NCCS

Application Performance Tools on Discover

Tyler Simon

21 May 2009

NCCS

Overview

1. ftnchek - static Fortran code analysis
2. Cachegrind - source annotation for cache use
3. Ompp - OpenMP profiling
4. IPM – MPI performance monitoring
5. Intel MPI – MPI performance Statistics
6. resmon-System level resource monitoring

NCCS

ftnchek

- From Fordham University
  http://www.dsm.fordham.edu/~ftnchek/

- Variables which are never used

- Uninitialized variables

- Variables which are not declared may not have the intended type

- It is not intended to catch all syntax errors.

NCCS

Ftnchek usage

```
>/usr/local/other/ftnchek/ftnchek-3.3.1/bin/ftnchek mem1d.f
```

```
Warning near line 15 col 14 file /home/tasimon/memtest/mem1d.f:
real expr
 OPERATIONS/1000000.0 promoted to real*8 MOPS: may not give
desired precision
     26         LoopSize= BASIS * (2**Length)
                        ^
Warning near line 26 col 17 file /home/tasimon/memtest/mem1d.f:
intg*8 expr
 BASIS*(2**LENGTH) truncated to intg*4 LOOPSIZE
     27         LoopIterations = 2**(max-Length)
                               ^
Warning near line 27 col 24 file /home/tasimon/memtest/mem1d.f:
intg*8 expr
 2**(MAX-LENGTH) truncated to intg*4 LOOPITERATIONS
     35         PRINT 100, LoopSize*4,
WTIME,MOPS/WTIME,CTIME,CTIME/WTIME
                                                      ^
Warning near line 35 col 42 file /home/tasimon/memtest/mem1d.f:
promotion may
 not give desired precision: real*8 MOPS / real WTIME
```

NCCS

Valgrind, Cachegrind

- From University of Cambridge
  http://valgrind.org/docs/manual/cg-manual.html

- Performs detailed simulation of the I1, D1 and L2 caches in your CPU

- Allows you to accurately pinpoint the sources of cache misses in your code.

- Identifies the number of cache misses, memory references and instructions executed for each line of source code.

- Provides per-function, per-module and whole-program summaries.

- Useful with programs written in any language.

- Cachegrind runs programs about 20--100x slower than normal.

Cachegrind usage

NCCS

```
>mpiicc –g –o app app.c
>/usr/local/other/valgrind/src/3.3.1/coregrind/valgrind  –tool=cachegrind
./app
```

```
>cg_annotate –auto=yes cachegrind.out.<pid>
```

```
I1 cache:         32768 B, 64 B, 8-way associative
D1 cache:         32768 B, 64 B, 8-way associative
L2 cache:         1048576 B, 64 B, 4-way associative
Command:          ./loop_reorder
Data file:        cachegrind.out.14993
Events recorded:  Ir I1mr I2mr Dr D1mr D2mr Dw D1mw D2mw
Events shown:     Ir I1mr I2mr Dr D1mr D2mr Dw D1mw D2mw
Event sort order: Ir I1mr I2mr Dr D1mr D2mr Dw D1mw D2mw
Auto-annotation:  off

--------------------------------------------------------------------------------
Ir  I1mr  I2mr            Dr          D1mr          D2mr            Dw    D1mw    D2mw
--------------------------------------------------------------------------------
65,753,894,034 9,074 5,634 29,058,213,941 2,286,259,566 2,284,949,320 2,176,170,055 400,235 399,128
PROGRAM TOTALS

--------------------------------------------------------------------------------
Ir I1mr I2mr       Dr       D1mr          D2mr            Dw    D1mw    D2mw  file:function
--------------------------------------------------------------------------------
33,349,140,816   13   13 15,066,054,781 1,077,207,053 1,075,929,094 1,082,163,291       6       6
/gpfsm/home/tasimon/tyler_bmarks/loop_reorder.F:hpijkblock_
32,217,502,794    8    8 13,960,742,929 1,209,009,145 1,209,009,145 1,074,791,452       5       5
/gpfsm/home/tasimon/tyler_bmarks/loop_reorder.F:hpnonblock_
```

NCCS

OpenMP Profiling with ompp

- From University of Tennessee and UC Berkeley
http://www.cs.utk.edu/~karl/ompp.html
- Profiling tool for OpenMP applications written in C/C++ or FORTRAN.
- Profiling report available immediately after program termination
- Supports the measurement of hardware performance counters using PAPI
- Overhead analysis and detection of common inefficiency situations

NCCS

ompp Usage

Location:

```
/usr/local/other/ompp0.6.7/kinst/
```

```
>kinst-ompp ifort -openmp foo.F bar.F -o foobar
>export OMP_NUM_THREADS=8
>./foobar
```

Manual instrumentation (FORTRAN):

```
!$POMP INST BEGIN(name)
CALL DO_WORK()
!$POMP INST END(name)
```

Manual instrumentation (C):

```
#pragma pomp inst begin(region1)
do_work();
#pragma pomp inst end(region1)
```

NCCS

ompp Output

```
----------------------------------------------------------------------
----     ompP General Information     --------------------------------
----------------------------------------------------------------------
Start Date      : Wed May 20 13:58:20 2009
End Date        : Wed May 20 14:02:30 2009
Duration        : 250.69 sec
Application Name: matmult
Type of Report  : final
User Time       : 1610.84 sec
System Time     : 343.31 sec
Max Threads     : 8
ompP Version    : 0.6.5
ompP Build Date : Apr 28 2009 11:22:47
PAPI Support    : not available

----------------------------------------------------------------------
----     ompP Region Overview     -------------------------------------
----------------------------------------------------------------------
PARALLEL: 1 region:
 * R00001 omp_mm.c (29-69)

LOOP: 4 regions:
 * R00003 omp_mm.c (42-45)
 * R00004 omp_mm.c (46-49)
 * R00005 omp_mm.c (50-53)
 * R00006 omp_mm.c (61-68)

USER REGION: 1 region:
 * R00002 omp_mm.c (39-55) ('region1')
```

NCCS

Integrated Performance Monitor (IPM)

- Developed by David Skinner at NERSC
  http://ipm-hpc.sourceforge.net/
- A portable profiling infrastructure which reports on the execution of a parallel job.
- Reports hardware counters data, MPI function timings, and memory usage.
- Provides a low overhead means to generate scaling studies or performance data.
- Performance summaries to stdout
- Web accessible summary of all your IPM jobs.
- Main objectives of IPM are ease-of-use and scalability in performance analysis.

NCCS

IPM Usage

- Include ipm library at link or compile stage.

```
>mpicc –o mpiapp mpiapp.c –L/usr/local/other/ipm/lib –lipm
>mpirun –np <procs> ./mpiapp
```

- When run as above, you will get a <username>.<epoch>.<jobid>.0 file in your directory and you will get a brief report in your job output file

- Parse the output to create a series of web pages and images.

```
>setenv IPM_ROOT /usr/local/other/ipm
>setenv IPM_KEYFILE $IPM_ROOT
>$IPM_ROOT/bin/ipm_parse –html <username>.<epoch>.<Jobid>.0
```

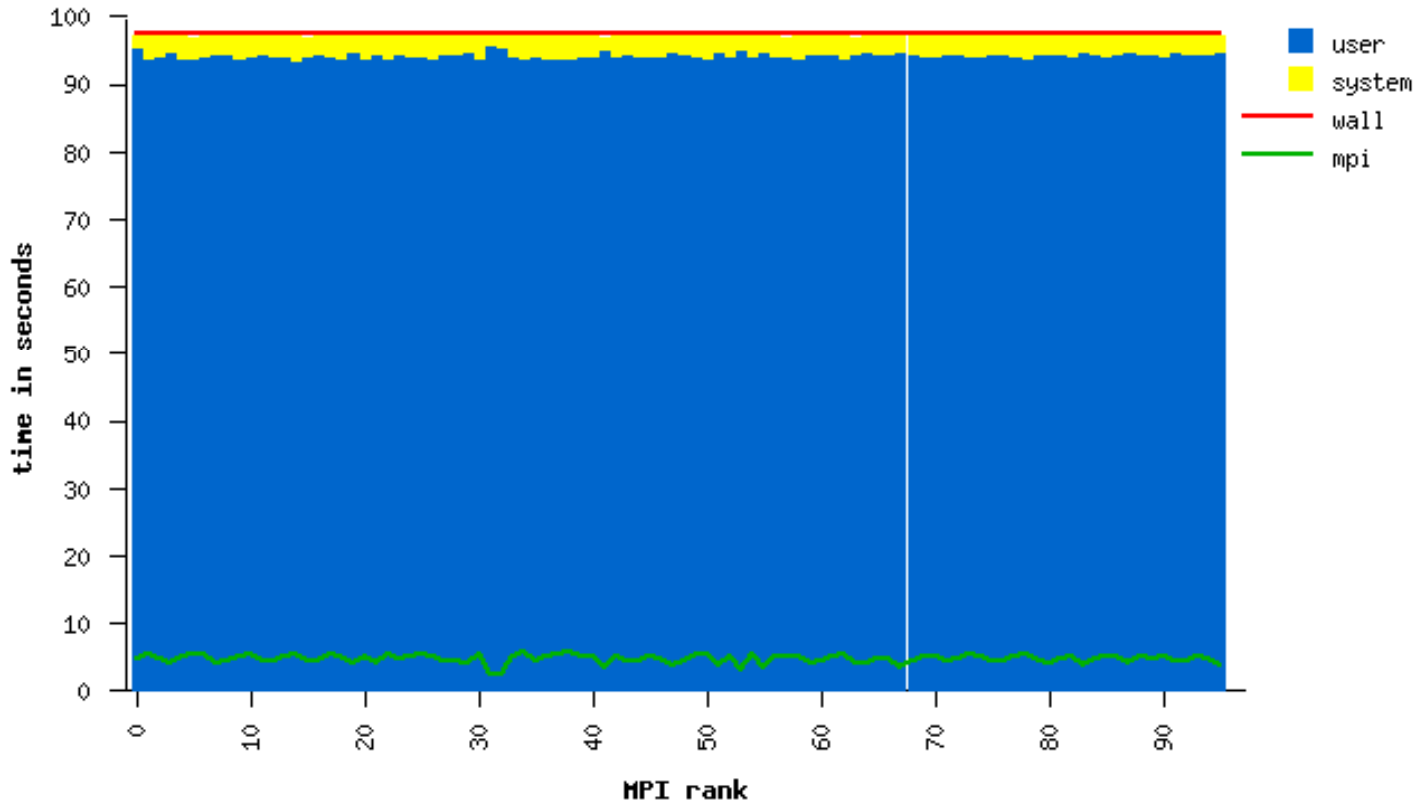
NCCS

Text output (fvcore)

```
##IPMv0.923##########################################################################
#
# command : unknown (completed)
# host    : borgh043/x86_64_Linux          mpi_tasks : 96 on 12 nodes
# start   : 03/25/09/17:17:27              wallclock : 228.741434 sec
# stop    : 03/25/09/17:21:16              %comm     : 12.33
# gbytes  : 0.00000e+00 total              gflop/sec : 0.00000e+00 total
#
##################################################################################
# region  : *       [ntasks] =     96
#
#                        [total]        <avg>          min          max
# entries                     96            1            1            1
# wallclock               21956.1       228.71      228.658      228.741
# user                   20275.8      211.206      205.521      222.694
# system                  1647.48      17.1612      5.67635      22.7454
# mpi                    2706.47      28.1924      8.70475      39.2119
# %comm                                12.325      3.80683      17.1431
#
#                            [time]       [calls]       <%mpi>      <%wall>
# MPI_Wait                 2578.26    3.6325e+06        95.26        11.74
# MPI_Allreduce              69.8645         49536         2.58         0.32
# MPI_Recv                   22.8316          9216         0.84         0.10
# MPI_Isend                  19.9544   1.82086e+06         0.74         0.09
# MPI_Irecv                  8.80108   1.81164e+06         0.33         0.04
# MPI_Barrier                4.51097          1152         0.17         0.02
# MPI_Bcast                  1.49277          9312         0.06         0.01
# MPI_Gatherv               0.728684           288         0.03         0.00
# MPI_Comm_size            0.0213632            96         0.00         0.00
# MPI_Comm_rank            0.0083791            96         0.00         0.00
##################################################################################
```

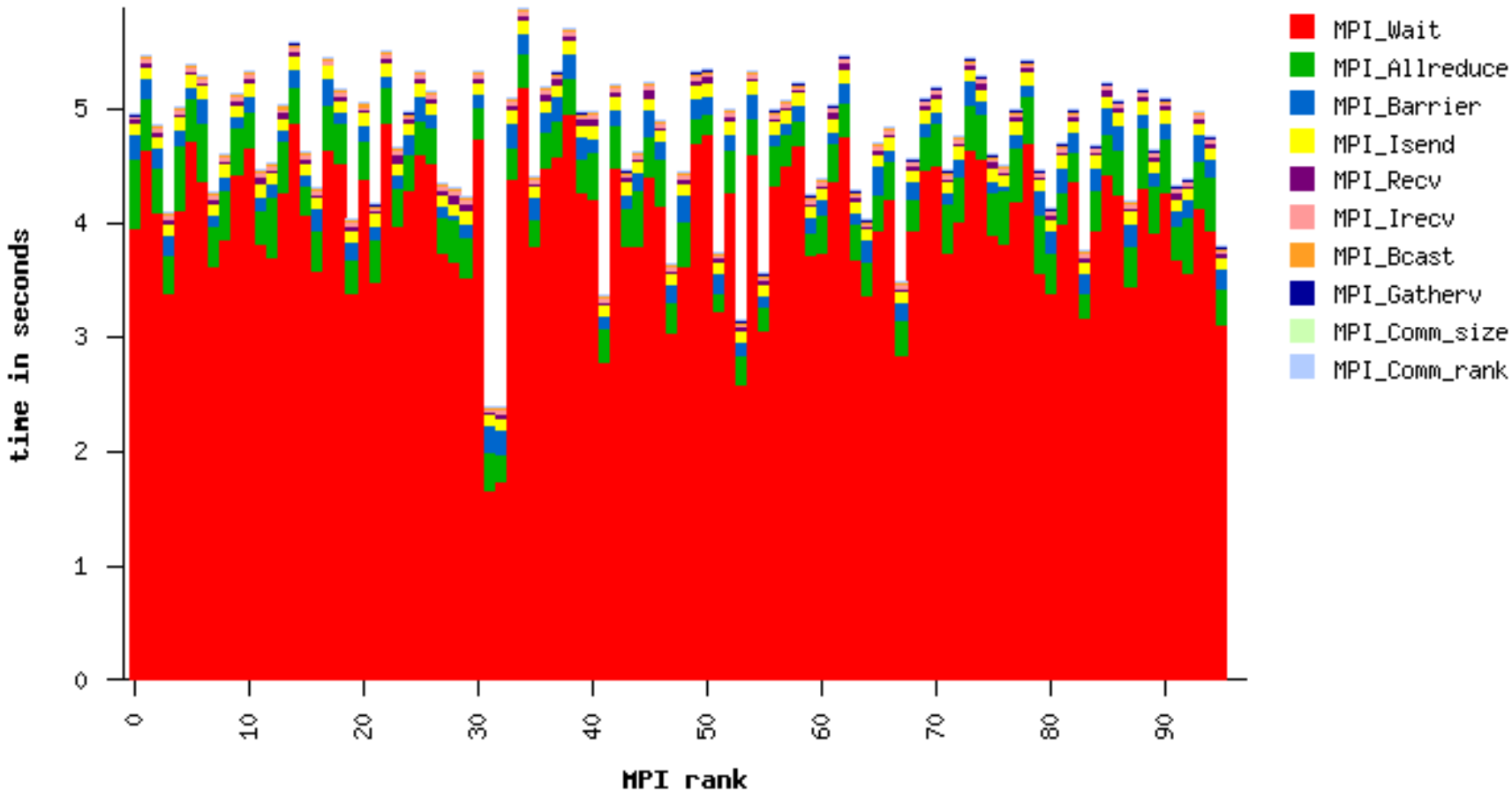
NCCS

CPU time and MPI time 96 core run



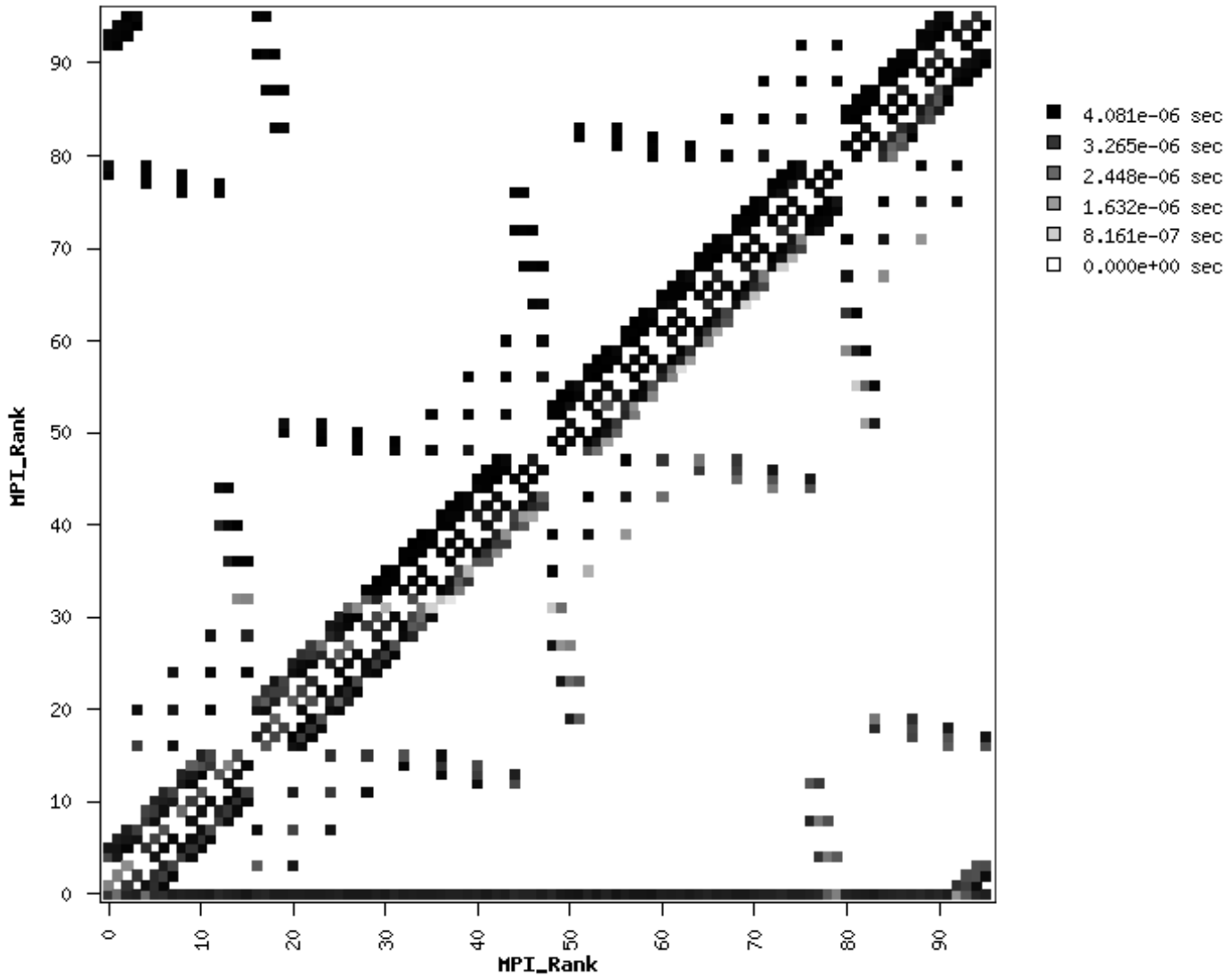
NCCS

Time spent in MPI by Rank



NCCS

MPI Communication Topology



NCCS

Intel MPI Statistics

- *setenv IMPI_STATS <level>*
- *<level> Indicate level of statistics information provided*
- 0 - Do not collect any statistics. This is the default value
- 1 - Output the amount of data sent by each process
- 2 - Output the number of calls and amount of transferred data
- 3 - Output statistics combined according to the actual arguments
- 4 - Output statistics defined by a buckets list
- 10 - Output collective operation statistics for all communication contexts

Intel MPI Stats with fvcore

NCCS

export I_MPI_STATS=2

```
Intel(R) MPI Library Version 3.2
____ MPI Communication Statistics ____
Stats level: 2
P2P scope:< FULL >
Collectives scope:< FULL >
~~~~ Process 0 of 24 on node borgh001
Data Transfers
Src     Dst     Amount(MB)      Transfers
----------------------------------------
000 --> 000     0.000000e+00    0
000 --> 001     1.560249e+03    38774
000 --> 002     1.560063e+03    32595
000 --> 003     1.026902e+02    31850
000 --> 004     9.918213e-05    2
…
```

```
Process 0 Communication Activity
Operation       Volume(MB)      Calls
----------------------------------------
P2P
Csend           1.856689e-01    6304
Send            6.548324e+03    225830
Bsend           0.000000e+00    0
Rsend           0.000000e+00    0
Ssend           0.000000e+00    0
Collectives
Barrier         0.000000e+00    12
Bcast           9.918213e-05    25
Gather          0.000000e+00    0
Scatter         0.000000e+00    0
Reduce          0.000000e+00    0
Alltoall        0.000000e+00    0
Allgather       0.000000e+00    0
Allreduce       1.843185e-01    6160
Reduce_scatter  0.000000e+00    0
Scan            0.000000e+00    0
Exscan          0.000000e+00    0
Gatherv         1.547241e-02    3
```

Resource Monitor (resmon)

NCCS

- tyler.simon@nasa.gov
- Prints a record of resource usage on a per minute basis
- Recording starts via application MPI_INIT or manual instrumentation
- Works for Fortran and C programs
- Per core cpu usage as a percent, and average
- Per node system memory usage as a percent and total
- Per node IO statistics
  - Total number disk transfers per second as well as reads and writes
  - Total blocks written per second
  - Total blocks read per second
- Total number of inodes used per minute
  - Tells us how many files were opened over a period

- Process Memory map “pmap” output

NCCS

resmon Usage

- Include the resmon library at link or compile stage.

```
>mpicc –o mpiapp mpiapp.c –L/usr/local/other/lib/ –lresmon
>mpirun –np <procs> ./mpiapp
```

- You can manually instrument non MPI code by using:

```
monitor_start("Region name");
monitor_stop("Region name");
```

- When run as above, you will get a "*resusage.nodeid*" in your directory and you will get a brief report in your job output file

- If you would like to see additional features please let me know.

A 1 minute disk write

NCCS

```
Linux 2.6.16.53-0.16-smp (borgi028)     05/06/2009
#CPU Stats
10:40:01 AM       CPU     %user     %nice   %system   %iowait     %idle
10:41:01 AM       all     25.01      0.00      0.00      0.04     74.95
10:41:01 AM         0     99.98      0.00      0.02      0.00      0.00
10:41:01 AM         1    100.00      0.00      0.00      0.00      0.00
10:41:01 AM         2      0.02      0.00      0.02      0.02     99.95
10:41:01 AM         3      0.02      0.00      0.00      0.07     99.92
10:41:01 AM         4      0.00      0.00      0.00      0.00    100.00
10:41:01 AM         5      0.02      0.00      0.00      0.00     99.98
10:41:01 AM         6      0.02      0.00      0.03      0.25     99.70
10:41:01 AM         7      0.00      0.00      0.02      0.00     99.98
Average:          all     25.01      0.00      0.00      0.04     74.95

#I O Stats
10:40:01 AM       tps      rtps      wtps   bread/s   bwrtn/s
10:41:01 AM      1.08      0.00      1.08      0.00     18.66
Average:         1.08      0.00      1.08      0.00     18.66

# Memory Stats
10:40:01 AM kbmemfree kbmemused  %memused kbbuffers  kbcached kbswpfree kbswpused  %swpused  kbswpcad
10:41:01 AM  15397424   1039692      6.33    150164    103748   7874544    318596      3.89    305320
Average:     15397424   1039692      6.33    150164    103748   7874544    318596      3.89    305320
# Inode stats
10:40:01 AM  dentunusd   file-sz  inode-sz  super-sz %super-sz  dquot-sz %dquot-sz  rtsig-sz %rtsig-sz
10:41:01 AM  35610      1792      5784         0      0.00         0      0.00         0      0.00
Average:     35610      1792      5784         0      0.00         0      0.00         0      0.00
```

NCCS

Process performance, memory map

```
*** node borgg059 pid 3791 memory map ***
3791: loop_reorder
START       SIZE     RSS   DIRTY PERM MAPPING
00400000    536K    228K      0K r-xp /gpfsm/dhome/tasimon/memtest/monitor/loop_reorder
00585000     16K     12K     12K rwxp /gpfsm/dhome/tasimon/memtest/monitor/loop_reorder
00589000  25136K  24944K  24944K rwxp [heap]
2b3eed2e6000    108K     96K      0K r-xp /lib64/ld-2.4.so
2b3eed301000      8K      8K      8K rwxp [anon]
2b3eed400000      8K      8K      8K rwxp /lib64/ld-2.4.so
2b3eed402000   2368K    668K      0K r-xp /usr/local/intel/mpi/3.2.011/lib64/libmpi_dbg.so.3.2
2b3eed652000   1020K      0K      0K ---p /usr/local/intel/mpi/3.2.011/lib64/libmpi_dbg.so.3.2
2b3eed751000     60K     60K     60K rwxp /usr/local/intel/mpi/3.2.011/lib64/libmpi_dbg.so.3.2
…
2b3eee7c2000      4K      4K      4K rwxp /usr/lib64/libmthca-rdmav2.so
2b3eee7c3000      4K      0K      0K ---p [anon]
2b3eee7c4000 15390540K   2016K   2016K rwxp [anon]
7fffbd7ae000     88K     32K     32K rwxp [stack]
ffffffffff600000   8192K      0K      0K ---p [vdso]
Total:   15444568K  29876K  27936K

Crosscheck from ps output:
%MEM   RSS    PSR POL NLWP %CPU     ELAPSED RUSER    CMD
0.1        29872  4     TS       2    99.8          04:24    tasimon  loop_reorder
```

NCCS

Questions?

On to the Demos!