Enhanced Productivity Using the Cray Performance Analysis Toolset

Heidi Poxon
Performance Tools Technical Lead
Cray Inc.
heidi@cray.com

May 6, 2009

CRAY
THE SUPERCOMPUTER COMPANY

Outline

- Performance tuning challenges
- Approaches to address challenges
- Cray performance analysis tool design overview
- Recent activities that enhance productivity
- Next steps

Performance Tuning Challenges

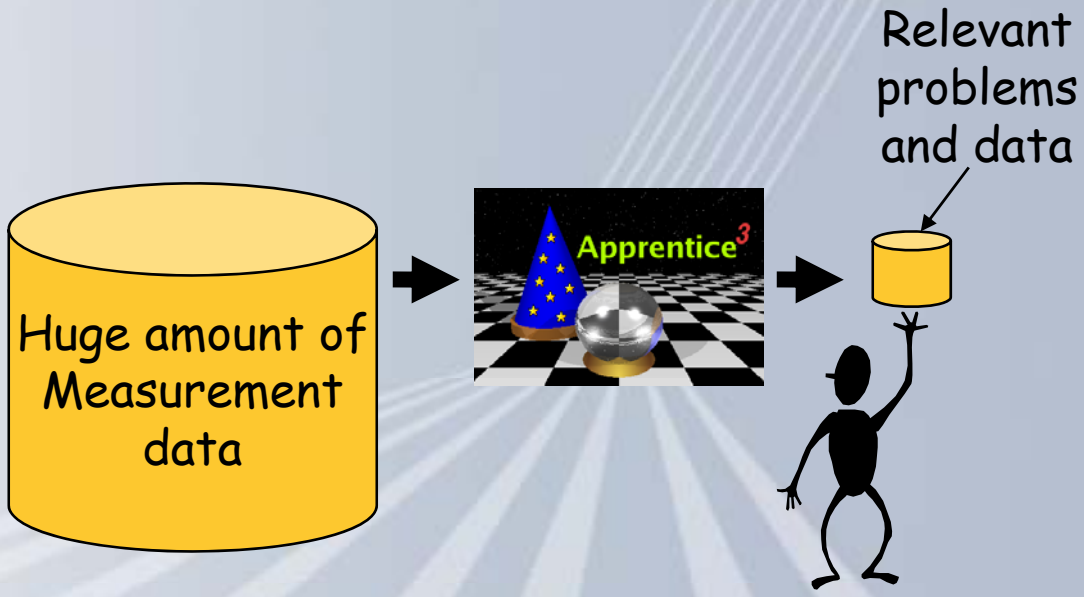
- Peta-scale systems complexity
  - Burden is on users to understand details of system software and architecture to correlate observations between application performance data and the system, to understand performance behavior of an application
- Scalability
  - For complex problems, traditional performance measurement techniques generate excessive amounts of information
- Presentation
  - Performance tools must be able to bridge the semantic gap between measured performance data and application-level abstraction

Approaches to Address Challenges

- Intelligently collect data
- Collapse, reduce, filter data
- Provide data storage format for efficient search engine
- Use performance models to automatically identify and expose performance anomalies
  - Load imbalance
  - Communication / synchronization / I/O problems
  - Inefficient execution patterns
  - Inefficient rank placement
  - Performance outliers (distinguish between “similar” and “different” application behavior)

The Next Generation of Cray Performance Tools

- Provide automatic performance analysis tools that use Cray optimization knowledge when analyzing data, in order to identify and expose performance anomalies



Cray Toolset Design Goals

- Assist the user with application performance analysis and optimization
  - Help user identify important and meaningful information from potentially massive data sets
  - Help user identify problem areas instead of just reporting data
  - Bring optimization knowledge to a wider set of users
- Focus on ease of use and intuitive user interfaces
  - Automatic program instrumentation
  - Automatic analysis
- Target scalability issues in all areas of tool development
  - Data management
    - Storage, movement, presentation

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
- CrayPat
  - pat_build: automatic instrumentation (no source code changes needed)
  - run-time library for measurements (transparent to the user)
  - pat_report for performance analysis reports
  - pat_help: online help utility
- Cray Apprentice^2
  - Graphical performance analysis and visualization tool
  - Can be used off-line on Linux system

Cray Performance Tools Recent Activities

- Automatic profiling analysis
  - Identifies top time consuming routines
  - Automatically creates instrumentation template customized to application
- Load imbalance analysis
  - Identifies computational code regions and synchronization calls that could benefit most from load balance optimization
  - Estimates savings if corresponding section of code were balanced
- Recommendation infrastructure
  - Directs users to meaningful performance data
  - Assists users who have little or no performance analysis experience

Automatic Profiling Analysis

- Analyze the performance data and direct the user to meaningful information

- Simplifies the procedure to instrument and collect performance data for novice users

- Based on a two phase mechanism
  1. Automatically detects the most time consuming functions in the application and feeds this information back to the tool for further (and focused) data collection

  2. Provides performance information on the most significant parts of the application

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-
          Oapa.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/homm
          e/standard.cray-xt.PE-2.1.56HD.pgi-8.0.amd64.pat-5.0.0.2-
          Oapa.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/amd6
          4/homme/pgi/pat-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.
```

Load Imbalance Analysis

Feedback provided to promote balanced use of requested computing resources

- MPI sync time
- MPI rank placement suggestions
- Imbalance metrics (both user and MPI functions)
- OpenMP imbalance analysis

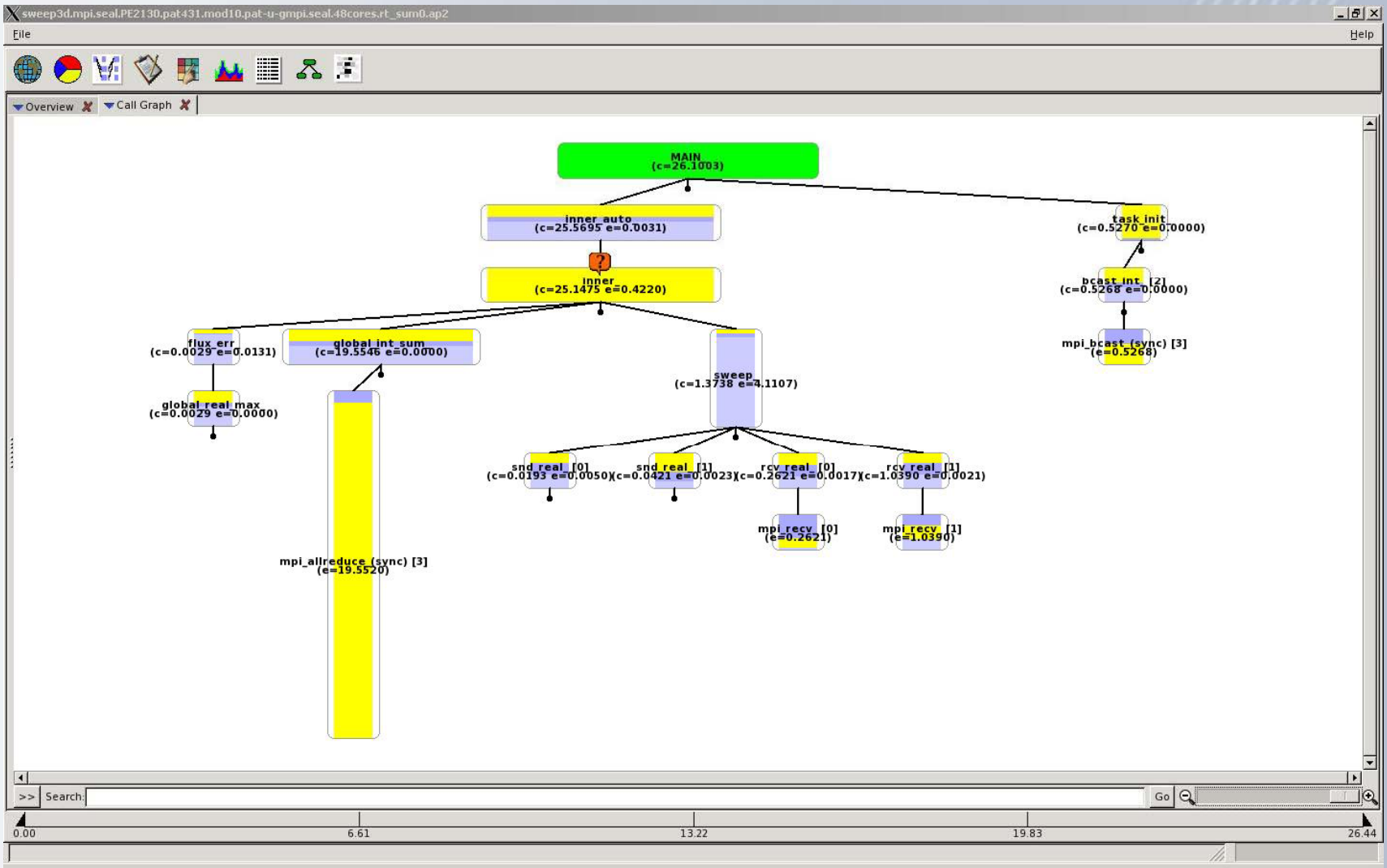
MPI Rank Placement Suggestions

- When to use?
  - Point-to-point communication consumes significant fraction of the program time and have a significant imbalance
  - When there seems to be a load imbalance of another type
    - Can get a suggested rank order file based on user time
    - Can have a different metric for load balance
- Prioritized placement information in resulting report
- Custom placement files automatically generated
- Notes section of pat_report describes how to use

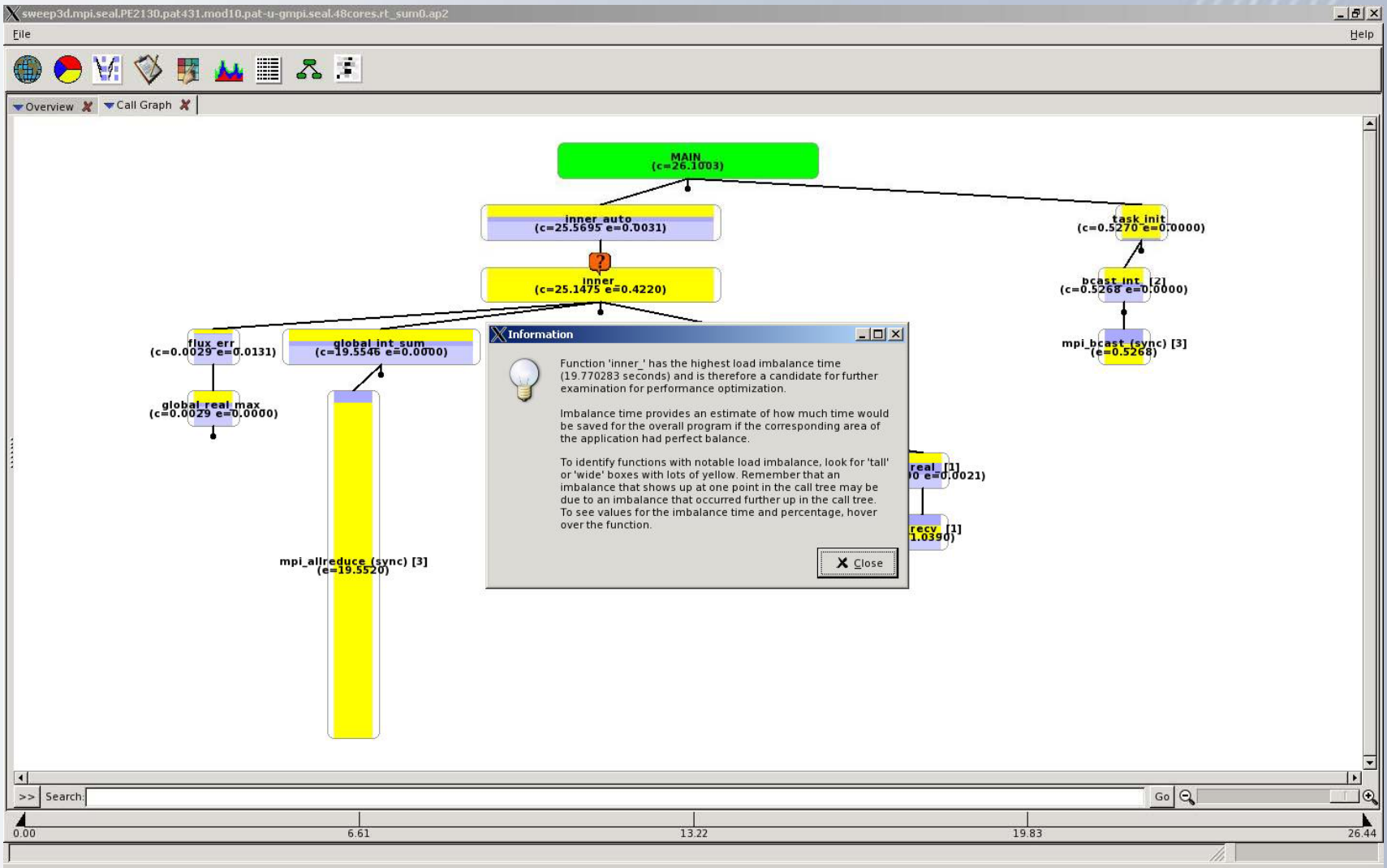
Example: -O mpi_rank_order (asura)

```
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
```

Call Tree Visualization (Swim3d)



Discrete Unit of Help



Cray Performance Tools Roadmap

CrayPat & Cray Apprentice^2 5.0 (July 2009)

- Hierarchical internal data format
- New architecture to support client / server model
- Usability enhancements

Benefits of New Architecture and Format

- Faster creation of processed data files
- Faster creation of default reports by pat_report
  - Example: 141 seconds reduced to 0.4 seconds for a ~30MB file
- Faster initial load of data into Cray Apprentice^2
  - Example: ~30 seconds reduced to < 1 second
  - Examples of 21% speedup for the full loading of data
- Summarization data is now pre-calculated
- New format allows efficient perusing of data for analysis

Usability Enhancements

- Frequently Asked Questions in pat_help
  - Brings answers to technical questions to all users
- Cray Apprentice^2 panel help
  - Explains key performance indicators for current display
  - Encourages “learn as you go” use of the tool

Next Steps

- Remote client support
  - Bulk of data resides on service node
  - Reduces tool response delays due to passing X11 data across ssh connection
- Consistent presentation of data between pat_report and Cray Apprentice^2
- More sophisticated data aggregation and association for wider range of analysis on a set of data
  - Detect outliers, perform hierarchical analysis, etc.
- Pat_report integrated into Cray Apprentice^2

Enhanced Productivity Using the Cray Performance Analysis Toolset

Questions / Comments

Thank You!

May 6, 2009

CRAY
THE SUPERCOMPUTER COMPANY