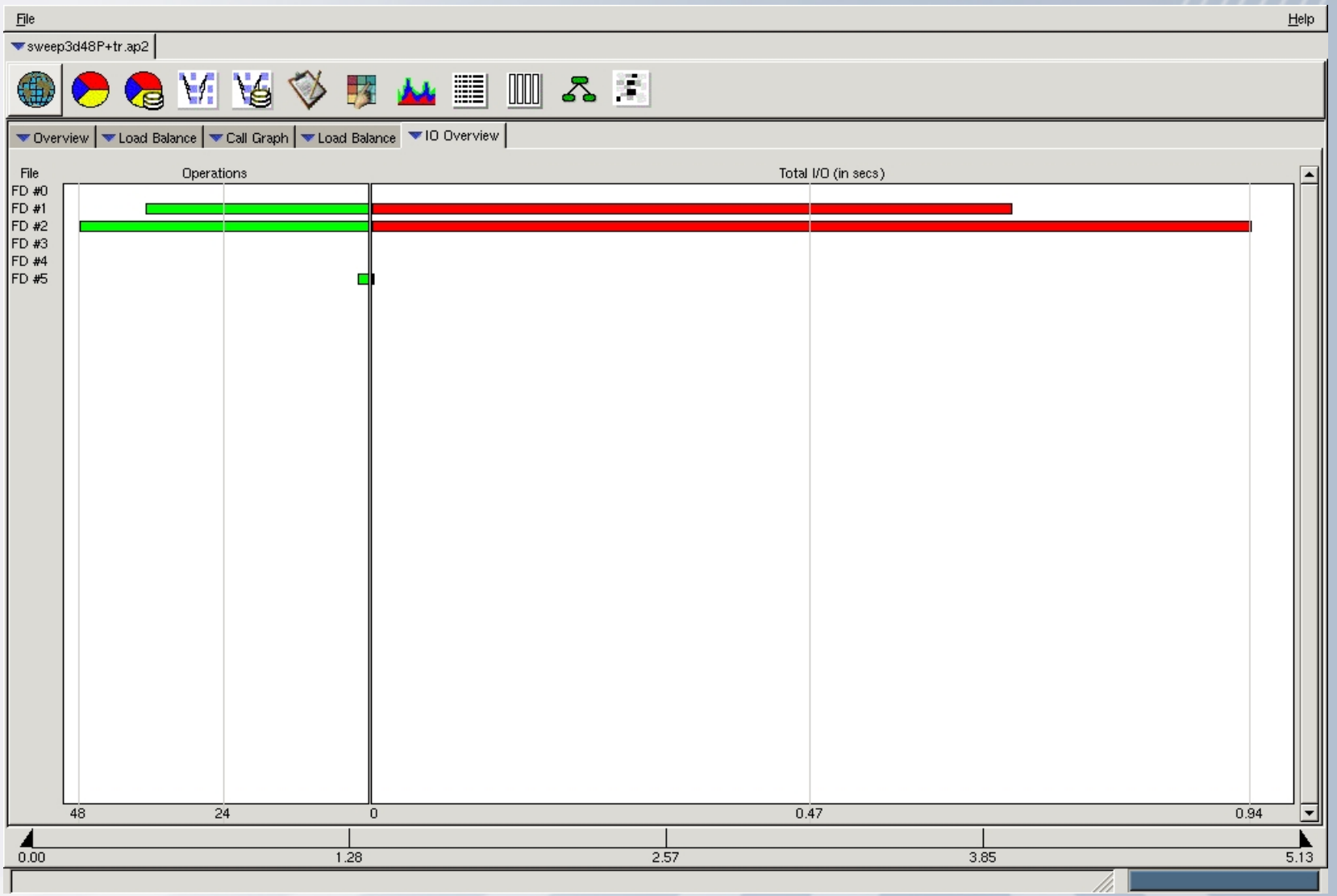
Activity View



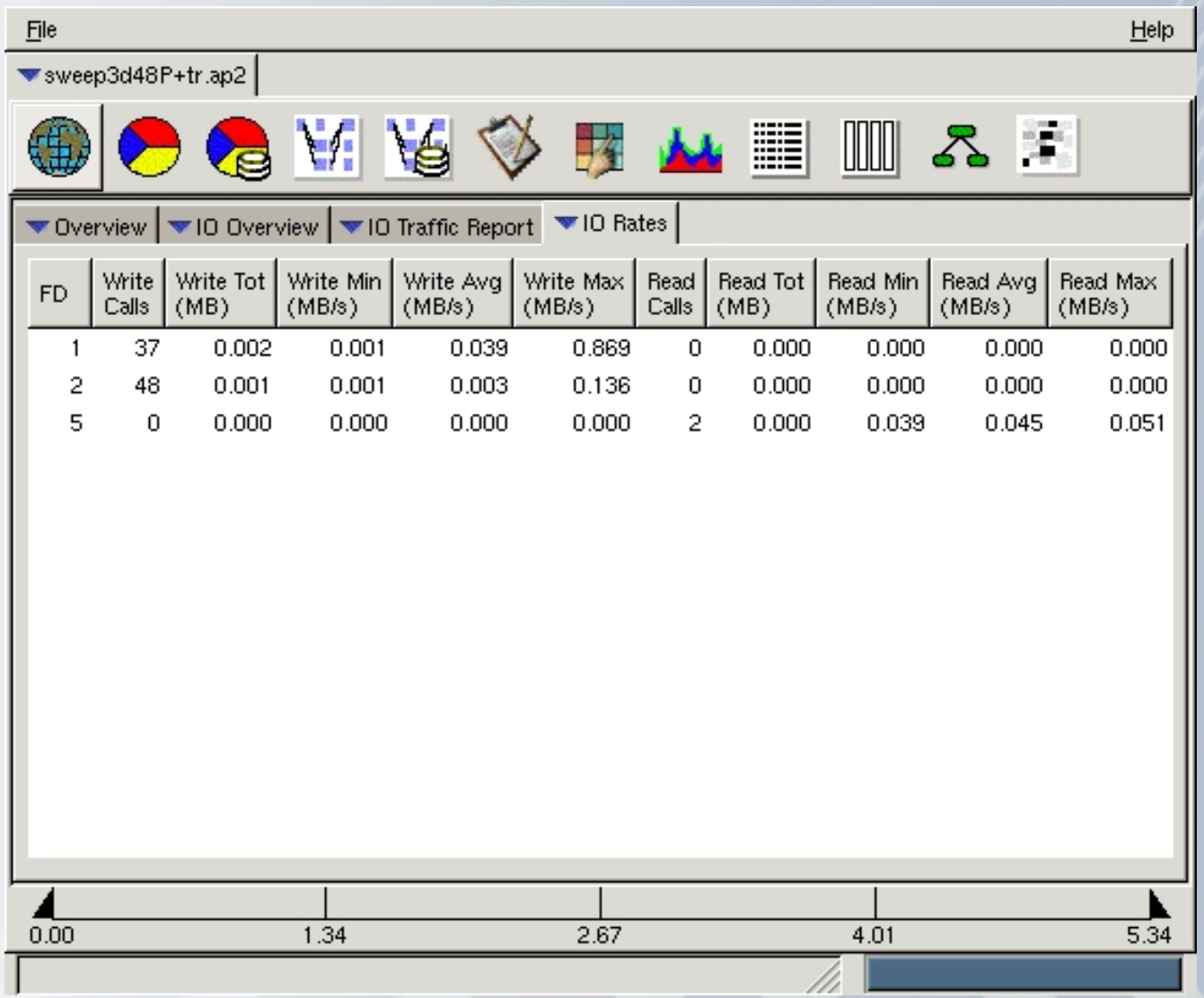
Pair-wise Communication



I/O Overview

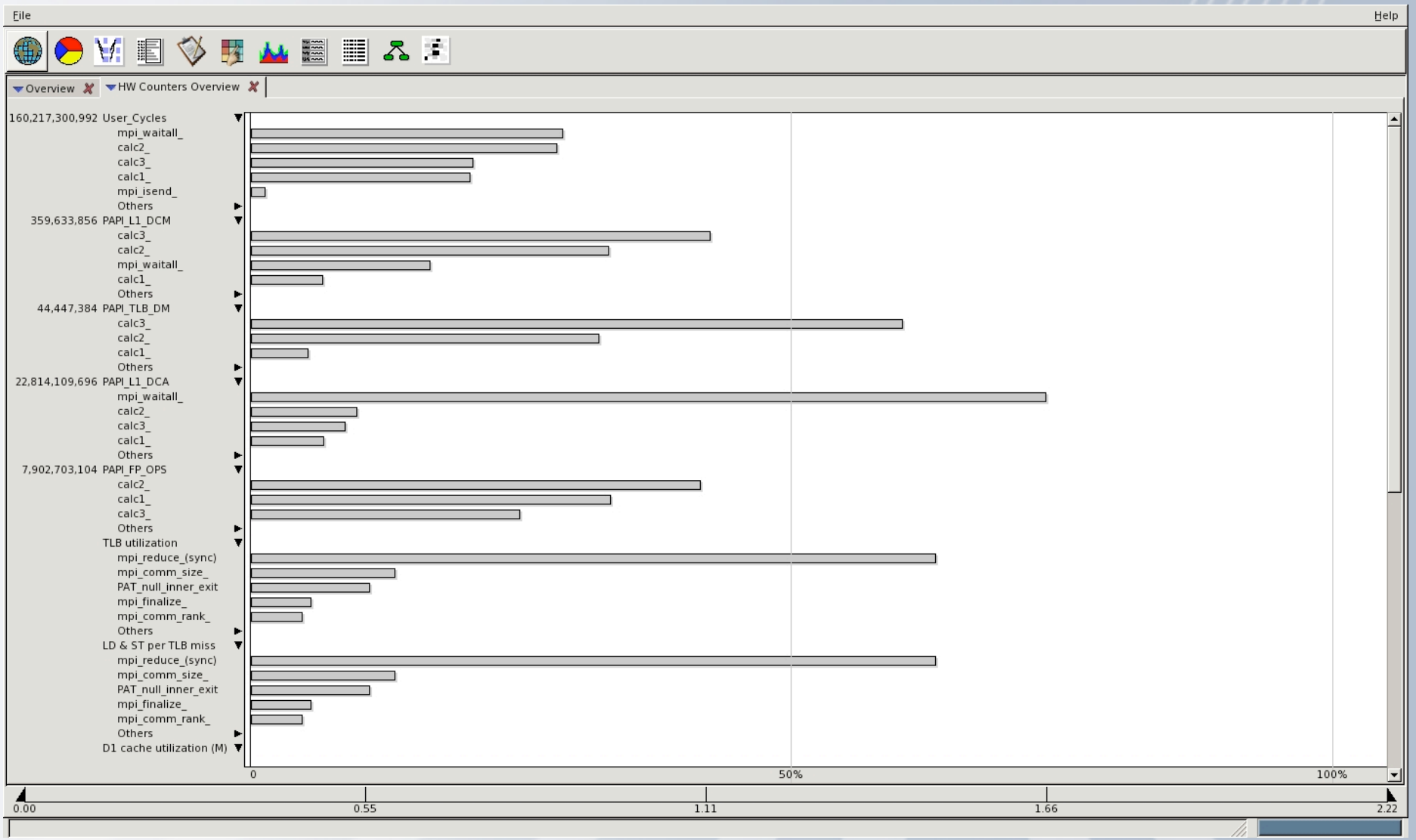

File
Help
sweep3d48P+tr.ap2
Overview
Load Balance
Call Graph
Load Balance
IO Overview
File
Operations
Total I/O (in secs)
FD #0
FD #1
FD #2
FD #3
FD #4
FD #5
48
24
0
0.47
0.94
0.00
1.28
2.57
3.85
5.13

I/O Rates

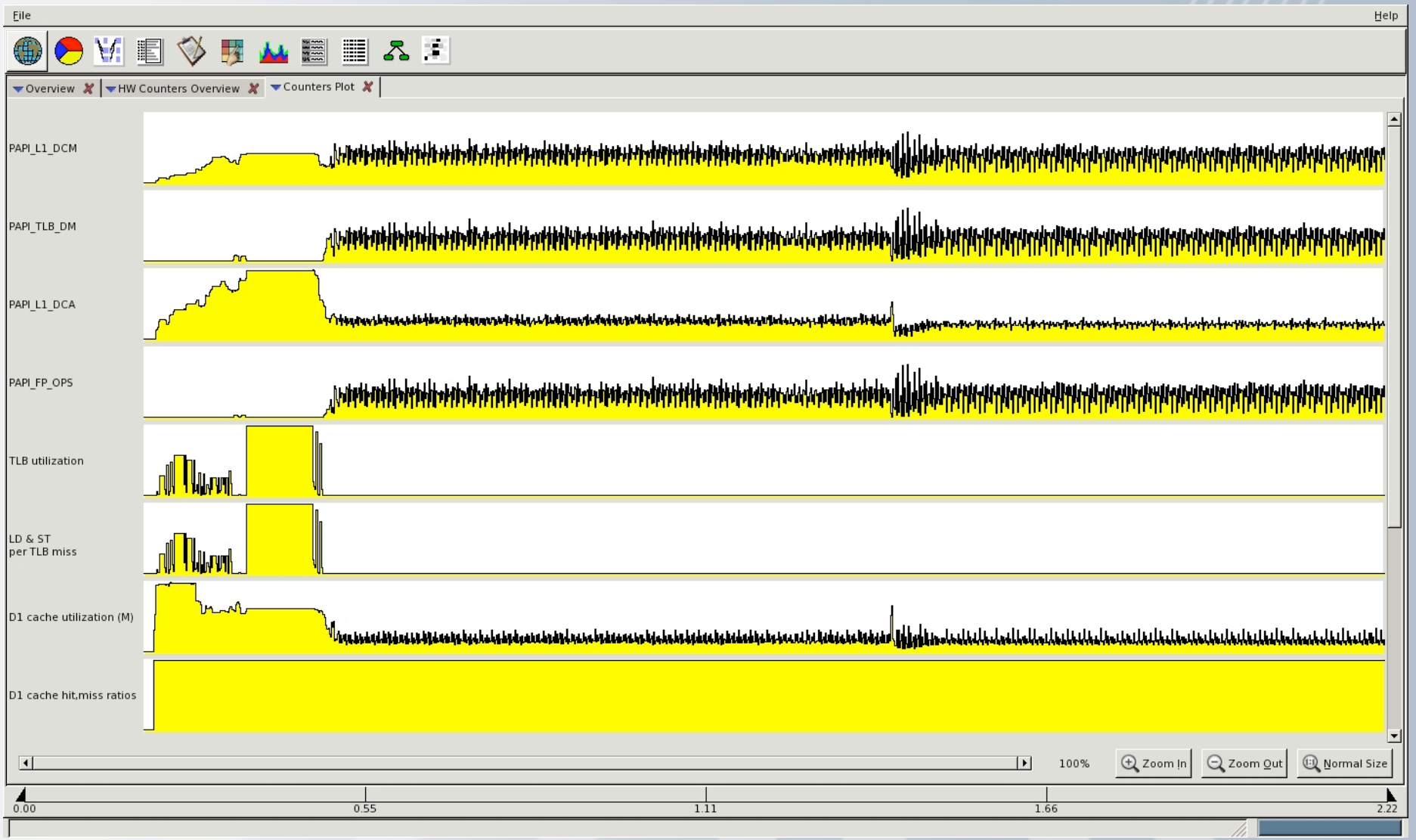


FD	Write Calls	Write Tot (MB)	Write Min (MB/s)	Write Avg (MB/s)	Write Max (MB/s)	Read Calls	Read Tot (MB)	Read Min (MB/s)	Read Avg (MB/s)	Read Max (MB/s)
1	37	0.002	0.001	0.039	0.869	0	0.000	0.000	0.000	0.000
2	48	0.001	0.001	0.003	0.136	0	0.000	0.000	0.000	0.000
5	0	0.000	0.000	0.000	0.000	2	0.000	0.039	0.045	0.051

Hardware Counters Overview



Hardware Counters Time Line



Controlling Performance File Size

- Performance files can be quite large. There are several run-time environment variables to keep data files down to reasonable sizes
- The particular run-time environment variables to use vary depending on the type of experiment being conducted
- Sampling:
  - PAT_RT_RECORD_PE
    - Collect trace for a subset of the PEs
  - PAT_RT_RECORD_THREAD
    - Collect trace for a subset of the threads
  - PAT_RT_INTERVAL
    - Specifies the interval, at which the instrumented program is sampled
  - PAT_RT_CALLSTACK
    - Limit the depth to trace the call stack
  - PAT_RT_HWPC
    - Avoid collecting hardware counters (unset)
  - PAT_RT_SIZE
    - The number of contiguous bytes in the text segment available for sampling
  - PAT_RT_WRITE_BUFFER_SIZE
    - Specifies the size, of a buffer that collects measurement data for a single thread

Controlling Trace File Size

- Tracing:
  - PAT_RT_CALLSTACK
    - Limit the depth to trace the call stack
  - PAT_RT_HWPC
    - Avoid collecting hardware counters (unset)
  - PAT_RT_RECORD_PE
    - Collect trace for a subset of the PEs
  - PAT_RT_RECORD_THREAD
    - Collect trace for a subset of the threads
  - PAT_RT_TRACE_FUNCTION_ARGS
    - Limit the number of function arguments to be traced
  - PAT_RT_TRACE_FUNCTION_LIMITS
    - Avoid tracing indicated functions
  - PAT_RT_TRACE_FUNCTION_MAX
    - Limit the maximum number of traces generated for all functions for a single process
  - PAT_RT_TRACE_THRESHOLD_PCT
    - Specifies a % of time threshold to enforce when executing in full trace mode
  - PAT_RT_TRACE_THRESHOLD_TIME
    - Specifies a time threshold to enforce when executing in full trace mode
- Use the limit built-in command for ksh(1) or csh(1) to control how much disk space the trace file can consume

Additional API Functions

- int PAT_state (int state)
  - State can have one of the following:
    - PAT_STATE_ON
    - PAT_STATE_OFF
    - PAT_STATE_QUERY
- int PAT_record (int state)
  - Controls the state for all threads on the executing PE. As a rule, use PAT_record() unless there is a need for different behaviors for sampling and tracing
    - int PAT_sampling_state (int state)
    - int PAT_tracing_state (int state)
- int PAT_trace_function (const void *addr, int state)
  - Activates or deactivates the tracing of the instrumented function
- int PAT_flush_buffer (void)

Parallel Performance Analysis and Visualization on the Cray XT

Questions / Comments

Thank You!

CSCS
July 13-15, 2009

CRAY
THE SUPERCOMPUTER COMPANY

Luiz DeRose (ldr@cray.com)