Cray Performance Tools

Colleen Bertoni

Outline

- Quick introduction to profiling terminology
- Overview of Cray Performance Tools
- Introduction to CrayPat-lite
  - Usage and output
  - Learning more without rerunning
- Experiments to try

Quick intro to profiling terminology

- Goal of profiling is to determine where a program is spending time and help tune code to the target system
- Two ways of collecting data:

Sampling

- At given intervals, samples the program counters to see what function the code is in
- Statistical, might miss some functions with small runtime
- Usually lower overhead than other methods

Tracing

- Logs information on enter and exit of function (traces function calls)
- More accurate about the performance, but more overhead since each function is instrumented

Overview of Cray Performance Tools

- CrayPat: Cray's **P**erformance **A**nalysis **T**ool
- Used to analyze program performance on Cray Supercomputers
- Both sampling and tracing modes
- Supports a wide range of programming models (including MPI, OpenMP, and CUDA)
- Can be used to identify hotspots, load imbalances, MPI rank communication information, I/O, and memory usage
- Supports Cray, Intel, and GCC compilers

Cray Performance Tools Components

- **CrayPat-lite and CrayPat**
  - Can be used to identify hotspots, load imbalances, MPI rank communication information, I/O, and memory usage, etc.
- **Cray Apprentice2**
  - Visualization of load imbalance, communication
- **Reveal**
  - Tool to show Cray compiler feedback about where to vectorization, blocking, and advice on modifying code to add OpenMP
  - module PrgEnv-cray should be used
- **PAPI**
  - Standard API for performance counters
  - Can set CrayPat environment variables to monitor and display hardware counters

CrayPat-lite

- Very easy-to-use version of CrayPat
- Simple interface
- Default is sampling-based profiling
- Automatically does data processing and generation of text report on the compute nodes at the end of a job

Using CrayPat-lite: Overview

1. Environment setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools-lite
```

2. Compiling the code to use CrayPat-lite

```
user@thetalogin6:~> make
```

3. Running the code

```
user@thetalogin6:~> qsub ./jobscript
```

4. Analyzing the output

```
Condensed report to stdout
```

Using CrayPat-lite: Overview

1. Environment setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools-lite
```

2. Compiling the code to use CrayPat-lite

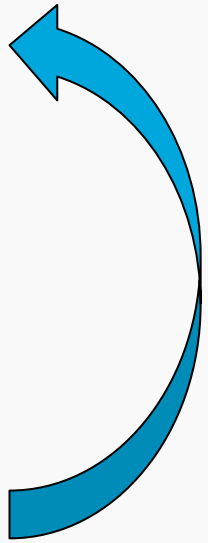
```
user@thetalogin6:~> make
```

3. Running the code

```
user@thetalogin6:~> qsub ./jobscript
```

4. Analyzing the output

```
Condensed report to stdout
```



More information with pat_report or Apprentice2 without re-running

Document results, make changes, repeat

Using CrayPat-lite

1. Environment Setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools-lite
```

Darshan needs to be unloaded since it has conflicts with CrayPat

"module load perftools-lite" loads performance instrumentation module, and will instrument programs when they are compiled.

perftools-base module should already be listed under "module list" If not, load perftools base with "module load perftools-base". This provides access to man pages and help system, and does not instrument code. (Should be loaded by default.)

Using CrayPat-lite

1. Environment Setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools-lite
```

```
user@thetalogin6:~> module list
Currently Loaded Modulefiles:
 1) modules/3.2.10.6                                 14) dvs/2.7_2.2.366.0.4.1_16
 2) eproxy/2.0.16-6.0.4.1_3.1__g001b199.ari          15) alps/6.4.2-6.0.4.1_3.1
 3) intel/18.0.0.128                                 16) rca/2.2.15-6.0.4.1_13.1_
 4) craype-network-aries                                 17) atp/2.1.1
 5) craype/2.5.14                                        18) perftools-base/7.0.1
 6) cray-libsci/18.04.1                                  19) PrgEnv-intel/6.0.4
 7) udreg/2.3.2-6.0.4.0_12.2__g2f9c3ee.ari               20) craype-mic-knl
 8) ugni/6.0.14-6.0.4.0_14.1__ge7db4a2.ari               21) cray-mpich/7.7.0
 9) pmi/5.0.13                                           22) nompirun/nompirun
10) dmapp/7.1.1-6.0.4.0_46.2__gb8abda2.ari               23) trackdeps
11) gni-headers/5.0.11-6.0.4.0_7.2__g7136988.ari         24) xalt
12) xpmem/2.2.2-6.0.4.1_18.2__g43b0535.ari               25) perftools-lite
13) job/2.2.2-6.0.4.0_8.2__g3c644b5.ari
```

Using CrayPat-lite

2. Compiling the code to use CrayPat-lite

(Any build script, not just make)

```
user@thetalogin6:~> make clean
user@thetalogin6:~> make
```

Rebuild code as usual

```
user@thetalogin6:~> make
ftn -O3 -qopt-report=5  -align array64byte -c test.f90 -o test.o
...
...
INFO: creating the CrayPat-instrumented executable
'test' (sample_profile) ...OK
```

Using CrayPat-lite

3. Running the code

```
user@thetalogin6:~> qsub -A proj -n 8 -t 30 ./jobscript.sh
Job routed to queue "default".
229865
```

Run code as usual

```
user@thetalogin6:~> cat jobscript.sh
#!/bin/bash -x
#COBALT --attrs mcdram=cache:numa=quad
#COBALT -t 30
#COBALT -n 8
#COBALT -A proj

echo "Starting Cobalt job script"
aprun -n 512 -N 64 test
```

Using CrayPat-lite

4. Analyzing the output

```
Condensed report to stdout (likely at the end of the
Cobalt .output file), and more information in *.ap2, *.xf
files and possibly a MPICH_RANK_ORDER file in a created
subdirectory
```

```
user@thetalogin6:~> ls –ltr
total 230912
-rwxr-xr-x 1 user users  54508616 May  7 03:06 test+orig
-rwxr-xr-x 1 user users  60700832 May  7 03:06 test
-rw-r--r-- 1 user users       237 May  7 03:29 230010.error
drwxr-x--- 5 user users       512 May  7 03:30 test+65086-1481s
-rw-r--r-- 1 user users      8619 May  7 03:31 230010.output
-rw-r--r-- 1 user cobalt     2021 May  7 03:32 230010.cobaltlog
```

Directory with additional files for analysis

Condensed text report in .output file

```
user@thetalogin6:~> cat 230010.output
 ... (normal program output)

########################################################################
#                                                                      #
#            CrayPat-lite Performance Statistics                       #
#                                                                      #
########################################################################

CrayPat/X:  Version 7.0.1 Revision a43a9d5  03/07/18 16:37:36
Experiment:                   lite  lite/sample_profile
Number of PEs (MPI ranks):     512
Numbers of PEs per Node:        64  PEs on each of  8  Nodes
Numbers of Threads per PE:       1
Number of Cores per Socket:     64
Execution start time:  Tue May  8 03:12:29 2018
System name and speed:  nid02670  1.301 GHz (nominal)
Intel Knights Landing CPU  Family:  6  Model: 87  Stepping:  1
MCDRAM: 7.2 GHz, 16 GiB available as quad, cache (100% cache)

Avg Process Time:           167.37 secs
High Memory:              21,104.6 MBytes     41.2 MBytes per PE
Observed CPU clock boost:     107.7 %
Instr per Cycle:               1.12
Observed CPU cycle rate:       1.39 GHz
I/O Read Rate:             2.730396 MBytes/sec
I/O Write Rate:            0.526600 MBytes/sec
```

```
user@thetalogin6:~> cat 230010.output
 ... (normal program output)

########################################################################
#                                                                      #
#            CrayPat-lite Performance Statistics                       #
#                                                                      #
########################################################################

CrayPat/X:  Version 7.0.1 Revision a43a9d5  03/07/18 16:37:36
Experiment:                   lite  lite/sample_profile
Number of PEs (MPI ranks):    512
Numbers of PEs per Node:       64  PEs on each of  8  Nodes
Numbers of Threads per PE:      1
Number of Cores per Socket:    64
Execution start time:  Tue May  8 03:12:29 2018
System name and speed:  nid02670  1.301 GHz (nominal)
Intel Knights Landing CPU  Family:  6  Model: 87  Stepping:  1
MCDRAM: 7.2 GHz, 16 GiB available as quad, cache (100% cache)

Avg Process Time:           167.37 secs
High Memory:              21,104.6 MBytes     41.2 MBytes per PE
Observed CPU clock boost:    107.7 %
Instr per Cycle:              1.12
Observed CPU cycle rate:      1.39 GHz
I/O Read Rate:            2.730396 MBytes/sec
I/O Write Rate:           0.526600 MBytes/sec
```

```
user@thetalogin6:~> cat 230010.output
 ... (normal program output)

########################################################################
#                                                                      #
#            CrayPat-lite Performance Statistics                       #
#                                                                      #
########################################################################

CrayPat/X:  Version 7.0.1 Revision a43a9d5  03/07/18 16:37:36
Experiment:                         lite  lite/sample_profile
Number of PEs (MPI ranks):           512
Numbers of PEs per Node:              64  PEs on each of   8  Nodes
Numbers of Threads per PE:             1
Number of Cores per Socket:           64
Execution start time:  Tue May  8 03:12:29 2018
System name and speed:  nid02670  1.301 GHz (nominal)
Intel Knights Landing CPU  Family:  6  Model: 87  Stepping:  1
MCDRAM: 7.2 GHz, 16 GiB available as quad, cache (100% cache)

Avg Process Time:                167.37 secs
High Memory:                  21,104.6 MBytes     41.2 MBytes per PE
Observed CPU clock boost:         107.7 %
Instr per Cycle:                   1.12
Observed CPU cycle rate:           1.39 GHz
I/O Read Rate:                2.730396 MBytes/sec
I/O Write Rate:               0.526600 MBytes/sec
```

Information about the job

Table 1: Profile by Function

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE
100.0%	16,655.5	--	--	Total
74.8%	12,453.9	--	--	USER
54.5%	9,078.6	972.4	9.7%	genral_
10.8%	1,800.9	187.1	9.4%	xyzint_
7.0%	1,167.2	100.8	8.0%	rt123_
1.0%	174.2	43.8	20.1%	build_abket_
13.1%	2,173.7	--	--	BLAS
4.9%	808.4	71.6	8.2%	gotoblas_dgemm_kernel_knl
4.7%	783.5	129.5	14.2%	gotoblas_dgetrf_single_knl
1.2%	206.8	47.2	18.6%	gotoblas_dlaswp_plus_knl
9.3%	1,546.3	--	--	MPI
9.1%	1,516.4	440.6	22.6%	MPI_ALLREDUCE
2.8%	474.2	--	--	ETC
2.7%	449.9	68.1	13.2%	__svml_exp8_mask_b3

```
Table 1:  Profile by Function
  Samp% |      Samp |  Imb.  |  Imb.  | Group
        |           |  Samp  | Samp%  |  Function=[MAX10]
        |           |        |        |   PE=HIDE
 100.0% | 16,655.5  |    --  |    --  | Total
|-----------------------------------------------------
|  74.8% | 12,453.9  |    --  |    --  | USER
||----------------------------------------------------
||  54.5% |  9,078.6  | 972.4  |  9.7%  | genral_
||  10.8% |  1,800.9  | 187.1  |  9.4%  | xyzint_
||   7.0% |  1,167.2  | 100.8  |  8.0%  | rt123_
||   1.0% |    174.2  |  43.8  | 20.1%  | build_abket_
||====================================================
|  13.1% |  2,173.7  |    --  |    --  | BLAS
||----------------------------------------------------
||   4.9% |    808.4  |  71.6  |  8.2%  | gotoblas_dgemm_kernel_knl
||   4.7% |    783.5  | 129.5  | 14.2%  | gotoblas_dgetrf_single_knl
||   1.2% |    206.8  |  47.2  | 18.6%  | gotoblas_dlaswp_plus_knl
||====================================================
|   9.3% |  1,546.3  |    --  |    --  | MPI
||----------------------------------------------------
||   9.1% |  1,516.4  | 440.6  | 22.6%  | MPI_ALLREDUCE
||====================================================
|   2.8% |    474.2  |    --  |    --  | ETC
||----------------------------------------------------
||   2.7% |    449.9  |  68.1  | 13.2%  | __svml_exp8_mask
|====================================================
```

User-defined functions

Math library functions

MPI functions

Not able to associate calls with a user function

Table 1: Profile by Function

Samp%	Samp	Imb. Samp	Imb. Samp%	Group Function=[MAX10] PE=HIDE
100.0%	16,655.5	--	--	Total
74.8%	12,453.9	--	--	USER
54.5%	9,078.6	972.4	9.7%	genral_
10.8%	1,800.9	187.1	9.4%	xyzint_
7.0%	1,167.2	100.8	8.0%	rt123_
1.0%	174.2	43.8	20.1%	build_abket_
13.1%	2,173.7	--	--	BLAS
4.9%	808.4	71.6	8.2%	gotoblas_dgemm_kernel_knl
4.7%	783.5	129.5	14.2%	gotoblas_dgetrf_single_knl
1.2%	206.8	47.2	18.6%	gotoblas_dlaswp_plus_knl
9.3%	1,546.3	--	--	MPI
9.1%	1,516.4	440.6	22.6%	MPI_ALLREDUCE
2.8%	474.2	--	--	ETC
2.7%	449.9	68.1	13.2%	__svml_exp8_mask_b3

By default, sampling experiment with 100 samples per second

```
Table 1:  Profile by Function
  Samp% |      Samp |  Imb.  |  Imb.  | Group
        |           |  Samp  | Samp%  |  Function=[MAX10]
        |           |        |        |   PE=HIDE
 100.0% | 16,655.5  |    --  |    --  | Total
|-----------------------------------------------
|  74.8% | 12,453.9 |     --  |    --  | USER
||----------------------------------------------
||  54.5% |  9,078.6 | 972.4  |  9.7%  | gen
||  10.8% |  1,800.9 | 187.1  |  9.4%  | xyz
||   7.0% |  1,167.2 | 100.8  |  8.0%  | rt1
||   1.0% |    174.2 |  43.8  | 20.1%  | bui
||==============================================
|  13.1% |  2,173.7 |     --  |    --  | BLAS
||----------------------------------------------
||   4.9% |    808.4 |  71.6  |  8.2%  | gotoblas_dgemm_kernel_knl
||   4.7% |    783.5 | 129.5  | 14.2%  | gotoblas_dgetrf_single_knl
||   1.2% |    206.8 |  47.2  | 18.6%  | gotoblas_dlaswp_plus_knl
||==============================================================
|   9.3% |  1,546.3 |     --  |    --  | MPI
||--------------------------------------------------------------
||   9.1% |  1,516.4 | 440.6  | 22.6%  | MPI_ALLREDUCE
||==============================================================
|   2.8% |    474.2 |     --  |    --  | ETC
||--------------------------------------------------------------
||   2.7% |    449.9 |  68.1  | 13.2%  | __svml_exp8_mask_b3
|==============================================================
```

Samp % is the percent of total samples taken which occurred in the given routine, averaged over all processes.

```
Table 1:  Profile by Function
  Samp% |      Samp |  Imb.  |  Imb.  | Group
        |           |  Samp  | Samp%  |  Function=[MAX10]
        |           |        |        |   PE=HIDE
 100.0% | 16,655.5  |    --  |    --  | Total
|-----------------------------------------
|  74.8% | 12,453.9  |    --  |    --  | U
||----------------------------------------
||  54.5% |  9,078.6  | 972.4  |  9.7%  |
||  10.8% |  1,800.9  | 187.1  |  9.4%  |
||   7.0% |  1,167.2  | 100.8  |  8.0%  |
||   1.0% |    174.2  |  43.8  | 20.1%  |
||========================================
|  13.1% |  2,173.7  |    --  |    --  | BLAS
||----------------------------------------------------------
||   4.9% |    808.4  |  71.6  |  8.2%  | gotoblas_dgemm_kernel_knl
||   4.7% |    783.5  | 129.5  | 14.2%  | gotoblas_dgetrf_single_knl
||   1.2% |    206.8  |  47.2  | 18.6%  | gotoblas_dlaswp_plus_knl
||==========================================================
|   9.3% |  1,546.3  |    --  |    --  | MPI
||----------------------------------------------------------
||   9.1% |  1,516.4  | 440.6  | 22.6%  | MPI_ALLREDUCE
||==========================================================
|   2.8% |    474.2  |    --  |    --  | ETC
||----------------------------------------------------------
||   2.7% |    449.9  |  68.1  | 13.2%  | __svml_exp8_mask_b3
|==========================================================
```

Samp is the number of samples which occurred in the given routine, averaged over all processes.

```
Table 1:  Profile by Function
  Samp% |     Samp |  Imb.  |  Imb.  | Group
        |          |  Samp  | Samp%  |  Function=[MAX10]
        |          |        |        |   PE=HIDE
 100.0% | 16,655.5 |    --  |    --  | Total
|-----------------------------------------------------------------------
|  74.8% | 12,453.9 |    --  |    --  |
||----------------------------------------------------------------------
||  54.5% |  9,078.6 | 972.4  |  9.7%  |
||  10.8% |  1,800.9 | 187.1  |  9.4%  |
||   7.0% |  1,167.2 | 100.8  |  8.0%  |
||   1.0% |    174.2 |  43.8  | 20.1%  |
||======================================================================
|  13.1% |  2,173.7 |    --  |    --  |
||----------------------------------------------------------------------
||   4.9% |    808.4 |  71.6  |  8.2%  | gotoblas_dgemm_kernel_knl
||   4.7% |    783.5 | 129.5  | 14.2%  | gotoblas_dgetrf_single_knl
||   1.2% |    206.8 |  47.2  | 18.6%  | gotoblas_dlaswp_plus_knl
||======================================================================
|   9.3% |  1,546.3 |    --  |    --  | MPI
||----------------------------------------------------------------------
||   9.1% |  1,516.4 | 440.6  | 22.6%  | MPI_ALLREDUCE
||======================================================================
|   2.8% |    474.2 |    --  |    --  | ETC
||----------------------------------------------------------------------
||   2.7% |    449.9 |  68.1  | 13.2%  | __svml_exp8_mask_b3
|======================================================================
```

Imb. Samp is ((maximum number of samples taken in given routine by one process) – (average number of samples taken in that routine)).

```
Table 1:  Profile by Function
  Samp% |      Samp |  Imb.  |  Imb.  | Group
        |           |  Samp  | Samp%  |  Function=[MAX10]
        |           |        |        |   PE=HIDE
 100.0% | 16,655.5  |    --  |    --  | Total
|-------------------------------------------------------------
|  74.8% | 12,453.9 |    --  |    --  | USER
||------------------------------------------------------------
||  54.5% |  9,078.6 | 972.4  |  9.7%  | genral_
||  10.8% |  1,800.9 | 187.1  |  9.4%  | xyzint_
||   7.0% |  1,167.2 | 100.8  |  8.0%  | rt123_
||   1.0% |    174.2 |  43.8  | 20.1%  | build_abket_
||============================================================
|  13.1% |
||--------
||   4.9%
||   4.7%
||   1.2%
||========
|   9.3% |  1,546.3 |    --  |    --  | MPI
||------------------------------------------------------------
||   9.1% |  1,516.4 | 440.6  | 22.6%  | MPI_ALLREDUCE
||============================================================
|   2.8% |    474.2 |    --  |    --  | ETC
||------------------------------------------------------------
||   2.7% |    449.9 |  68.1  | 13.2%  | __svml_exp8_mask_b3
|============================================================
```

Imb. Samp% = *(Imb. Samp)/(maximum samples by one process)*

(number of processes)/(number of processes − 1) * 100%

Table 2: Profile by Group, Function, and Line

```
  Samp% |      Samp |  Imb. |  Imb. | Group
        |           |  Samp | Samp% |  Function=[MAX10]
        |           |       |       |   Source
        |           |       |       |    Line
        |           |       |       |     PE=HIDE

 100.0% | 16,655.5 |    -- |    -- | Total
|-----------------------------------------------------------
|  74.8% | 12,453.9 |    -- |    -- | USER
||----------------------------------------------------------
||  54.5% |  9,078.6 |    -- |    -- | genral_
3|        |          |       |       |  user/test.f90
||||-------------------------------------------------------
4|||   1.0% |    158.4 |  38.6 | 19.6% | line.3725
4|||   2.6% |    435.9 |  74.1 | 14.6% | line.3729
4|||   1.4% |    232.8 |  43.2 | 15.7% | line.3731
4|||   1.6% |    263.8 |  64.2 | 19.6% | line.3748
4|||   2.3% |    388.5 |  62.5 | 13.9% | line.3818
4|||   2.0% |    325.1 |  50.9 | 13.6% | line.3829
4|||   3.9% |    654.0 | 168.0 | 20.5% | line.3840
4|||   1.3% |    214.5 |  44.5 | 17.2% | line.3856
4|||   2.1% |    351.2 |  68.8 | 16.4% | line.3862
4|||   3.8% |    627.0 | 113.0 | 15.3% | line.3867
4|||   3.5% |    583.2 |  68.8 | 10.6% | line.3868
4|||   1.2% |    200.5 |  42.5 | 17.5% | line.3870
```

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% 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    On-Node     On-Node   MPICH_RANK_REORDER_METHOD
        Order   Bytes/PE   Bytes/PE%
                            of Total
                            Bytes/PE


       Custom  4.050e+09      34.77%  3
          SMP  2.847e+09      24.45%  1
         Fold  1.025e+08       0.88%  2
    RoundRobin  6.098e+01       0.00%  0
```

MPI rank reordering

- Ideally, one would maximize on-node communication between MPI ranks and minimize inter-node communication
- “Observations” in output helps detect point-to-point MPI communication and suggests ways to reorder MPI ranks to reduce inter-node communication
- In addition to other files, a MPICH_RANK_ORDER is produced in the subdirectory
- If CrayPat-lite decides work is well balanced across the nodes, it will not be produced

```
user@thetalogin6:~> pat_help balance mpi_rank_order
user@thetalogin6:~> pat_help balance mpi_rank_order examples
```

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% 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    On-Node     On-Node   MPICH_RANK_REORDER_METHOD
        Order   Bytes/PE   Bytes/PE%
                            of Total
                            Bytes/PE


       Custom  4.050e+09      34.77%  3
          SMP  2.847e+09      24.45%  1
         Fold  1.025e+08       0.88%  2
    RoundRobin  6.098e+01       0.00%  0
```

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% 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.
```

In the subdirectory test+65086-1481s. Note that the instructions for using each MPICH_RANK_ORDER file are included within that file

Rank Order	On-Node Bytes/PE	On-Node Bytes/PE% of Total Bytes/PE	MPICH_
Custom	4.050e+09	34.77%	3
SMP	2.847e+09	24.45%	1
Fold	1.025e+08	0.88%	2
RoundRobin	6.098e+01	0.00%	0

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

MPI Grid Detection:

    There appears to be point-to-point MP                    X 60
    grid pattern. The 20.3% of the total                     MPI
    functions might be reduced with a ran
    communication between ranks on the sa
    several rank orders is estimated belo

    A file named MPICH_RANK_ORDER.Grid wa                    his
    report and contains usage instructions and the Custom rank order
    from the following table.


         Rank    On-Node     On-Node   MPICH_RANK_REORDER_METHOD
        Order   Bytes/PE   Bytes/PE%
                            of Total
                            Bytes/PE


       Custom  4.050e+09      34.77%  3
          SMP  2.847e+09      24.45%  1
         Fold  1.025e+08       0.88%  2
   RoundRobin  6.098e+01       0.00%  0
```

Percentage of total message bytes sent per PE stayed within each local compute node (the higher the percentage the better). In this case, the Custom order was a little better than the default SMP order.

Table 3: Memory Bandwidth by Numanode

Memory Traffic GBytes	DDR Memory Traffic GBytes	MCDRAM Memory Traffic GBytes	Thread Time	Memory Traffic GBytes / Sec	Numanode NodeId=[max,min] PE=HIDE
1,896	0.12	1,896	167.048792	11.35	Total
1,896	0.12	1,896	167.048792	11.35	numanode.0
1,907	0.07	1,907	167.048458	11.42	nid.2673
1,887	0.03	1,887	167.049105	11.29	nid.2679

Table 4: File Input Stats by Filename

Read Time	Read MBytes	Read Rate MBytes/sec	Reads	Bytes/ Call	File Name=!x/^/(proc|sys)/ PE=HIDE
0.128841	0.351787	2.730396	43,374.0	8.50	Total
0.106878	0.037503	0.350897	39,325.0	1.00	stdin
0.000202	0.000191	0.946528	25.0	8.00	_UnknownFile_

```
Table 5:  File Output Stats by Filename

    Write |    Write | Write Rate | Writes | Bytes/ | File Name[max15]
     Time |   MBytes | MBytes/sec |        |   Call |  PE=HIDE


 0.065824 | 0.034663 |   0.526600 |  783.0 |  46.42 | Total
|-------------------------------------------------------------------
| 0.061260 | 0.029872 |   0.487622 |  631.0 |  49.64 | /gpfs/mira-
home/user/output
| 0.004356 | 0.004333 |   0.994863 |  142.0 |  32.00 | _UnknownFile_
| 0.000208 | 0.000458 |   2.198979 |   10.0 |  48.00 | stdout
|===================================================================
Program invocation:  /home/user/test

For a complete report with expanded tables and notes, run:
  pat_report /gpfs/mira-home/user/test+20468-2670s

For help identifying callers of particular functions:
  pat_report -O callers+src /gpfs/mira-home/user/test+20468-2670s
To see the entire call tree:
  pat_report -O calltree+src /gpfs/mira-home/user/test+20468-2670s

For interactive, graphical performance analysis, run:
  app2 /gpfs/mira-home/user/test+20468-2670s

================  End of CrayPat-lite output  ======================
```

```
Table 5:  File Output Stats by Filename

    Write |    Write | Write Rate | Writes | Bytes/ | File Name[max15]
     Time |   MBytes | MBytes/sec |        |   Call |  PE=HIDE


 0.065824 | 0.034663 |   0.526600 |  783.0 |  46.42 | Total
|-----------------------------------------------------------------
| 0.061260 | 0.029872 |   0.487622 |  631.0 |  49.64 | /gpfs/mira-
home/user/output
| 0.004356 | 0.004333 |   0.994863 |  142.0 |
| 0.000208 | 0.000458 |   2.198979 |   10.0 |
|=================================================================
Program invocation:  /home/user/test

For a complete report with expanded tables and notes, run:
  pat_report /gpfs/mira-home/user/test+20468-2670s

For help identifying callers of particular functions:
  pat_report -O callers+src /gpfs/mira-home/user/test+20468-2670s
To see the entire call tree:
  pat_report -O calltree+src /gpfs/mira-home/user/test+20468-2670s

For interactive, graphical performance analysis, run:
  app2 /gpfs/mira-home/user/test+20468-2670s

===============  End of CrayPat-lite output  ======================
```

More information without rerunning

```
Table 5:  File Output Stats by Filename

    Write |    Write | Write Rate | Writes | Bytes/ | File Name[max15]
     Time |   MBytes | MBytes/sec |        |   Call |  PE=HIDE


 0.065824 | 0.034663 |   0.526600 |  783.0 |  46.42 | Total
|-----------------------------------------------------------------
| 0.061260 | 0.029872 |   0.487622 |  631.0 |  49.64 | /gpfs/mira-
home/user/output
| 0.004356 | 0.004333 |   0.994863 |  142.0 |
| 0.000208 | 0.000458 |   2.198979 |   10.0 |
|=================================================================
Program invocation:  /home/user/test

For a complete report with expanded tables and notes, run:
  pat_report /gpfs/mira-home/user/test+20468-2670s

For help identifying callers of particular functions:
  pat_report -O callers+src /gpfs/mira-home/user/test+20468-2670s
To see the entire call tree:
  pat_report -O calltree+src /gpfs/mira-home/user/test+20468-2670s

For interactive, graphical performance analysis, run:
  app2 /gpfs/mira-home/user/test+20468-2670s

================  End of CrayPat-lite output  ======================
```

More information without rerunning

Using CrayPat-lite: More information without re-running

```
user@thetalogin6:~> pat_report /gpfs/mira-home/user/test+20468-2670s
```

More details than the default report, including notes for each table

```
Notes for table 1:
  Table option:
    -O samp_profile
  Options implied by table option:
    -d sa%@0.95,sa,imb_sa,imb_sa% -b gr,fu,pe=HIDE

  Options for related tables:
    -O samp_profile+src        -O profile_max

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

Table 1:  Profile by Function

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

 100.0% | 16,655.5 |    -- |    -- | Total
|-----------------------------------------------------------
|  74.8% | 12,453.9 |    -- |    -- | USER
||----------------------------------------------------------
||  54.5% |  9,078.6 | 972.4 |  9.7% | genral_
```

```
Notes for table 1:
  Table option:
    -O samp_profile
  Options implied by table option:
    -d sa%@0.95,sa,imb_sa,imb_sa% -b gr,fu,pe=HIDE

  Options for related tables:
    -O samp_profile+src        -O profile_max

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

Table 1:  Profile by Function

  Samp% |     Samp |  Imb. |
        |          |  Samp | S
|       |          |       |          PE=HIDE

 100.0% | 16,655.5 |    -- |    -- | Total
|-----------------------------------------------------------
|  74.8% | 12,453.9 |    -- |    -- | USER
||----------------------------------------------------------
||  54.5% |  9,078.6 | 972.4 |  9.7% | genral_
```

Explanations of details of the tables can be found by running “man pat_report”

Using CrayPat-lite: More information without re-running

```
user@thetalogin6:~> pat_report /gpfs/mira-home/user/test+20468-2670s
```

- Includes a load imbalance table organized by maximum samplings
- Shows the maximum and minimum samplings in a given function

Table 2: Profile of maximum function times (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Function PE=[max,min]
100.0%	10,051.0	972.4	9.7%	genral_
100.0%	10,051.0	--	--	pe.192
87.0%	8,747.0	--	--	pe.126
19.8%	1,988.0	187.1	9.4%	xyzint_
19.8%	1,988.0	--	--	pe.192
16.5%	1,663.0	--	--	pe.322
19.5%	1,957.0	440.6	22.6%	MPI_ALLREDUCE
19.5%	1,957.0	--	--	pe.2
0.5%	52.0	--	--	pe.192
12.6%	1,268.0	100.8	8.0%	rt123_
12.6%	1,268.0	--	--	pe.117
10.7%	1,074.0	--	--	pe.462
9.1%	913.0	129.5	14.2%	gotoblas_dgetrf_single_knl

Table 2: Profile of maximum function times (limited entries shown)

Samp%	Samp	Imb. Samp	Imb. Samp%	Function PE=[max,min]
100.0%	10,051.0	972.4	9.7%	genral_
100.0%	10,051.0	--	--	pe.192
87.0%	8,747.0	--	--	pe.126
19.8%	1,988.0	187.1	9.4%	xyzint_
19.8%	1,988.0	--	--	pe.192
16.5%	1,663.0	--	--	pe.322
19.5%	1,957.0	440.6	22.6%	MPI_ALLREDUCE
19.5%	1,957.0	--	--	pe.2
0.5%	52.0	--	--	pe.192
12.6%	1,268.0	100.8	8.0%	rt123_
12.6%	1,268.0	--	--	pe.117
10.7%	1,074.0	--	--	pe.462
9.1%	913.0	129.5	14.2%	gotoblas_dgetrf_single_knl

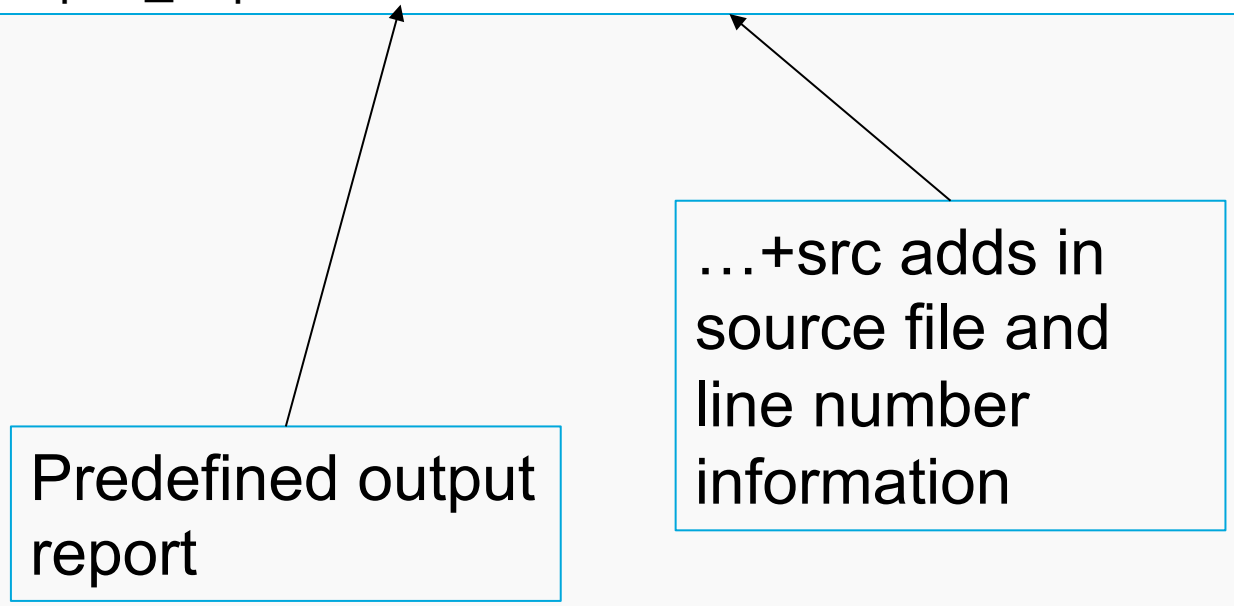
Samp is the maximum number of samplings in a given function by a PE

```
Table 5:  File Output Stats by Filename

    Write |    Write | Write Rate | Writes | Bytes/ | File Name[max15]
     Time |   MBytes | MBytes/sec |        |   Call |  PE=HIDE


 0.065824 | 0.034663 |   0.526600 |  783.0 |  46.42 | Total
|-----------------------------------------------------------------
| 0.061260 | 0.029872 |   0.487622 |  631.0 |  49.64 | /gpfs/mira-
home/user/output
| 0.004356 | 0.004333 |   0.994863 |  142.0 |
| 0.000208 | 0.000458 |   2.198979 |   10.0 |
|=================================================================
Program invocation:  /home/user/test

For a complete report with expanded tables and notes, run:
  pat_report /gpfs/mira-home/user/test+20468-2670s

For help identifying callers of particular functions:
  pat_report -O callers+src /gpfs/mira-home/user/test+20468-2670s
To see the entire call tree:
  pat_report -O calltree+src /gpfs/mira-home/user/test+20468-2670s

For interactive, graphical performance analysis, run:
  app2 /gpfs/mira-home/user/test+20468-2670s

==============  End of CrayPat-lite output  ======================
```

More information without rerunning

Using CrayPat-lite: More information without re-running

```
user@thetalogin6:~> pat_report -O callers+src test+20468-2670s
```

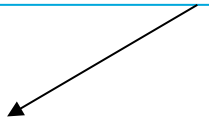


Using CrayPat-lite: More information without re-running

```
user@thetalogin6:~> pat_report -O callers+src test+20468-2670s
```

```
Table 1:  Profile by Function and Callers, with Line Numbers

  Samp% |      Samp | Group
        |           |  Function
        |           |   Caller
        |           |    PE=HIDE
 100.0% | 16,655.5 | Total
|-----------------------------------------------
|  74.8% | 12,453.9 | USER
||----------------------------------------------
||  54.5% |  9,078.6 | genral_
3|  54.5% |  9,077.2 |  int2e_:test.f90:line.3487
||||-------------------------------------------
4|||  36.8% |  6,121.2 | energy_:test.f90:line.1482
5|||        |          |  MAIN__:test.f90:line.654
6|||        |          |   main
4|||  17.7% |  2,948.0 | energy_:test.f90:line.1490
5|||        |          |  MAIN__:test.f90:line.654
6|||        |          |   main
```

Functions which call top functions



Using CrayPat-lite: More information without re-running

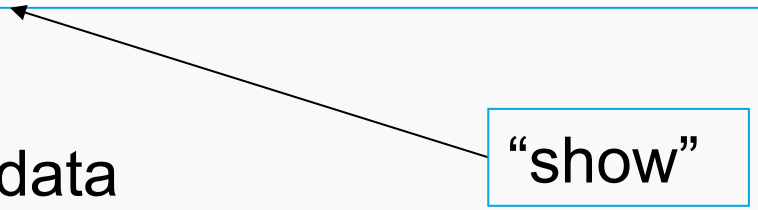
```
user@thetalogin6:~>pat_report -O calltree+src test+20468-2670s
```

```
Table 1:  Calltree View with Callsite Line Numbers

  Samp% |     Samp | Calltree
        |          |  PE=HIDE

 100.0% | 16,655.5 | Total
|-----------------------------------------------------------
|  87.0% | 14,482.7 | main
||----------------------------------------------------------
||  77.9% | 12,979.8 | MAIN__:test.f90:line.654
|||---------------------------------------------------------
3||  51.4% |  8,561.7 | energy_:test.f90:line.1482
4||  50.9% |  8,470.5 |  int2e_:test.f90:line.3487
|||||-------------------------------------------------------
5||||  10.1% |  1,685.4 | genral_:test.f90:line.3840
5||||   6.4% |  1,064.5 | genral_:test.f90:line.3818
```

Functions called by top functions

Using CrayPat-lite: More information without re-running

```
user@thetalogin6:~> pat_report -s pe=ALL test+20468-2670s
```



- Seeing per-rank or per-thread data
- Fine-grained imbalance information

```
Table 1:  Profile by Function

  Samp% |      Samp |  Imb.  |  Imb.  | Group
        |           |  Samp  | Samp%  |  Function
        |           |        |        |   PE

 100.0% | 16,655.5 |    -- |    -- | Total
|-----------------------------------------------------------
|  74.8% | 12,453.9 |    -- |    -- | USER
||----------------------------------------------------------
||  54.5% |  9,078.6 | 972.4 |  9.7% | genral_
|||---------------------------------------------------------
3||  60.3% | 10,051.0 |    -- |    -- | pe.192
3||  59.1% |  9,839.0 |    -- |    -- | pe.320
3||  58.5% |  9,746.0 |    -- |    -- | pe.384
3||  57.9% |  9,643.0 |    -- |    -- | pe.256
3||  57.7% |  9,615.0 |    -- |    -- | pe.0
3||  57.2% |  9,532.0 |    -- |    -- | pe.448
3||  57.0% |  9,491.0 |    -- |    -- | pe.121
3||  56.6% |  9,435.0 |    -- |    -- | pe.10
3||  56.5% |  9,415.0 |    -- |    -- | pe.186
3||  56.4% |  9,396.0 |    -- |    -- | pe.144
3||  56.4% |  9,390.0 |    -- |    -- | pe.288
3||  56.3% |  9,370.0 |    -- |    -- | pe.64
3||  56.2% |  9,367.0 |    -- |    -- | pe.128
```

Samplings in every MPI rank

Using CrayPat-lite: More information without re-running

- Missing a function?
  1. Samples in function were attributed to a caller function (associating lower level library routines with callers)
     - Disable this adjustment

```
user@thetalogin6:~> pat_report –P test+20468–2670s
```

  2. Function was below sampling threshold (0.95%)
     - Turn off thresholding:

```
user@thetalogin6:~> pat_report –T test+20468–2670s
```

Using CrayPat-lite: Overview

1. Environment setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools-lite
```

2. Compiling the code to use CrayPat-lite

```
user@thetalogin6:~> make
```

3. Running the code

```
user@thetalogin6:~> qsub ./jobscript
```

4. Analyzing the output

```
Condensed report to stdout
```

Helpful experiments to try

```
module avail perftools
```

- Identify time-consuming areas
  - perftools-lite
- Identify time-consuming loops (needs Cray compiler)
  - perftools-lite-loops

```
man perftools-lite
```

- Identify MPI communication issues
  - perftools-lite
  - perftools (pat_build –g mpi) to collect more MPI-specific information (analyzing size of MPI messages)
- Analyze hardware counters
  - perftools
  - papi_avail on compute nodes to see available counters
  - Set environment variable PAT_RT_PERFCTR

```
man hwpc
```

Using CrayPat: MPI communication

1. Environment setup

```
user@thetalogin6:~> module unload darshan
user@thetalogin6:~> module load perftools
```

2. Compiling the code to use CrayPat

```
user@thetalogin6:~> make
user@thetalogin6:~> pat_build –g mpi program
```

3. Running the code (inside a submission script)

```
user@thetalogin6:~> qsub –A proj –n ..../jobscript.sh
user@thetalogin6:~> cat jobscript.sh
#!/bin/bash –x
aprun –n 512 –N 64 program+pat
```

4. Analyzing the output

```
pat_report program+pat+41757–3827t/
```

Using CrayPat: MPI communication

1. Environment setup

```
user@thetalogin6:~> module unload
user@thetalogin6:~> module load p
```

Produces an instrumented copy of the original executable, program+pat

2. Compiling the code to use CrayPat

```
user@thetalogin6:~> make
user@thetalogin6:~> pat_build -g mpi program
```

3. Running the code (inside a submission script)

```
user@thetalogin6:~> qsub -A proj -n ..../jobscript.sh
user@thetalogin6:~> cat jobscript.sh
#!/bin/bash -x
aprun -n 512 -N 64 program+pat
```

4. Analyzing the output

https://pubs.cray.com

```
pat_report program+pat+41757-3827t/
```

References

- User manual “Using the Cray Performance Measurement and Analysis Tools” available at http://pubs.cray.com
- pat_help (after module perftools-base is loaded)

```
user@thetalogin6:~> pat_help
```

- Man pages

```
# basic usage for craypat-lite
user@thetalogin6:~> man perftools-lite
# output report information
user@thetalogin6:~> man pat_report
# basic usage and environment variables info
user@thetalogin6:~> man intro_craypat
```

Summary

- CrayPat-lite
  - Easy-to-use, simple interface
  - Lets you run on many nodes, look at performance when running at scale
- CrayPat
  - More control over functions traced, more MPI communication output
  - Lets you run on many nodes, look at performance when running at scale
- Try it out!

Acknowledgements

- Previous CrayPat tutorials
- Scott Parker, Ray Loy

Using CrayPat-lite: OpenMP

- OpenMP information (overhead from entering and leaving OpenMP regions, per-thread timings, thread load imbalances)
  - Collected by default
  - Most detail from using the Cray compiler

function.REGION@li.49
function.LOOP@li.53

```
user@thetalogin6:~> module swap PrgEnv-intel PrgEnv-cray
```

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% of the total execution time spent in MPI
    functions might be reduced with a rank order that maximizes
    communication between ranks on th
    several rank orders is estimated

    A file named MPICH_RANK_ORDER.Gri                              is
    report and contains usage instruc                              er
    from the following table.
```

This is the grid that pat_report identified by studying MPI message traffic. It can be changed by the user via the -s rank_grid_dim option.

Rank Order	On-Node Bytes/PE	On-Node Bytes/PE% of Total Bytes/PE	MPICH_RANK_REORDER_METHOD
Custom	4.050e+09	34.77%	3
SMP	2.847e+09	24.45%	1
Fold	1.025e+08	0.88%	2
RoundRobin	6.098e+01	0.00%	0

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% 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
    different mappings 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.

                                    Node     On-Node   MPICH_RANK_REORDER_METHOD
          Order   Bytes/PE   Bytes/PE%
                              of Total
                              Bytes/PE


         Custom  4.050e+09      34.77%  3
            SMP  2.847e+09      24.45%  1
           Fold  1.025e+08       0.88%  2
     RoundRobin  6.098e+01       0.00%  0
```

This MPI-based rank order is calculated only if this application shows that significant (>10%) time is spent doing MPI-related work.

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

MPI Grid Detection:

    There appears to be point-to-point MPI communication in a 35 X 60
    grid pattern. The 20.3% 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.
```

In the subdirectory test+65086-1481s. Note that the instructions for using each MPICH_RANK_ORDER file are included within that file

Rank Order	On-Node Bytes/PE	On-Node Bytes/PE% of Total Bytes/PE	MPICH_
Custom	4.050e+09	34.77%	3
SMP	2.847e+09	24.45%	1
Fold	1.025e+08	0.88%	2
RoundRobin	6.098e+01	0.00%	0

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

MPI Grid Detection:

    There appears to be point-to-point MP                    X 60
    grid pattern. The 20.3% of the total                     MPI
    functions might be reduced with a ran
    communication between ranks on the sa
    several rank orders is estimated belo

    A file named MPICH_RANK_ORDER.Grid wa                    his
    report and contains usage instructio                     der
    from the following table.

         Rank     On-Node      On-Node    MPICH_RANK_REORDER_METHOD
        Order    Bytes/PE    Bytes/PE%
                             of Total
                             Bytes/PE

       Custom  4.050e+09       34.77%  3
          SMP  2.847e+09       24.45%  1
         Fold  1.025e+08        0.88%  2
   RoundRobin  6.098e+01        0.00%  0
```

Custom rank order was able to arrange the ranks such that 34% of the total MPI message bytes sent per PE stayed within each local compute node (the higher the percentage the better). In this case, the Custom order was a little better than the default SMP order.