Performance Analysis Tools and Cray Reveal

Helen He
NERSC User Services Group

Feb 3, 2014
NUG 2014
Outline

• Performance Analysis Tools: brief introductions
  – IPM
  – Allinea MAP
  – Cray perftools and perftools-lite

• Cray Reveal: a tool to help adding OpenMP
Performance Analysis Tools
IPM (Integrated Performance Monitoring)

- IPM is a portable high level profiling tool, very low overhead.
- Reports: hardware counter data, MPI function timings, and memory usage, ...
- Available on Hopper, Edison, and Carver.
- Sample IPM output (in job stdout)

```bash
##IPM2v0.xx#########################################
# command : ./poisson_mpi.ipm
# start   : Sat Feb 01 00:58:05 2014   host   : nid01888
# stop    : Sat Feb 01 00:58:25 2014   wallclock : 20.10
# mpi_tasks : 24 on 1 nodes  %comm  : 20.19
# mem [GB]  : 0.92      gflop/sec : 8.85
#
# : [total]    <avg>  min  max
# wallclock : 482.13   20.09  20.08  20.10
# MPI      : 97.32     4.06   3.83   4.39
# %wall    :
# MPI      : 20.19     19.07  21.86
# #calls   :
# MPI      : 2266248   94427  69094  96730
# mem [GB]  : 0.92     0.04   0.04   0.04
#
# MPI_Allreduce   64.04  663264  13.51
# MPI_Waitall     20.21  331632  4.26
# MPI_Isend       0.52   635628  0.11
# MPI_Irecv      0.26   635628  0.06
# MPI_Comm_size   0.00   24     0.00
# MPI_Comm_rank   0.00   24     0.00
# MPI_Init    0.00     24     0.00
# MPI_Finalize  0.00     24     0.00
##IPM2v0.xx#########################################
```
Allinea MAP

- Allinea MAP is a parallel MPI profiler with GUI, small overhead.
- Reports: Memory usage, MPI usage, CPU time, CPU instructions, I/O, etc. as a function of time.
- Available on Hopper, Edison, and Carver.
- Documentations:
  - http://www.allinea.com/products/map/
- Sample screen shots
  - Source code mapping with time spent
  - More metrics
  - Load balance info
  - Can zoom in on a specific time interval
Cray perftools and perftools-lite

- Cray perftools is a performance analysis tool available on Cray systems.
- Reports: execution time, memory high water mark, aggregate FLOPS rate, top time consuming user function, MPI information, IO information, hardware performance counters, load balance ...
- Perftools-lite is a simplified version of perftools which reports basic information automatically with simple steps. Recommend to start with the lite version.
- Available on Hopper and Edison.
- Documentations:
  - https://www.nersc.gov/users/software/debugging-and-profiling/crpat/
- Sample perftools-lite output (in job stdout)

```
CrayPat/X: Version 6.1.3 Revision 12145 (xf12007) 11/18/13 21:56:10
Experiment: lite_sample_profile
Number of PEs (MPI ranks): 240
Numbers of PEs per Node: 24 PEs on each of 10 Nodes
Numbers of Threads per PE: 1
Number of Cores per Socket: 12
Execution start time: Sun Feb 2 13:38:33 2014
System name and speed: nid01665 2401 MHz
...Wall Clock Time: 290.822940 secs
High Memory: 243.36 MBytes
MFLOPS (aggregate): Not supported (see observation below)
I/O Read Rate: 46.30 MBytes/Sec
I/O Write Rate: 5.91 MBytes/Sec
```

Table 1: Profile by Function Group and Function (top 10 functions shown)
```
<table>
<thead>
<tr>
<th>Function Group</th>
<th>Function</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0%</td>
<td></td>
<td>28484.6</td>
</tr>
<tr>
<td>61.8%</td>
<td></td>
<td>17598.4</td>
</tr>
<tr>
<td>36.3%</td>
<td></td>
<td>10328.2</td>
</tr>
<tr>
<td>29.6%</td>
<td></td>
<td>8432.1</td>
</tr>
<tr>
<td>9.0%</td>
<td></td>
<td>2571.0</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

A file named MPICH_RANK_ORDER.USER_Samp was generated

Table 2: File Input Stats by Filename (top 10 files shown)
```
```

Table 3: File Output Stats by Filename (top 10 files shown)
```
Cray Reveal
What is Reveal

• A tool developed by Cray to help developing the hybrid programming model

• Part of the Cray Perftools software package, based on:
  – Performance data collected from Craypat
  – Cray CCE program library for loopmark and source code analysis

• Identifies top time consuming loops, with compiler feedback on dependency and vectorization

• Loop scope analysis provides variable scope and compiler directive suggestions for inserting OpenMP parallelism to a serial or pure MPI code

• Only works under PrgEnv-cray, available on Hopper and Edison.
Steps to use Reveal

- **Load the user environment**
  - % module swap PrgEnv-intel PrgEnv-cray  (Edison)
  - or % module swap PrgEnv-pgi PrgEnv-cray  (Hopper)
  - % module unload darshan
  - % module load perftools
- **Generate loop work estimates**
  - % ftn –c –h profile_generate myprogram.f90
  - % ftn –o myprogram –h profile_generate myprogram.o
    - *Good to separate compile and link to keep object files*
    - *Optimization flags disabled with –h profile-generate*
  - % pat_build –w myprogram  (-w enables tracing)
    - *It will generate executable “myprogram+pat”*
  - Run the program “myprogram+pat”
    - *It will generate one or more myprogram+pat+...xf files*
  - % pat_report myprogram+pat...xf > myprogram.rpt
    - *It will generate myprogram+pat....ap2 file*
Steps to use Reveal (2)

• Generate a program library
  – % ftn –O3 –hpl=myprogram.pl –c myprogram.f90
  – Optimization flags can be used
  – Build one source code at a time, with “-c” flag
  – Use absolute path for program library if sources are in multiple directories
  – User needs to clean up program library from time to time

• Save a copy of the original code

• Launch Reveal
  – % reveal myprogram.pl myprogram+pat...ap2
  – See a quick demo
Right click to select loops

Start Scoping
Reveal helps to start adding OpenMP

- Only under PrgEnv-crays with CCE compiler
- Start from most time consuming loops first
- Insert Reveal suggested OpenMP directives
- There will be unresolved and incomplete variable scopes
  - User still needs to understand OpenMP, and resolves the issues.
  - Compile the new code with OpenMP enabled. OK under any PrgEnv. Resolve compilation warnings and errors. Verify correctness.
  - Compare performance between the original and new OpenMP enabled codes.
- Repeat as necessary
- No OpenMP tasks, barrier, critical, atomic regions, etc
More information

- **% module load training**
  - See example codes, reports, detailed steps in README at:
    $EXAMPLES/NUG/Reveal

- **Documentations:**
  - % man reveal (when the “perftools” module is loaded)
  - Using Cray Performance Measurement and Analysis Tools
    [http://docs.cray.com/books/S-2376-613/S-2376-613.pdf](http://docs.cray.com/books/S-2376-613/S-2376-613.pdf)
  - More detailed presentation at 2013 “Performance on Edison” training
Thank you.