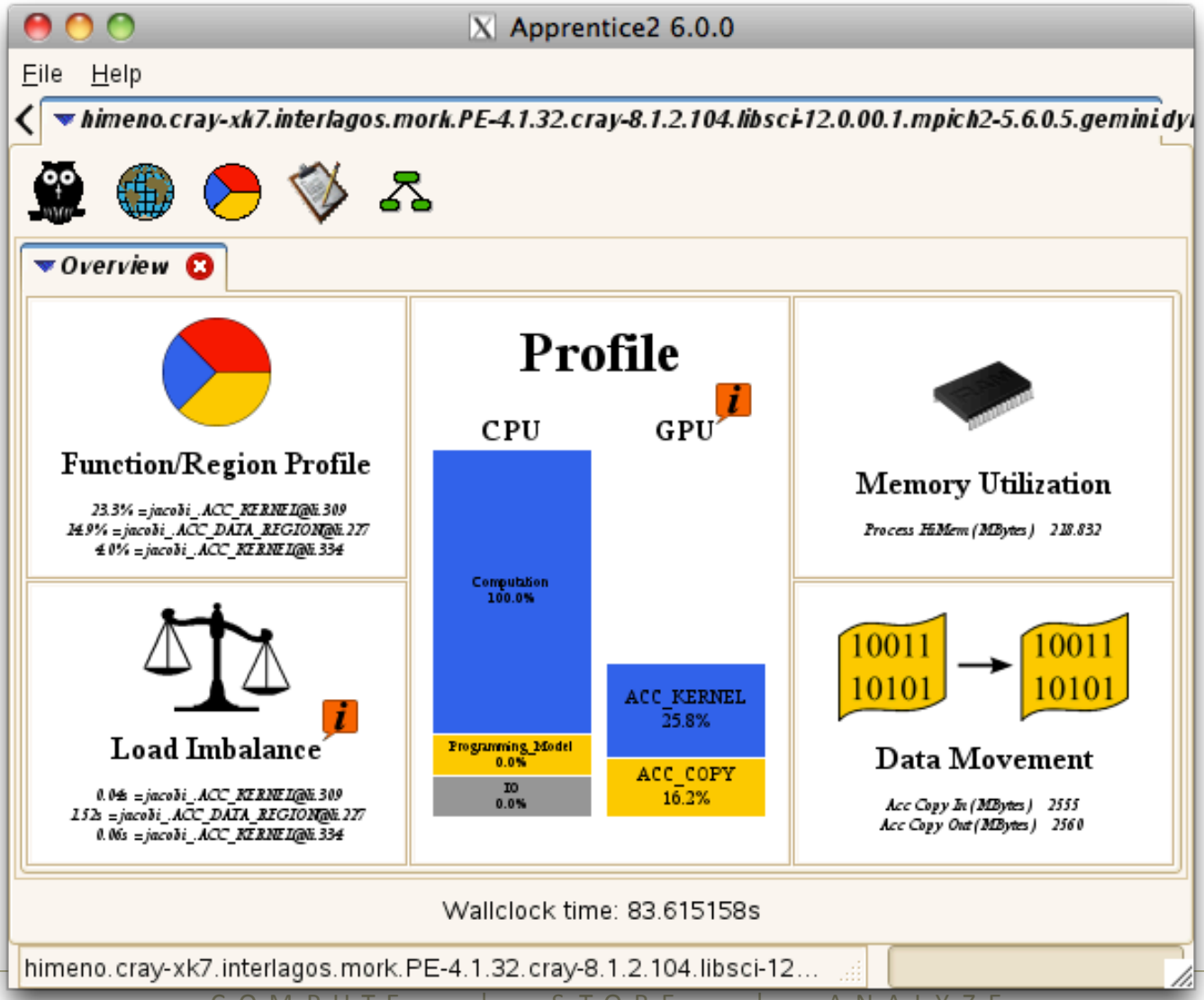
Cray Performance Measurement, Analysis and Porting Tools

Luiz DeRose
Sr. Principal Engineer, Programming Environments Director

Heidi Poxon
Technical Lead & Sr. Manager, Performance Tools

Cray Performance Tools Refresher



Copyright 2014 Cray Inc.

Safe Harbor Statement

This presentation may contain forward-looking statements that are based on our current expectations. Forward looking statements may include statements about our financial guidance and expected operating results, our opportunities and future potential, our product development and new product introduction plans, our ability to expand and penetrate our addressable markets and other statements that are not historical facts. These statements are only predictions and actual results may materially vary from those projected. Please refer to Cray's documents filed with the SEC from time to time concerning factors that could affect the Company and these forward-looking statements.

Copyright 2014 Cray Inc.

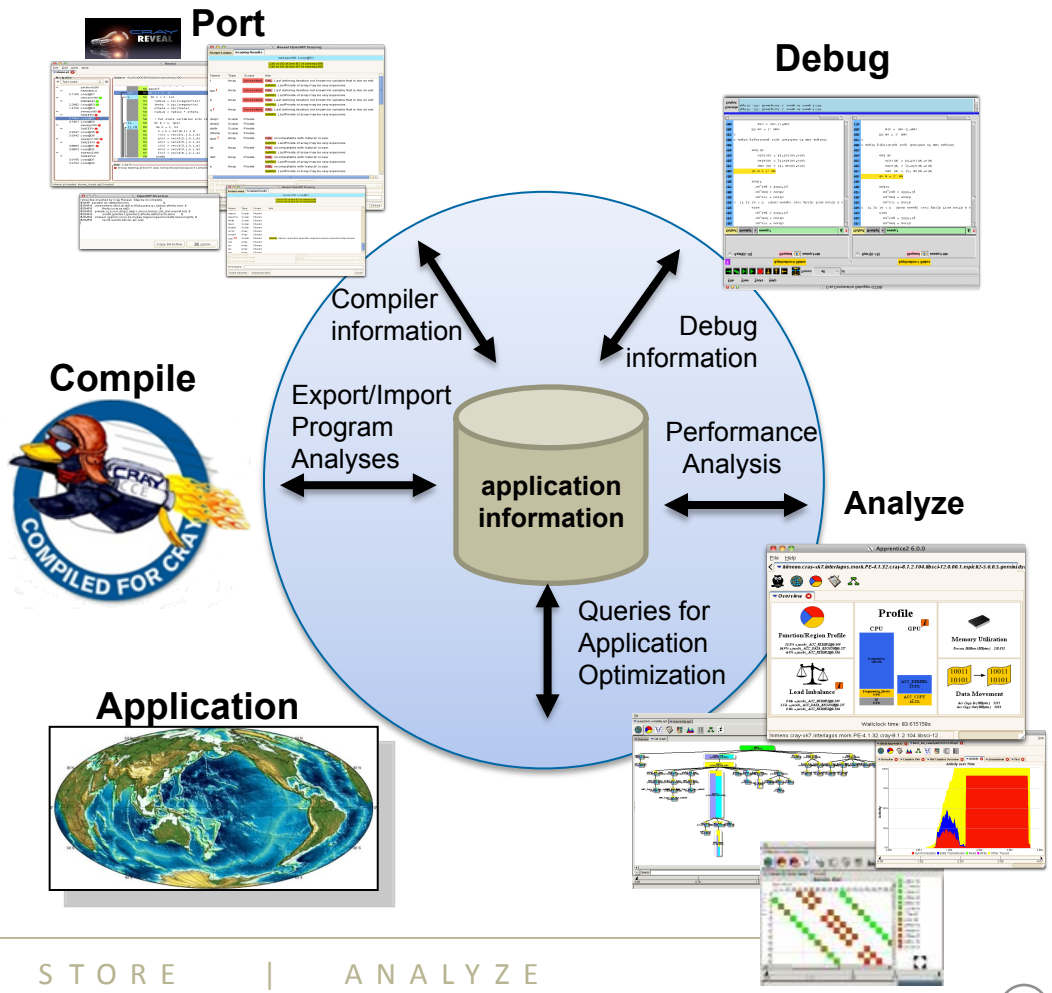
Agenda

- 10:20 – 10:30 Overview of Cray PE topics for the day
- 10:30 – 11:15 Cray Performance Tools refresher
- 11:15 – 12:00 lab (CrayPat-lite, loop stats, Rank reorder)
- 12:00 – 13:00 Lunch
- 13:00 – 14:00 Reveal
- 14:00 – 14:30 CCE and OpenACC update
- 14:30 – 15:00 lab with Reveal
- 15:00 – 15:20 Break
- 15:20 – 16:00 CCDB
- 16:00 – 17:00 find a bug using CCDB, general lab time

Copyright 2014 Cray Inc.

The Programming Environment Mission

- It is the role of the Programming Environment to **close the gap** between observed performance and achievable performance
- Support the **application development life cycle** by providing a **tightly coupled** environment with compilers, libraries, and tools that will **hide the complexity** of the system
  - Address issues of scale and complexity of HPC systems
  - Target **ease of use** with extended **functionality** and increased **automation**
  - Close **interaction with users**
    - For feedback targeting functionality enhancements



Copyright 2014 Cray Inc.

Cray Programming Environment Focus

- **Performance**
  - Help users maximize the cycles to the application
    - Address issues of scale and complexity of HPC systems
- **Programmability**
  - How do you get intuitive behavior and best performance with the least amount of effort
    - Provide the best environment to develop, debug, analyze, and optimize applications for production supercomputing
    - Provide programming environment consistency across Cray platforms

Copyright 2014 Cray Inc.

Recent Enhancements

- Cray Apprentice2 available for Linux, Mac and Windows with .dmg and .exe installers
  - Available in $CRAYPAT_ROOT/share/desktop_installers
- Router-aware MPI rank placement
- Simplified Interface to performance counters
- Program timeline
- I/O statistics in CrayPat-lite
- pat_build default is now pat_build –O apa

Copyright 2014 Cray Inc.

Two Interfaces to the Performance Tools

- **CrayPat** offers a wealth of performance measurement, analysis and presentation options for in-depth performance investigation and tuning assistance
- **CrayPat-lite** offers easy access to an application performance summary for users not familiar with the Cray performance tools or who may not be familiar with performance analysis
- **CrayPat classic and CrayPat-lite are designed to compliment each other**
  - Produce files with same format
  - Users familiar with CrayPat can easily switch back and forth between the two interfaces
  - CrayPat-lite users become familiar with reporting style also used with CrayPat

Copyright 2014 Cray Inc.

CrayPat-lite

- **Produces application performance statistics at the end of a job**
  - Focus is to offer a simplified interface to basic application performance information for users not familiar with the Cray performance tools and perhaps new to application performance analysis
  - Gives sites the option to enable/disable application performance data collection for all users for a period of time
- **Compliments "classic" perftools**
- **Provides a simple way to transition from perftools-lite to perftools to encourage further tool use for more in-depth performance analysis**

Copyright 2014 Cray Inc.

Using CrayPat-lite

Access light version of performance tools software

```
> module load perftools-lite
```

Build program

```
> make
```

a.out (instrumented program)

Run program (no modification to batch script)

```
aprun a.out
```

Condensed report to stdout
a.out*.rpt (same as stdout)
a.out*.ap2
MPICH_RANK_XXX files

Copyright 2014 Cray Inc.

Performance Statistics Available

- **Set of predefined experiments, enabled with the CRAYPAT_LITE environment variable**
  - Sample_profile
  - Event_profile
  - GPU
- **Job information**
  - Number of MPI ranks, ranks per node, number of threads
  - Wallclock
  - High memory water mark
  - Aggregate MFLOPS (CPU only)
  - I/O
- **Profile of top time consuming routines with load balance**
- **Observations**
- **Instructions on how to get more information**

Copyright 2014 Cray Inc.

CrayPat-lite Output Example

```
CrayPat/X:  Version 6.1.4.12457 Revision 12457 (xf 12277)  02/26/14 13:58:24
Experiment:                  lite  lite/sample_profile
Number of PEs (MPI ranks):   8164
Numbers of PEs per Node:       16  PEs on each of  510  Nodes
                                4  PEs on    1  Node
Numbers of Threads per PE:      1
Number of Cores per Socket:     8
Execution start time:  Fri Feb 28 23:06:31 2014
System name and speed:  hera2 2100 MHz

Wall Clock Time:    999.595275 secs
High Memory:            475.52 MBytes
MFLOPS (aggregate):  806112.33 M/sec
I/O Read Rate:           33.57 MBytes/Sec
I/O Write Rate:         215.40 MBytes/Sec

Table 1:  Profile by Function Group and Function (top 7 functions shown)
 Time%  |      Time  |    Imb.  |  Imb.  |    Calls  |Group
        |            |    Time  | Time%  |           | Function
        |            |          |        |           |  PE=HIDE
 100.0% | 101.961423 |       -- |     -- | 5315211.9 |Total
|-------------------------------------------------------------------------------------
|  92.5% |  94.267451 |       -- |     -- | 5272245.9 |USER
||------------------------------------------------------------------------------------
||  75.8% |  77.248585 | 2.356249 |   3.0% |    1001.0 |LAMMPS_NS::PairLJCut::compute
||   6.5% |   6.644545 | 0.105246 |   1.6% |      51.0 |LAMMPS_NS::Neighbor::half_bin_newton
||   4.1% |   4.131842 | 0.634032 |  13.5% |       1.0 |LAMMPS_NS::Verlet::run
||   3.8% |   3.841349 | 1.241434 |  24.8% | 5262868.9 |LAMMPS_NS::Pair::ev_tally
||   1.3% |   1.288463 | 0.181268 |  12.5% |    1000.0 |LAMMPS_NS::FixNVE::final_integrate
||====================================================================================
|   7.0% |   7.110931 |       -- |     -- |   42637.0 |MPI
||------------------------------------------------------------------------------------
||   4.8% |   4.851309 | 3.371093 |  41.6% |   12267.0 |MPI_Send
||   1.5% |   1.536106 | 2.592504 |  63.8% |   12267.0 |MPI_Wait
|====================================================================================
```


Copyright 2014 Cray Inc.

The Cray Performance Analysis Framework

- **Supports traditional post-mortem performance analysis**
  - Automatic identification of performance problems
    - Indication of causes of problems
    - Suggestions of modifications for performance improvement
  - pat_build: provides automatic instrumentation
  - CrayPat run-time library collects measurements (transparent to the user)
  - pat_report performs analysis and generates text reports
  - pat_help: online help utility
  - Cray Apprentice2: graphical visualization tool

Copyright 2014 Cray Inc.

Steps to Using CrayPat "classic"

Access performance tools software

```
> module load perftools
```

Build program, retaining .o files

```
> make
```

```
a.out
```

Instrument binary

```
> pat_build –O apa a.out
```

```
a.out+pat
```

Modify batch script and run program

```
aprun a.out+pat
```

```
a.out+pat*.xf
```

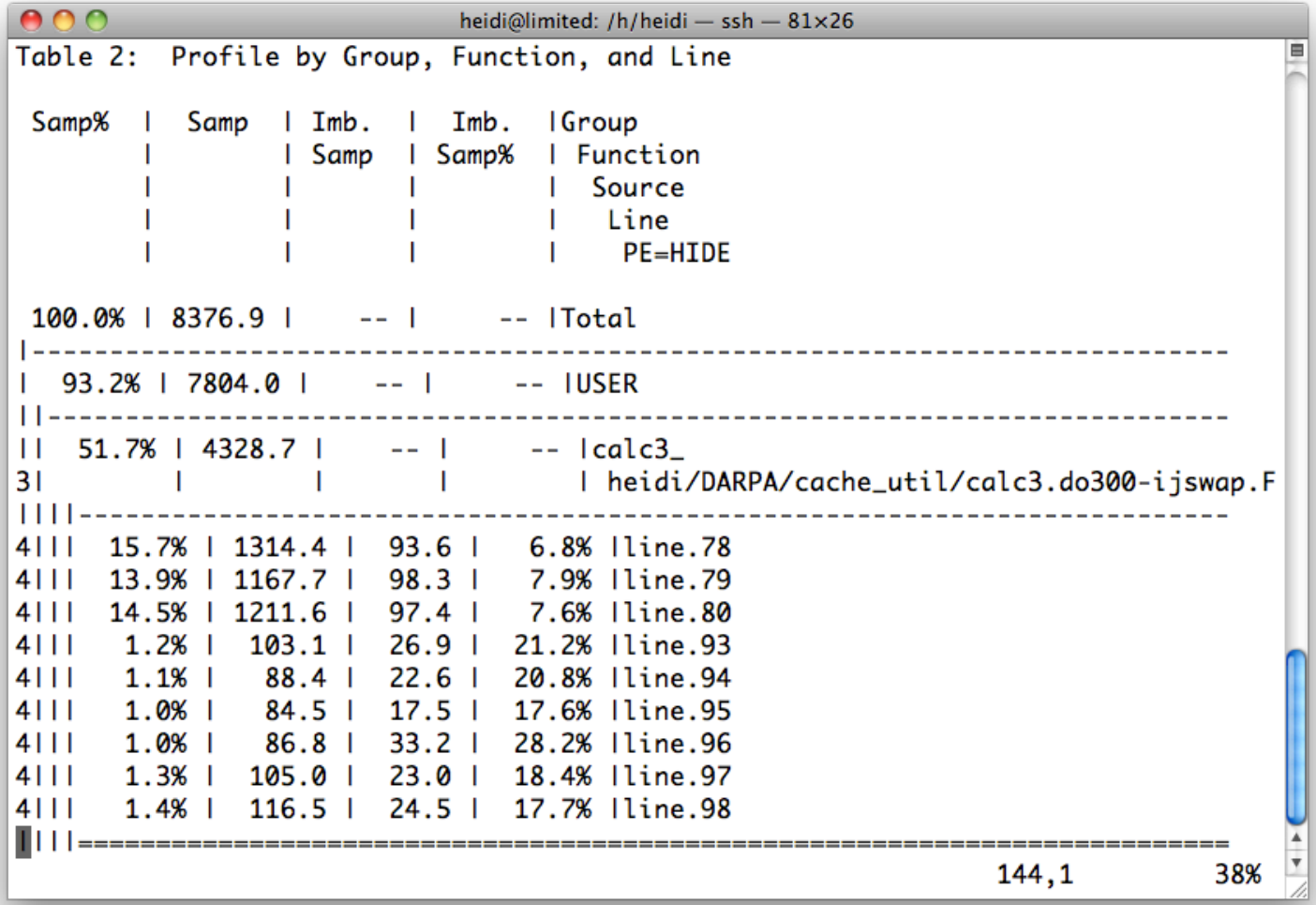
Process raw performance data and create report

```
> pat_report a.out+pat*.xf
```

```
a.out+pat*.ap2
Text report to stdout
a.out+pat*.apa
MPICH_RANK_XXX
```


Copyright 2014 Cray Inc.

Sampling with Line Number information



Copyright 2014 Cray Inc.

APA File Example

```
#  You can edit this file, if desired, and use it
#   to reinstrument the program for  tracing like this:
#
#          pat_build -O standard.cray-
     xt.PE-2.1.56HD.pgi-8.0.amd64.pat-5.0.0.2-0apa.512.quad.cores.seal.
     090405.1154.mpi.pat_rt_exp=default.pat_rt_hwpc=none.14999.xf.xf.apa
#
#  These suggested trace options are based on data from:
#
#    /home/users/malice/pat/Runs/Runs.seal.pat5001.2009Apr04/./pat.quad/
     homme/standard.cray-xt.PE-2.1.56HD.pgi-8.0.amd64.pat-5.0.0.2-0apa.
     512.quad.cores.seal.
     090405.1154.mpi.pat_rt_exp=default.pat_rt_hwpc=none.14999.xf.xf.cdb
# ------------------------------------------------------------------------

#       HWPC group to collect by default.

  -Drtenv=PAT_RT_HWPC=1  # Summary with TLB metrics.

# ------------------------------------------------------------------------

#       Libraries to trace.

  -g mpi

# ------------------------------------------------------------------------

#       User-defined functions to trace, sorted by % of samples.

#       The way these functions are filtered can be controlled with
#       pat_report options (values used for this file are shown):
#
#       -s apa_max_count=200    No more than 200 functions are listed.
#       -s apa_min_size=800     Commented out if text size < 800 bytes.
#       -s apa_min_pct=1        Commented out if it had < 1% of samples.
#       -s apa_max_cum_pct=90   Commented out after cumulative 90%.

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

  -w  # Enable tracing of user-defined functions.
      # Note: -u should NOT be specified as an additional option.
```

```
# 31.29%  38517 bytes
          -T prim_advance_mod_preq_advance_exp_

# 15.07%  14158 bytes
          -T prim_si_mod_prim_diffusion_

#  9.76%  5474 bytes
          -T derivative_mod_gradient_str_nonstag_

. . .

#  2.95%  3067 bytes
          -T forcing_mod_apply_forcing_

#  2.93%  118585 bytes
          -T column_model_mod_applycolumnmodel_

#  Functions below this point account for less than 10% of samples.

#  0.66%  4575 bytes
#          -T bndry_mod_bndry_exchangev_thsave_time_

#  0.10%  46797 bytes
#          -T baroclinic_inst_mod_binst_init_state_

#  0.04%  62214 bytes
#          -T prim_state_mod_prim_printstate_

. . .
#  0.00%  118 bytes
#          -T time_mod_timelevel_update_

# ------------------------------------------------------------------------

  -o preqx.cray-xt.PE-2.1.56HD.pgi-8.0.amd64.pat-5.0.0.2.x+apa
     # New instrumented program.

  /.AUTO/cray/css.pe_tools/malice/craypat/build/pat/2009Apr03/2.1.56HD/
     amd64/homme/pgi7pat-5.0.0.2/homme/2005Dec08/build.Linux/preqx.cray-
     xt.PE-2.1.56HD.pgi-8.0.amd64.pat-5.0.0.2.x  # Original program.
```


Copyright 2014 Cray Inc.

CrayPat Runtime Options

- **Runtime controlled through PAT_RT_XXX environment variables**
- **See intro_craypat(1) man page**
- **Examples of control**
  - Enable full trace
  - Change number of data files created
  - Enable collection of HW counters
  - Enable collection of network counters
  - Enable tracing filters to control trace file size (max threads, max call stack depth, etc.)

Copyright 2014 Cray Inc.

Example Runtime Environment Variables

- **Optional timeline view of program available**
  - export PAT_RT_SUMMARY=0
  - View trace file with Cray Apprentice^2
- **Write 1 file per node:**
  - export PAT_RT_EXPFILE_MAX=0
- **Request hardware performance counter information:**
  - export PAT_RT_PERFCTR=<HW counter group or event(s)>
  - Can specify individual events or predefined groups

Copyright 2014 Cray Inc.

Generating Profile from APA

- **Instrument application for further analysis (a.out+apa)**

```
% pat_build -O <apafile>.apa
```

- **Run application**

```
% aprun … a.out+apa   (or  qsub <apa script>)
```

- **Generate text report and visualization file (.ap2)**

```
% pat_report -o my_text_report.txt [<datafile>.xf |
  <datadir>]
```

- **View report in text and/or with Cray Apprentice^2**

```
% app2 <datafile>.ap2
```


Copyright 2014 Cray Inc.

Files Generated and the Naming Convention

File Suffix	Description
a.out+pat	Program instrumented for data collection
a.out…s.xf	Raw data for sampling experiment, available after application execution
a.out…t.xf	Raw data for trace (summarized or full) experiment, available after application execution
a.out…st.ap2	Processed data, generated by pat_report, contains application symbol information
a.out…s.apa	Automatic profiling pnalysis template, generated by pat_report (based on pat_build –O apa experiment)
a.out+apa	Program instrumented using .apa file
MPICH_RANK_ORDER.Custom	Rank reorder file generated by pat_report from automatic grid detection an reorder suggestions

Copyright 2014 Cray Inc.

Performance Counters

- **Cray supports raw counters, derived metrics and thresholds for:**
  - Processor
  - Network
  - Accelerator
  - Power
- **Predefined groups**
  - Groups together suggested counters for experiments
- **Single interface to access counters (PAT_RT_PERFCTR environment variable)**
- **PAPI with Cray custom components for network, uncore, power (available to 3rd party tool developers)**

Copyright 2014 Cray Inc.

Example: HW counter data and Derived Metrics

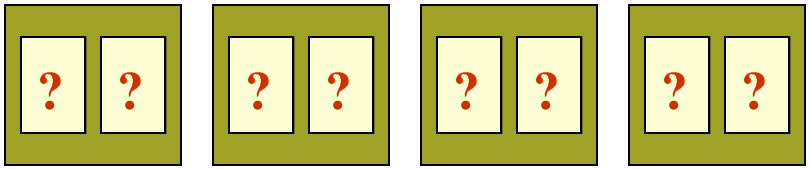
```
  PAPI_TLB_DM  Data translation lookaside buffer misses
  PAPI_L1_DCA  Level 1 data cache accesses
  PAPI_FP_OPS  Floating point operations
  DC_MISS      Data Cache Miss
  User_Cycles  Virtual Cycles
==========================================================================
USER
--------------------------------------------------------------------------
  Time%                                              98.3%
  Time                                           4.434402 secs
  Imb.Time                                             -- secs
  Imb.Time%                                            --
  Calls                          0.001M/sec         4500.0 calls
  PAPI_L1_DCM                   14.820M/sec       65712197 misses
  PAPI_TLB_DM                    0.902M/sec        3998928 misses
  PAPI_L1_DCA                  333.331M/sec     1477996162 refs
  PAPI_FP_OPS                  445.571M/sec     1975672594 ops
  User time (approx)             4.434 secs    11971868993 cycles  100.0%Time
  Average Time per Call                             0.000985 sec
  CrayPat Overhead : Time         0.1%
  HW FP Ops / User time        445.571M/sec     1975672594 ops    4.1%peak(DP)
  HW FP Ops / WCT              445.533M/sec
  Computational intensity         0.17 ops/cycle     1.34 ops/ref
  MFLOPS (aggregate)           1782.28M/sec
  TLB utilization               369.60 refs/miss    0.722 avg uses
  D1 cache hit,miss ratios       95.6% hits          4.4% misses
  D1 cache utilization (misses)  22.49 refs/miss    2.811 avg hits
==========================================================================
```

PAT_RT_HWPC=1
Flat profile data
Raw counts
Derived metrics

Copyright 2014 Cray Inc.

MPI Rank Reorder

- MPI rank placement with environment variable



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

Copyright 2014 Cray Inc.

MPI Rank Reorder (cont'd)

- Non-default MPI rank placements are useful when point-to-point communication consumes significant fraction of the program time, and there is a significant load imbalance
- Performance report contains a prioritized list of placement scenarios and includes instructions on how to choose one of the placements for subsequent program execution
- Custom placement files automatically generated, user just chooses one
- Utilities available to create MPI rank placements for applications with grid or lattice topologies.

Copyright 2014 Cray Inc.

Automatic Communication Grid Detection

- **Cray performance tools produce a custom rank order if it's beneficial based on grid size, grid order and cost metric**
- **Heuristics available for:**
  - MPI sent message statistics
  - User time (time spent in user functions) – can be used for PGAS codes
  - Hybrid of sent message and user time)
- **Summarized findings in report**
- **Available with sampling or tracing**
- **Describe how to re-run with custom rank order**

Copyright 2014 Cray Inc.

MPI Rank Order Observations

Table 1: Profile by Function Group and Function

Time%	Time	Imb. Time	Imb. Time%	Calls	Group Function PE=HIDE
100.0%	463.147240	--	--	21621.0	Total
52.0%	240.974379	--	--	21523.0	MPI
47.7%	221.142266	36.214468	14.1%	10740.0	mpi_recv
4.3%	19.829001	25.849906	56.7%	10740.0	MPI_SEND
43.3%	200.474690	--	--	32.0	USER
41.0%	189.897060	58.716197	23.6%	12.0	sweep_
1.6%	7.579876	1.899097	20.1%	12.0	source_
4.7%	21.698147	--	--	39.0	MPI_SYNC
4.3%	20.091165	20.005424	99.6%	32.0	mpi_allreduce_(sync)
0.0%	0.000024	--	--	27.0	SYSCALL

Copyright 2014 Cray Inc.

MPI Rank Order Observations (2)

MPI Grid Detection:

There appears to be point-to-point MPI communication in a **96 X 8 grid pattern**. The **52% 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 Custom rank order from the following table.

Rank Order	On-Node Bytes/PE	On-Node Bytes/PE% of Total Bytes/PE	MPICH_RANK_REORDER_METHOD
Custom	2.385e+09	95.55%	3
SMP	1.880e+09	75.30%	1
Fold	1.373e+06	0.06%	2
RoundRobin	0.000e+00	0.00%	0

Copyright 2014 Cray Inc.

MPICH_RANK_ORDER File

```
# The 'Custom' rank order in this file targets nodes with multi-core
# processors, based on Sent Msg Total Bytes collected for:
#
# Program:      /lus/nid00030/heidi/sweep3d/mod/sweep3d.mpi
# Ap2 File:     sweep3d.mpi+pat+27054-89t.ap2
# Number PEs:   48
# Max PEs/Node: 4
#
# To use this file, make a copy named MPICH_RANK_ORDER, and set the
# environment variable MPICH_RANK_REORDER_METHOD to 3 prior to
# executing the program.
#
# The following table lists rank order alternatives and the grid_order
# command-line options that can be used to generate a new order.
…
```


Copyright 2014 Cray Inc.

Auto-Generated MPI Rank Order File

```
# The 'USER_Time_hybrid' rank order in this file targets nodes with multi-core
# processors, based on Sent Msg Total Bytes collected for:
#
# Program:      /lus/nid00023/malice/craypat/WORKSHOP/bh2o-demo/Rank/sweep3d/src/sweep3d
# Ap2 File:     sweep3d.gmpi-u.ap2
# Number PEs:   768
# Max PEs/Node: 16
#
# To use this file, make a copy named MPICH_RANK_ORDER, and set the
# environment variable MPICH_RANK_REORDER_METHOD to 3 prior to
# executing the program.
#
0,532,64,564,32,572,96,540,8,596,72,524,40,604,24,588
104,556,16,628,80,636,56,620,48,516,112,580,88,548,120,612
1,403,65,435,33,411,97,443,9,467,25,499,105,507,41,475
73,395,81,427,57,459,17,419,113,491,49,387,89,451,121,483
6,436,102,468,70,404,38,412,14,444,46,476,110,508,78,500
86,396,30,428,62,460,54,492,118,420,22,452,94,388,126,484
129,563,193,531,161,571,225,539,241,595,233,523,249,603,185,555
153,587,169,627,137,635,201,619,177,515,145,579,209,547,217,611
7,405,71,469,39,437,103,413,47,445,15,509,79,477,31,501
111,397,63,461,55,429,87,421,23,493,119,389,95,453,127,485
134,402,198,434,166,410,230,442,238,466,174,506,158,394,246,474
190,498,254,426,142,458,150,386,182,418,206,490,214,450,222,482
128,533,192,541,160,565,232,525,224,573,240,597,184,557,248,605
168,589,200,517,152,629,136,549,176,637,144,621,208,581,216,613
5,439,37,407,69,447,103,415,13,471,45,503,29,479,77,511
53,399,85,431,21,463,61,391,109,423,93,455,117,495,125,487
2,530,34,562,66,538,98,522,10,570,42,554,26,594,50,602
18,514,74,586,58,626,82,546,106,634,90,578,114,618,122,610
135,315,167,339,199,347,259,307,231,371,239,379,191,331,247,299
175,363,159,323,143,355,255,291,207,275,183,283,151,267,215,223
133,406,197,438,165,470,229,414,245,446,141,478,237,502,253,398
157,510,189,462,173,430,205,390,149,422,213,454,181,494,221,486
130,316,260,340,194,372,162,348,226,308,234,380,242,332,250,300
202,364,186,324,154,356,138,292,170,276,178,284,210,218,268,146
4,535,36,543,68,567,100,527,12,599,44,575,28,559,76,607
52,591,20,631,60,639,84,519,108,623,92,551,116,583,124,615
3,440,35,432,67,400,99,408,11,464,43,496,27,472,51,504
19,392,75,424,59,456,83,384,107,416,91,488,115,448,123,480
132,401,196,441,164,409,228,433,236,465,204,473,244,393,188,497
252,505,140,425,212,457,156,385,172,417,180,449,148,489,220,481
131,534,195,542,163,566,227,526,235,574,203,598,243,558,187,606
251,590,211,630,179,638,139,622,155,550,171,518,219,582,147,614
761,660,737,652,705,668,745,692,673,700,641,684,713,644,753,724
729,732,681,756,721,716,764,676,697,748,689,657,740,665,649,708
760,528,736,536,704,560,744,520,672,568,712,592,752,552,640,600
728,584,680,624,720,512,696,632,688,616,664,544,608,656,648,576
762,659,738,651,706,667,746,643,714,691,674,699,754,683,730,723
722,731,763,658,642,755,739,675,707,650,682,715,698,666,690,747
257,345,265,313,281,305,273,337,609,369,577,377,617,329,513,529
545,297,633,361,625,321,585,537,601,289,553,353,593,521,569,561
256,373,261,341,264,349,280,317,272,381,269,309,285,333,277,365
352,301,320,325,288,357,328,304,360,312,376,293,296,368,336,344
258,338,266,346,282,314,274,370,766,306,710,378,742,330,678,362
646,298,750,322,718,354,758,290,734,662,686,670,726,702,694,654
262,375,263,343,270,311,271,351,286,319,278,342,287,350,279,374
294,318,358,383,359,310,295,382,326,303,327,367,366,335,302,334
765,661,709,663,741,653,711,669,767,655,743,671,749,695,679,703
677,727,751,693,647,701,717,687,757,685,733,725,719,735,645,759
```


Copyright 2014 Cray Inc.

grid_order Utility

- Can use grid_order utility without first running the application with the Cray performance tools if you know a program's data movement pattern
- Originally designed for MPI programs, but since reordering is done by PMI, it can be used by other programming models (since PMI is used by MPI, SHMEM and PGAS programming models)
- Utility available if perftools modulefile is loaded
- See grid_order(1) man page or run grid_order with no arguments to see usage information

Copyright 2014 Cray Inc.

Reorder Example for Bisection Bandwidth

- Assume 32 ranks
- Decide on row or column ordering:
- $ grid_order –R –g 2,16

```
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15
16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31
```

- $ grid_order –C –g 2,16

```
0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30
1,3,5,7,9,11,13,15,17,19,21,23,25,27,29,31
```

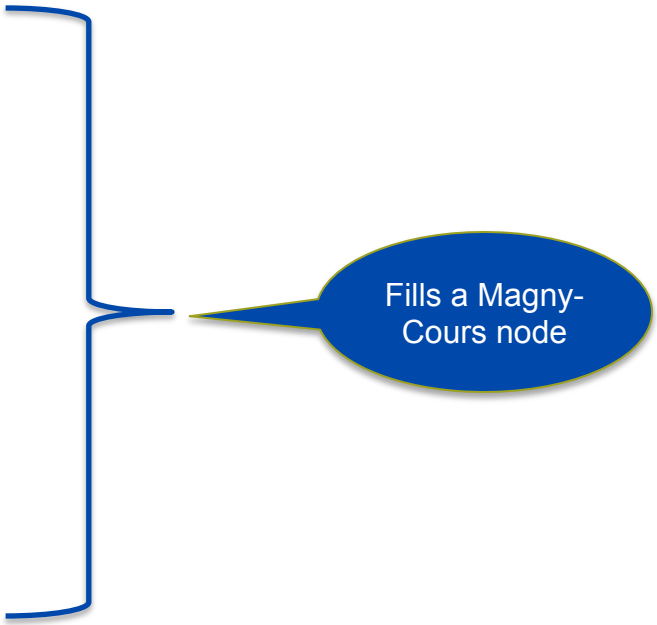
- Since rank 0 talks to rank 16, and not with rank 1, we choose Row ordering

Copyright 2014 Cray Inc.

Reorder Example for Bisection Bandwidth (2)

- Specify cell (or chunk) to make sure rank pairs live on same node (but don't care how many pairs live on a node)
- $ grid_order –R –g 2,16 –c 2,1

```
0,16
1,17
2,18
3,19
4,20
5,21
6,22
7,23
8,24
9,25
10,26
11,27
12,28
13,29
14,30
15,31
```



Copyright 2014 Cray Inc.

Using New Rank Order

- Save grid_order output to file called MPICH_RANK_ORDER
- Export MPICH_RANK_REORDER_METHOD=3
- Run non-instrumented binary with and without new rank order to check overall wallclock time for improvements

Copyright 2014 Cray Inc.

Collecting Loop Work Estimates

- Load PrgEnv-cray module (must use CCE)
- Load perftools module

- Compile **AND** link with –h profile_generate
  - cc –h profile_generate –o my_program my_program.c

- Instrument binary for tracing
  - pat_build –w my_program

- Run application

- Create report with loop statistics
  - pat_report my_program.xf > loops_report

pat_report produces report plus .ap2 file that can be used with Reveal

Copyright 2014 Cray Inc.

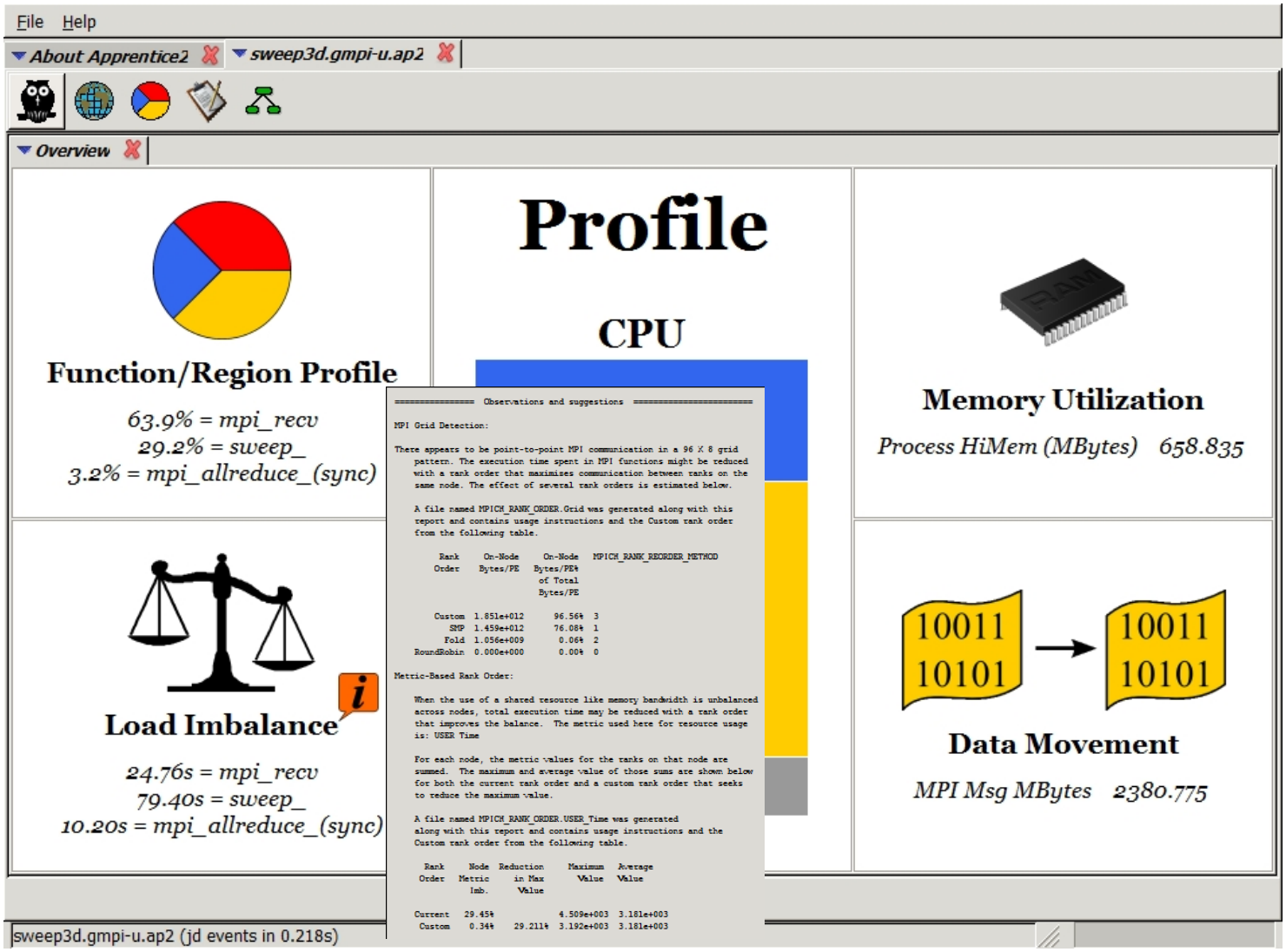
Example Report – Inclusive Loop Time

Table 2: Loop Stats by Function (from -hprofile_generate)

Loop Incl Time Total	Loop Hit	Loop Trips Avg	Loop Trips Min	Loop Trips Max	Function=/.LOOP[.] PE=HIDE
8.995914	100	25	0	25	sweepy_.LOOP.1.li.33
8.995604	2500	25	0	25	sweepy_.LOOP.2.li.34
8.894750	50	25	0	25	sweepz_.LOOP.05.li.49
8.894637	1250	25	0	25	sweepz_.LOOP.06.li.50
4.420629	50	25	0	25	sweepx2_.LOOP.1.li.29
4.420536	1250	25	0	25	sweepx2_.LOOP.2.li.30
4.387534	50	25	0	25	sweepx1_.LOOP.1.li.29
4.387457	1250	25	0	25	sweepx1_.LOOP.2.li.30
2.523214	187500	107	0	107	riemann_.LOOP.2.li.63
1.541299	20062500	12	0	12	riemann_.LOOP.3.li.64
0.863656	1687500	104	0	108	parabola_.LOOP.6.li.67

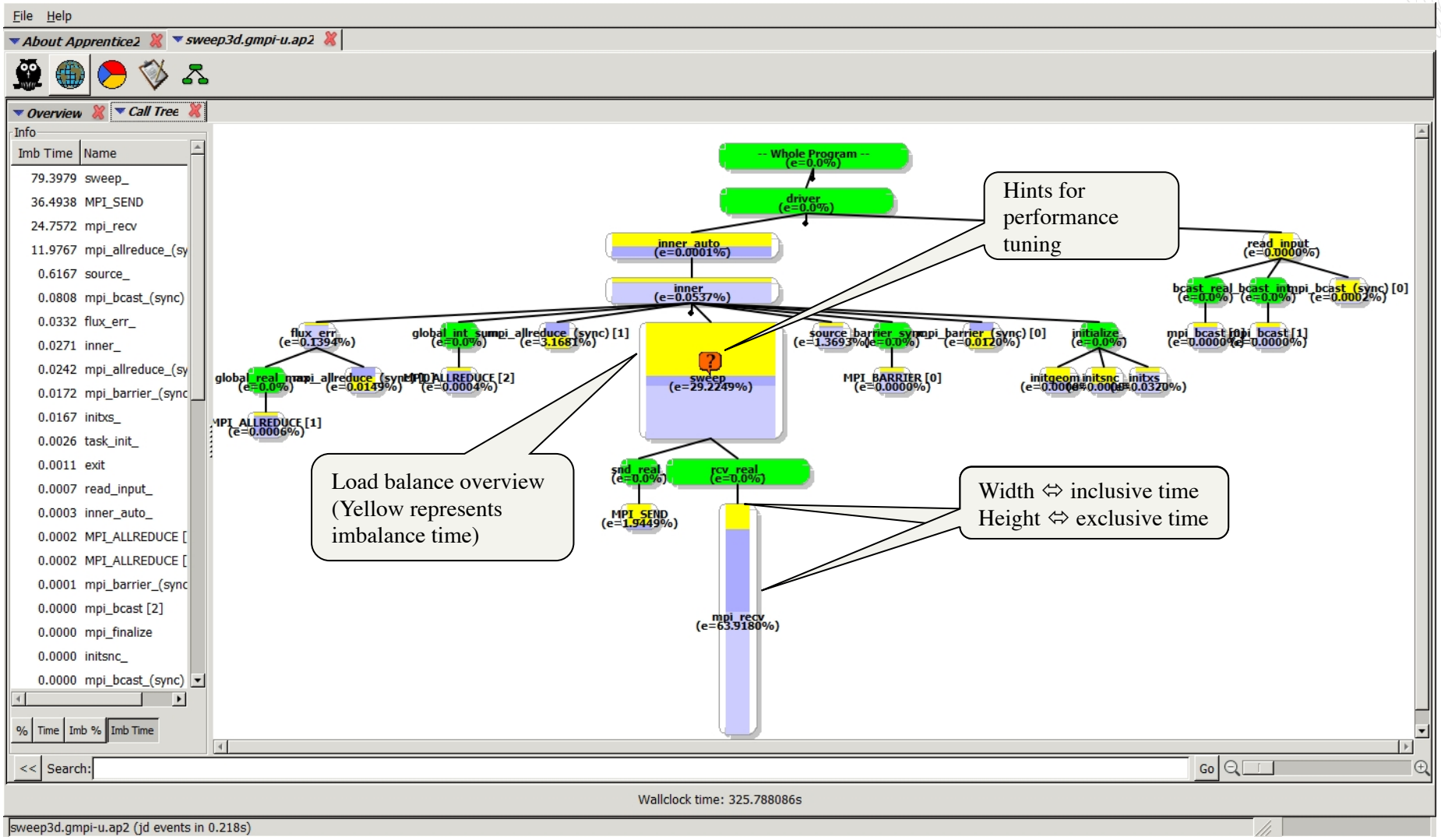
Copyright 2014 Cray Inc.

Cray Apprentice2 Overview



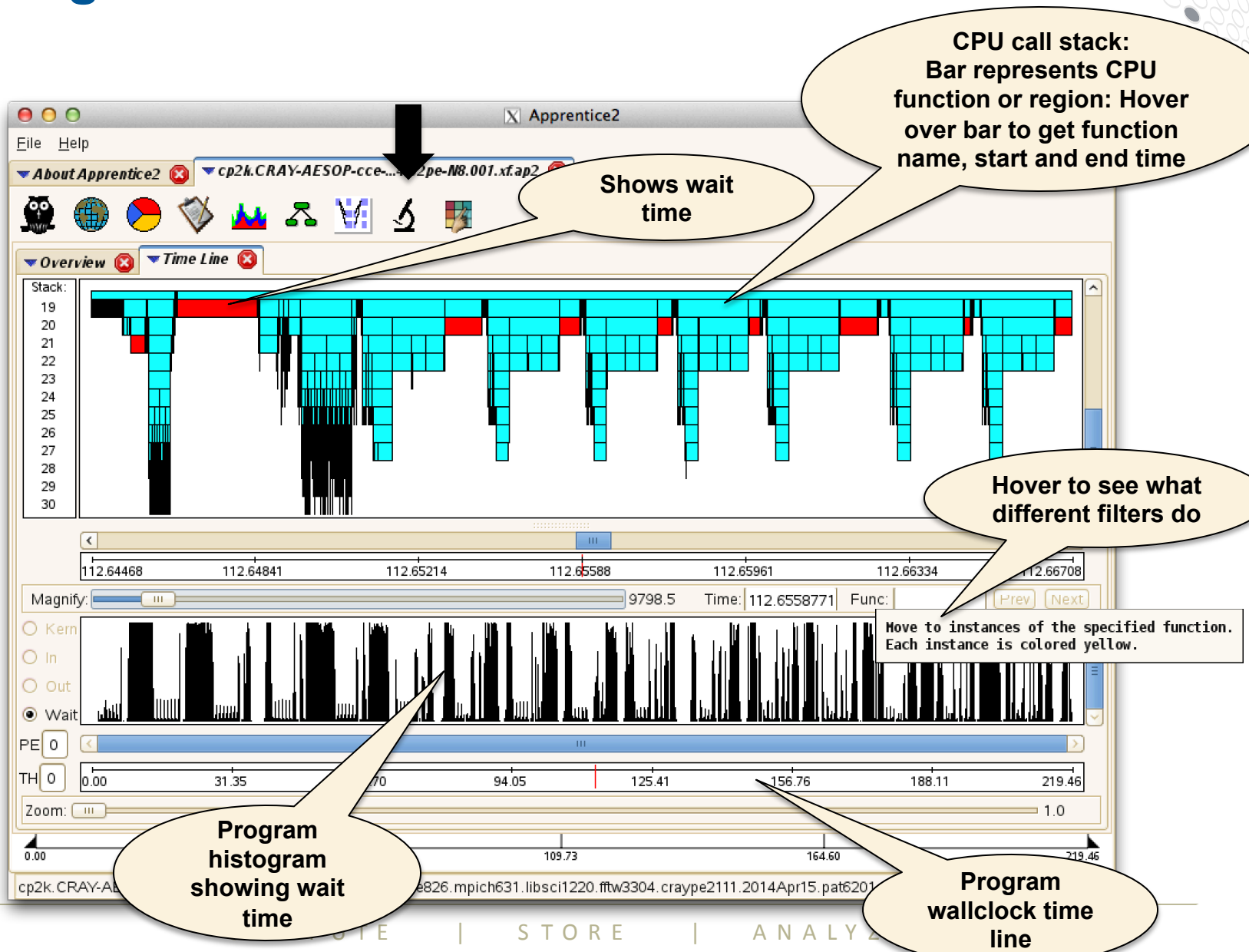
Copyright 2014 Cray Inc.

Call Tree View with Load Imbalance Information



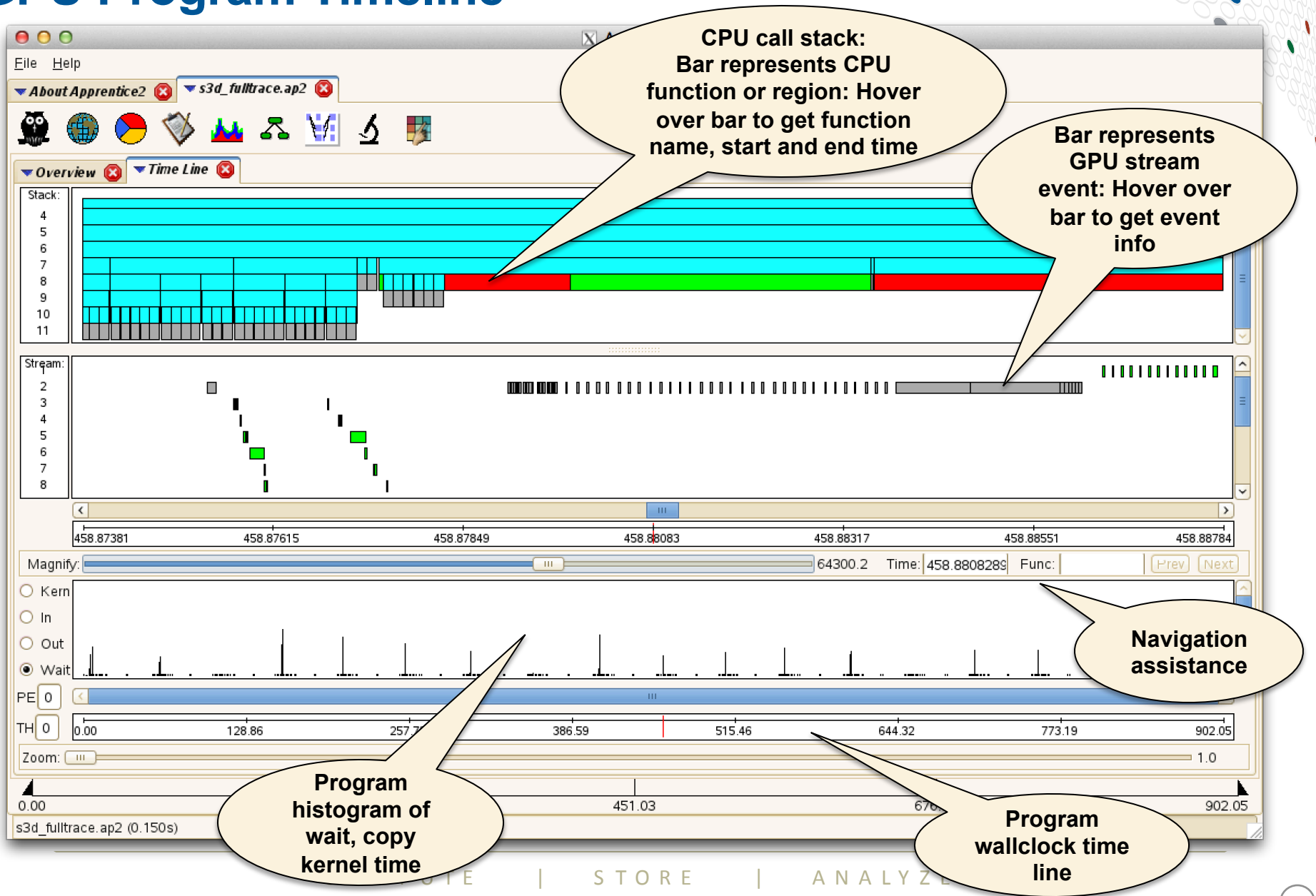
Copyright 2014 Cray Inc.

CPU Program Timeline: 36GB CP2K Full Trace



Copyright 2014 Cray Inc.

GPU Program Timeline



Copyright 2014 Cray Inc.

Example Default Accelerator Statistics

```
Table 1:  Time and Bytes Transferred for Accelerator Regions
  Host  |    Host  |     Acc  | Acc Copy  | Acc Copy  | Calls  |Calltree
 Time%  |    Time  |    Time  |       In  |      Out  |        | PE=HIDE
        |          |          | (MBytes)  | (MBytes)  |        |
 100.0% |   2.750  |   2.015  |  2812.760 |    13.568 |    103 |Total
|-------------------------------------------------------------------------------------------------------
| 100.0% |    2.750 |    2.015 |  2812.760 |    13.568 |    103 |lbm3d2p_d_
|        |          |          |           |           |        | lbm3d2p_d_.ACC_DATA_REGION@li.104
|||-----------------------------------------------------------------------------------------------------
3||  63.5% |    1.747 |    1.747 |  2799.192 |        -- |      1 |lbm3d2p_d_.ACC_COPY@li.104
3||  22.1% |    0.609 |    0.088 |    12.304 |    12.304 |     36 |streaming_
||||-----------------------------------------------------------------------------------------------------
4|||  20.6% |    0.566 |    0.046 |    12.304 |    12.304 |     27 |streaming_exchange_
5|||        |          |          |           |           |        | streaming_exchange_.ACC_DATA_REGION@li.526
6|||  18.8% |    0.517 |       -- |        -- |        -- |      1 |  streaming_exchange_.ACC_DATA_REGION@li.526(exclusive)
4|||   1.6% |    0.043 |    0.042 |        -- |        -- |      9 |streaming_.ACC_DATA_REGION@li.907
5|||   1.1% |    0.031 |    0.031 |        -- |        -- |      4 | streaming_.ACC_REGION@li.909
6|||   1.1% |    0.031 |       -- |        -- |        -- |      1 |  streaming_.ACC_REGION@li.909(exclusive)
||||=====================================================================================================

...
```


Copyright 2014 Cray Inc.

Lab Time

- **bw.ncsa.illinois.edu:~heidi/scratch/lab/**
  - CrayPat-lite
    - pr01
  - Loop statistics
    - pr05
  - MPI rank reorder
    - pr03
  - Apprentice2
    - pr04
  - Reveal
    - pr05, pr01
- **CCDB**

Copyright 2014 Cray Inc.

Questions
?

Copyright 2014 Cray Inc.

Legal Disclaimer

Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.

Cray Inc. may make changes to specifications and product descriptions at any time, without notice.

All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.

Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.

Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, URIKA, and YARCDATA. The following are trademarks of Cray Inc.: ACE, APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.

Copyright 2014 Cray Inc.