Performance Tools Documentation and Tips

Heidi Poxon
Technical Lead
Programming Environment
Cray Inc.

March 2016

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 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 names and brands may be claimed as the property of others. Other product and service names mentioned herein are the trademarks of their respective owners.

Cray PE Documentation Available

- Release Notes
  - `> module help` *`product/product_version`*

- User Guides
  - http://docs.cray.com

- Man pages, for example:
  - cc
  - crayftn
  - intro_directives
  - Intro_openacc

Perftools Documentation Available

- Release Notes
  - `> module help perftools/`*`version_number`*

- User manual “Using the Cray Performance Measurement and Analysis Tools” available at http://docs.cray.com

- pat_help – interactive help utility on the Cray Performance toolset

- Man pages

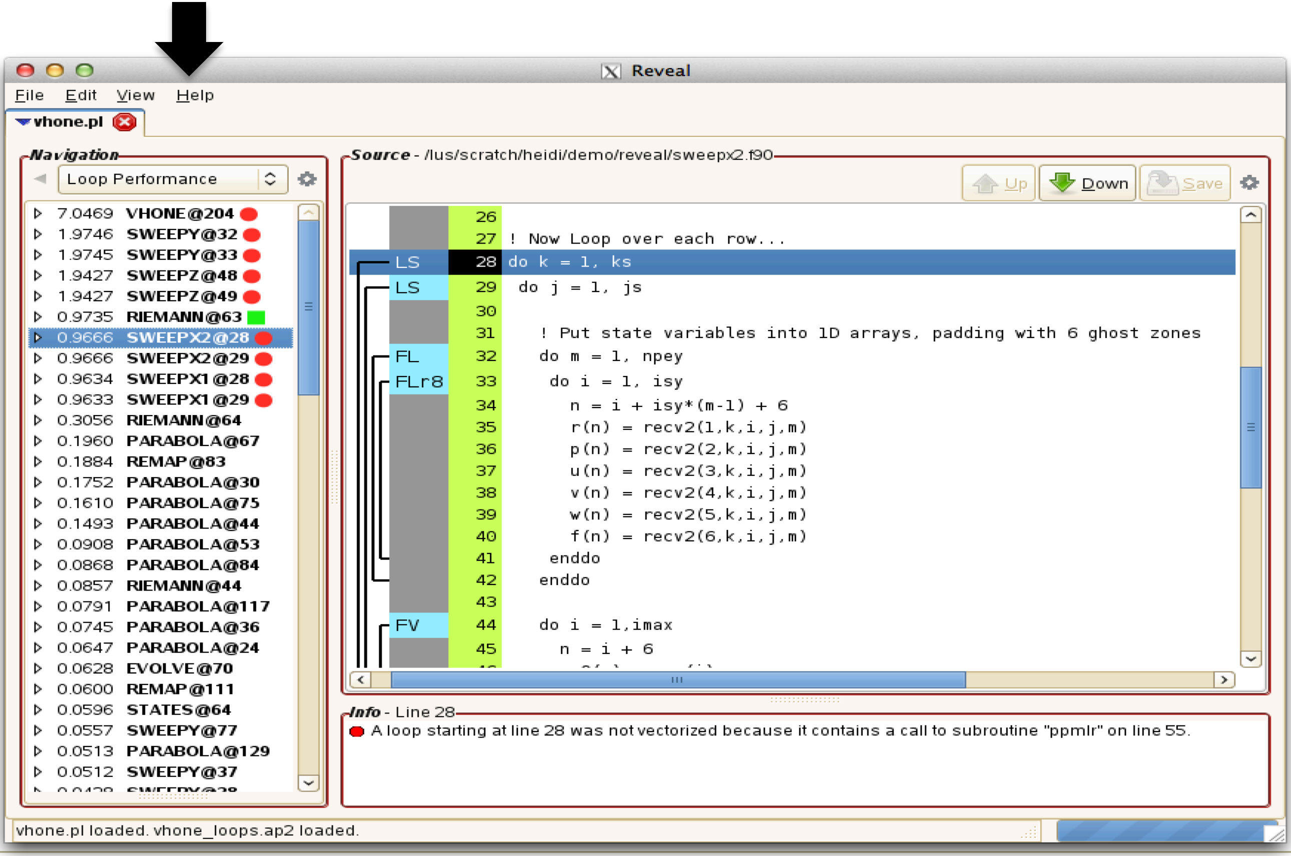
Man pages

- **intro_craypat(1)**
  - Introduces the craypat performance tool
  - Runtime environment variables (enable full trace, etc.)
- **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), etc.**
  - 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

Reveal Help



7.0469	VHONE@204	●
1.9746	SWEEPY@32	●
1.9745	SWEEPY@33	●
1.9427	SWEEPZ@48	●
1.9427	SWEEPZ@49	●
0.9735	RIEMANN@63	🟩
0.9666	SWEEPX2@28	●
0.9666	SWEEPX2@29	●
0.9634	SWEEPX1@28	●
0.9633	SWEEPX1@29	●
0.3056	RIEMANN@64	
0.1960	PARABOLA@67	
0.1884	REMAP@83	
0.1752	PARABOLA@30	
0.1610	PARABOLA@75	
0.1493	PARABOLA@44	
0.0908	PARABOLA@53	
0.0868	PARABOLA@84	
0.0857	RIEMANN@44	
0.0791	PARABOLA@117	
0.0745	PARABOLA@36	
0.0647	PARABOLA@24	
0.0628	EVOLVE@70	
0.0600	REMAP@111	
0.0596	STATES@64	
0.0557	SWEEPY@77	
0.0513	PARABOLA@129	
0.0512	SWEEPY@37	
0.0429	SWEEPY@38	

Source - /lus/scratch/heidi/demo/reveal/sweepx2.f90

```
      26
      27  ! Now Loop over each row...
LS    28  do k = 1, ks
LS    29   do j = 1, js
      30
      31     ! Put state variables into 1D arrays, padding with 6 ghost zones
FL    32     do m = 1, npey
FLr8  33      do i = 1, isy
      34        n = i + isy*(m-1) + 6
      35        r(n) = recv2(1,k,i,j,m)
      36        p(n) = recv2(2,k,i,j,m)
      37        u(n) = recv2(3,k,i,j,m)
      38        v(n) = recv2(4,k,i,j,m)
      39        w(n) = recv2(5,k,i,j,m)
      40        f(n) = recv2(6,k,i,j,m)
      41      enddo
      42     enddo
      43
FV    44     do i = 1,imax
      45       n = i + 6
```

Info - Line 28

● A loop starting at line 28 was not vectorized because it contains a call to subroutine "ppmlr" on line 55.

vhone.pl loaded. vhone_loops.ap2 loaded.

Reveal Usage Recipe

- Access Cray compiler
  - > module load PrgEnv-cray
- Access perftools
  - > module load perftools-base
- Enable loop work estimates program instrumentation
  - > module load perftools-lite-loops
- Build program (make)
- Run program to get loop work estimates in file with .ap2 suffix

Reveal Usage Recipe (2)

- **Disable loop work estimates program instrumentation so we can get fully optimized program now**
  - > module unload perftools-lite-loops

- **Create program library with CCE:**
  - Add –h pl=/*full_path*/my_program.pl to program's Makefile

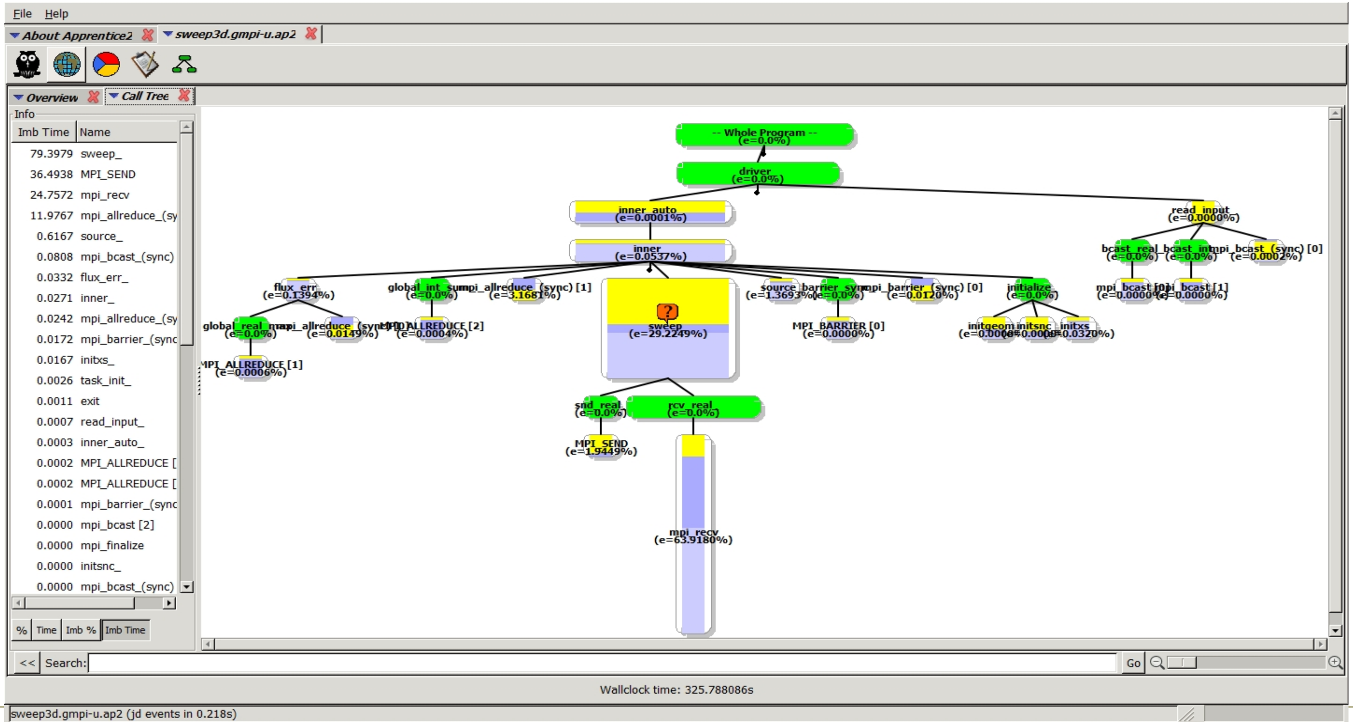
- **Rebuild application with full optimization**
  - > make clean
  - > make

- **Launch Reveal**
  - > reveal /*full_path*/my_program.pl loop_work_estimates.ap2

How to Install Apprentice2 on Your Laptop

- > module load perftools
- Go to:
  - $CRAYPAT_ROOT/share/desktop_installers/
- Download .dmg or .exe installer
- Double click on installer and follow directions to install

Apprentice2 Help



Imb Time	Name
79.3979	sweep_
36.4938	MPI_SEND
24.7572	mpi_recv
11.9767	mpi_allreduce_(sy
0.6167	source_
0.0808	mpi_bcast_(sync)
0.0332	flux_err_
0.0271	inner_
0.0242	mpi_allreduce_(sy
0.0172	mpi_barrier_(sync
0.0167	initxs_
0.0026	task_init_
0.0011	exit
0.0007	read_input_
0.0003	inner_auto_
0.0002	MPI_ALLREDUCE [
0.0002	MPI_ALLREDUCE [
0.0001	mpi_barrier_(sync
0.0000	mpi_bcast [2]
0.0000	mpi_finalize
0.0000	initsnc_
0.0000	mpi_bcast_(sync)

Wallclock time: 325.788086s

sweep3d.gmpi-u.ap2 (jd events in 0.218s)

Why Should I generate a “.ap2” file?

- The “.ap2” file is a self contained compressed performance file

- Normally it is about 5 times smaller than the “.xf” file

- Contains the information needed from the application binary
  - Can be reused, even if the application binary is no longer available or if it was rebuilt

- It is the only input format accepted by Cray Apprentice^2

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

More on pat_report Data

Data from pat_report

- Default reports are intended to be useful for most applications

- Don’t need to rerun program to get more detailed data

- Different aggregations, or levels of information available
  - Get fined-grained thread-imbalance information for OpenMP program

- Get list of tables available:
  - > pat_report –O –h

- Other formats available (txt, html, csv, xml)

A Useful Tip. . .

If you don’t see the function you are looking for in a report:

- Disable pruning: “pat_report –P . . .”
- Disable thresholding: “pat_report –T . . .”

Notes Section

- Check the Notes before each table in the text report

```
Notes for table 5:

  The Total value for Process HiMem (MBytes), Process Time is the avg
  for the PE values.

  The value shown for Process HiMem is calculated from information in
  the /proc/self/numa_maps files captured near the end of the program.
  It is the total size of all pages, including huge pages, that were
  actually mapped into physical memory from both private and shared
  memory segments.

  This table shows only the maximum, median, minimum PE entries,
    sorted by Process Time.
```

Questions About the Data?

- See Job summary information at top of report
- See Details section at bottom of report (may include warnings from CrayPat)
- Check pat_help
- Check man pages

Pat_help

- > pat_help environment . . .

```
pat_help environment (.=quit ,=back ^=up /=top ~=search)
=> PAT_RT_SAMPLING_DATA

    Specifies additional data to collect during a sampling
    experiment. The valid values are shown below.

    The value may be followed by '@ratio' which indicates the
    frequency at which the data is sampled. By default the data is
    sampled once for every 100 sampled program counter addresses. For
    example, if 'ratio' is '1', the additional data requested would
    be collected each time the program counter is sampled.
    If the 'ratio' is '1000', the additional data requested would
    be collected once every 1000 program counter samples.

    Collecting additional data during sampling is only supported in
    full-trace mode (see PAT_RT_SUMMARY).

  Additional topics that may follow "PAT_RT_SAMPLING_DATA":

    cray_pm      perfctr
    cray_rapl    rusage
    heap         sheap
    memory
```

Pat_help (2)

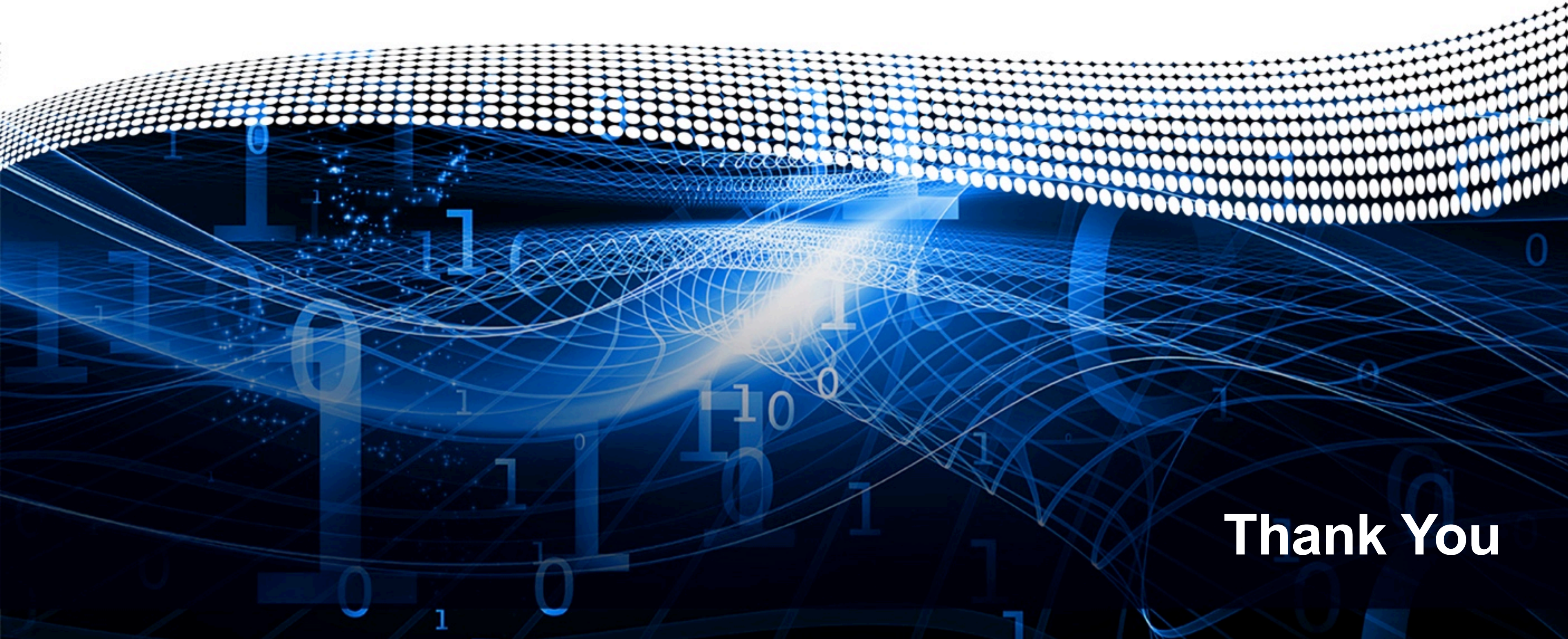
- > pat_help environment PAT_RT_SAMPLING_DATA memory

```
pat_help environment PAT_RT_SAMPLING_DATA
(.=quit ,=back ^=up /=top ~=search) => memory

    memory    collect data about the current state of memory

      himem          -  memory high water mark
      rss            -  resident set size
      peak           -  maximum virtual memory used
      priv           -  private resident memory
      shared         -  shared resident memory
      proportional -  proportional resident memory
```

COMPUTE | STORE | ANALYZE



Thank You