(HPSC **5576** ELIZABETH JESSUP)

HIGH PERFORMANCE SCIENTIFIC COMPUTING

:: Homework / 12

:: Student / Florian Rappl

1 problem / **10** points

Selected Problem: **2**

Task:

Trestles provides two file systems: high-performance storage via GPFS (*/oasis/$USER*) and node-local flash-based storage (*/scratch/$USER/$PBS_JOBID*). In this exercise, you'll run a few quick performance tests on Trestles to determine the rough performance of SSD vs. GPFS for a few types of I/O traffic. Here's an interactive job that I ran to verify that /scratch really is SSD, and not just a directory on a standard hard drive:

```
[matthewww@trestles matthewww]$ qsub -l nodes=1 -l walltime=1:00:00 -I -q
normal
qsub: waiting for job 18848.trestles-fe1.sdsc.edu to start
qsub: job 18848.trestles-fe1.sdsc.edu ready
[matthewww@trestles-1-16 ~]$ ls -ld /scratch
lrwxrwxrwx        1 root root        18 Mar  9 20:40 scratch ->
/state/partition1/
[matthewww@trestles-1-16 ~]$ mount | grep state
/dev/sda5 on /state/partition1 type ext3 (rw)
[matthewww@trestles-1-16 ~]$ cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
 Vendor: ATA          Model: INTEL SSDSA2M120 Rev: 2CV1
 Type:   Direct-Access                    ANSI SCSI revision: 05
[matthewww@trestles-1-16 ~]$ exit
qsub: job 18848.trestles-fe1.sdsc.edu completed
```

Note that seeing the "qsub: job … completed" confirms that your job is done, and that you're no longer being charged for the node.

Procedure:

Download the IOR parallel file system benchmark from SourceForge. Expand and compile IOR using the following commands:

```
tar xzvf /tutorial/public/IOR-2.10.3.tgz
cd IOR/src/C
make
```

Run a quick test to make sure it works. Note that the -o parameter is the file system being tested. This test will read/write a 1GB file (-b 1024m) sequentially in 32MB (-t 32m) chunks, for three iterations (-i 3).

```
#PBS -l walltime=00:45:00
#PBS -l nodes=1
cd $PBS_O_WORKDIR

./IOR -F -o /oasis/flo_ra/test1 -i 3 -m -t 32m -b 1024m -d 0.01 | tee
_oasis_1024_32.${PBS_JOBID}.txt
./IOR -F -o /scratch/flo_ra/${PBS_JOBID}/test1 -i 3 -m -t 32m -b 1024m -d
0.01 | tee _scratch_1024_32.${PBS_JOBID}.txt
```

Run ./IOR --help or check the USAGE_GUIDE in the IOR/ directory for more information on the parameters.

Solutions

Question:

Try running tests with 32MB transfer sizes with a few different file sizes, such as: 1024 MB, 512 MB, and 128 MB. What do you notice about the reported performance? What might explain those results?

Answer:

The SSD is always faster than the GPFS, which was obvious from the beginning. However the difference in the section of reading performance is really small between those two (for the biggest test with 1024 MB). We notice that both systems are getting faster the more data they have to read or write. This could be explained by latency / a kind of heating up or getting warm delay. This would explain why the GPFS is getting a lot more benefit from having to read or write more data than the SSD system gets (it still gets a bit – but percent-wise not as much as the GPFS system).

Question:

Fix the file size at 1024 MB, and try running a test on each file system with a much smaller transfer size (maybe between 512 bytes and 1 KB). Which file system performs better with big transfers? Which performs better with small transfers? What ops/sec rate could each achieve?

Answer:

While the SSD system was a lot better at reading or writing large files than small files, the writing performance of the GPFS system was more or less independent of the file size. However the reading performance showed again the same behavior as the SSD system did. The maximum operations per second rate were:

	read	write
SSD (Scratch) (512 byte)	630765.86	296891.68
SSD (Scratch) (32 MB)	59.56	21.22
GPFS (Oasis) (512 byte)	304587.55	115245.37
GPFS (Oasis) (32 MB)	58.22	1.82

Question:

Given all of your measurements above, how does the SSD performance compare to the vendor specifications[1]? How does SSD compare to GPFS? How/when would you choose to use SSD vs. GPFS on Trestles?

[1] http://www.thenerds.net/INTEL.Intel_X25_M_120_GB_Internal_Solid_State_Drive_1_Pack_Retail.SSDSA2MH120G2K5.html?affid=8

Answer:

The vendor: specifies a maximum read transfer rate of 250 MB/s and a maximum write transfer rate of 70 MB/s. Our test showed that the drive is actually able to go far beyond this. However I think that those specifications are done over averaged writing processes. SSD drives are really quick in writing data and reading data, but not in writing files (i.e. manipulating the file directory). That can be directly seen by moving one big file to a flash disk (e.g. a standard USB stick) compared to several (hundreds) of small files with the same overall size. In our test we just wrote (and read) one confined block of data (which was probably not distributed but available as a huge block of data). We only had a directory manipulation in the beginning of the test.

I would pick the GPFS for manipulations from multiple sites (i.e. from more than one node), where we need the data after the program finished. The SSD makes sense for temporary data which is always only written from one node (has not to be the same node – but never multiple nodes at the same time). We did not test the multiple writing performances but since it's just a flash disk on a single node we can just do one write process at a certain time.

Question:

Do you think that your tests have revealed the underlying hardware's performance characteristics for either file system? If not, what might be a possible way to do so?

Answer:

I think it revealed part of the performances. Since GPFS is distributed all over the network we have to take wire contention and network throughput into consideration. To really reveal the characteristics of this system we would have to do a lot of tests beginning with network tests (contention, throughput and latency) as well as disk(s) to executing node relation(s). Then we would have to do separate non network tests with the disks alone and combine results in order to find maximum (theoretical) transfer rates. This is then to be checked. For the SSD we did most of the testing – however the already mentioned directory writing (creating a new file entry, etc.) actions still have to be tested. We also need to test the kind of latency of writing data. Accumulating all this data (which seems a lot) would be a lot of work and would consume a lot of time. For our practical purposes the tests done with these interpretations are enough.