Experience using the IO-500

George Markomanolis
KAUST Supercomputing Laboratory

2017-11-15
The Virtual Institute of I/O and the IO-500 BOF

Denver, Colorado, USA
Why to use/contribute to IO500 benchmark?

• It is a community effort, we need your feedback, if something does not work, report it and we will try to find a solution

• Even a submission of results is contribution, people can find how your storage performs

• Present the results to the users of your system and educate them how the hard cases perform, in order to avoiding similar approaches.

• It is fun:
  • Exploring various filesystems
  • Decide for your next procurement
  • Keep track of performance on the storage with the new upcoming technologies, is quite interesting
Challenges

• Debugging on two nodes, could be totally different experience than one node

• Python and MPI caused us a lot of issues, some burnt core hours
  • As result, some unfinished executions, did not erase the created data

• A library for parallel find, demands to find the mpicc command, Cray systems do not have mpicc, fixing by modifying manually the configure file.

• In some cases, configure could fail and we just had to load latest autotools module, or fix something.

• Some commands on not classic filesystems, maybe they do not report back what you expect
How to run IO-500

• git clone https://github.com/VI4IO/io-500-dev
• cd io-500-dev
• ./utilities/prepare.sh
• ./io500.sh (submit this script if you use a scheduler)
• email results to submit@io500.org
Modify IO-500

• Modify io500.sh accordingly, for example:

  io500_mpirun="mpirun"
  io500_mpiargs="-np 2"
  io500_ior_easy_params="-t 2048k -b 2g -F"
  io500_mdtest_easy_files_per_proc=25000
• Modify io500.sh accordingly, select which experiments to be executed:

```bash
io500_run_ior_easy="True"
io500_run_md_easy="True"
...
io500_run_md_hard_delete="True"
```

• For **valid** submission, you need to execute all the tests while the write phases should take at least 5 minutes
Modify IO-500 III

• Modify io500.sh accordingly, uncomment these lines and declare the path to your pfind wrapper:

```bash
#io500_find_mpi="True"
#io500_find_cmd="$PWD/bin/pfind"
```
Example of a not valid test case

[RESULT] BW  phase 1  ior_easy_write  96.133 GB/s : time 187.24 seconds
[RESULT] BW  phase 2  ior_hard_write  11.230 GB/s : time  46.79 seconds
[RESULT] BW  phase 3  ior_easy_read   109.249 GB/s : time 164.76 seconds
[RESULT] BW  phase 4  ior_hard_read   7.871 GB/s : time  66.74 seconds
[RESULT] IOPS phase 1 mdtest_easy_write  49.231 kiops : time  19.61 seconds
[RESULT] IOPS phase 2 mdtest_hard_write  15.444 kiops : time  17.05 seconds
[RESULT] IOPS phase 3 find               8.120 kiops : time  98.45 seconds
[RESULT] IOPS phase 5 mdtest_easy_stat   5.313 kiops : time 127.18 seconds
[RESULT] IOPS phase 6 mdtest_hard_stat   6.772 kiops : time  30.43 seconds
[RESULT] IOPS phase 7 mdtest_easy_delete 14.873 kiops : time  49.98 seconds
[RESULT] IOPS phase 8 mdtest_hard_read   45.599 kiops : time  10.16 seconds
[RESULT] IOPS phase 9 mdtest_hard_delete 30.776 kiops : time  11.84 seconds

Experience with IO500 benchmark

• With not proper tuning, the benchmark will finish either too fast or too slow
• Start tuning with small values and increase them till you find the ones that produce the required outcome
• Be sure that you have enough space for the output data
• Check form the IOR output if it recognizes correctly the number of processes and how many are used per node
• If the benchmark is too slow without reason, check if other users execute intensive I/O applications
• Be sure that you do not harm the system, try to execute the benchmark when the system is not too busy or during maintenance
• For the IOR Hard, you could stripe the corresponding folder
KAUST – Lustre – IO-500

• 1000 compute nodes, 16000 processes, 144 OSTs

• ior_easy_params="-t 2m -b 5440m"
• ior_hard_writes_per_proc=792
• mdtest_hard_files_per_proc=380
• mdtest_easy_files_per_proc=452
KAUST – Cray DataWarp – IO-500

• 300 compute nodes, 2400 processes, 268 DataWarp nodes

• `ior_easy_params="-t 2m -b 192616m"`
• `ior_hard_writes_per_proc=77872`
• `mdtest_hard_files_per_proc=1630`
• `mdtest_easy_files_per_proc=10800`
Profiling IO500 with Darshan I
IOR easy
Profiling IO500 with Darshan II

IOR easy

I/O Sizes

Count (Total, All Proc)

0 1e+07 2e+07 3e+07 4e+07 5e+07 6e+07 7e+07 8e+07

0-100 101-1K 1K-10K 10K-100K 100K-1M 1M-10M 10M-100M 1G+

Read Write

I/O Pattern

Ops (Total, All Proc)

0 1e+07 2e+07 3e+07 4e+07 5e+07 6e+07 7e+07 8e+07

Read Write

Total Sequential Consecutive
**Profiling IO500 with Darshan III**

IOR easy

<table>
<thead>
<tr>
<th>Most Common Access Sizes</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>access size</td>
<td>1048576</td>
</tr>
</tbody>
</table>

### File Count Summary
(estimated by I/O access offsets)

<table>
<thead>
<tr>
<th>type</th>
<th>number of files</th>
<th>avg. size</th>
<th>max size</th>
</tr>
</thead>
<tbody>
<tr>
<td>total opened</td>
<td>2304</td>
<td>30G</td>
<td>30G</td>
</tr>
<tr>
<td>read-only files</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>write-only files</td>
<td>2304</td>
<td>30G</td>
<td>30G</td>
</tr>
<tr>
<td>read/write files</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>created files</td>
<td>2304</td>
<td>30G</td>
<td>30G</td>
</tr>
</tbody>
</table>
Profiling IO500 with Darshan IV
IOR easy
Profiling IO500 with Darshan
IOR Hard

Timespan from first to last access on files shared by all processes

All processes

00:00:00 00:00:15 00:00:30 00:00:45 00:01:00 00:01:15 00:01:30 00:01:45 00:02:00 00:02:15 00:02:30 00:02:45
## Profiling IO500 with Darshan II

### IOR Hard

#### Most Common Access Sizes

<table>
<thead>
<tr>
<th>access size</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>47008</td>
<td>2534400</td>
</tr>
</tbody>
</table>

#### Variance in Shared Files

<table>
<thead>
<tr>
<th>File Suffix</th>
<th>Processes</th>
<th>Fastest</th>
<th>Slowest</th>
<th>σ</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Rank</td>
<td>Time</td>
<td>Rank</td>
</tr>
<tr>
<td>...r_hard/IOR_file</td>
<td>2304</td>
<td>48</td>
<td>67.670351</td>
<td>1989</td>
</tr>
</tbody>
</table>
Profiling IO500 with Darshan
MD Hard
Profiling IO500 with Darshan II
MD Hard

<table>
<thead>
<tr>
<th>Most Common Access Sizes</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>access size</td>
<td></td>
</tr>
<tr>
<td>3901</td>
<td>1382400</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File Count Summary</th>
<th>number of files</th>
<th>avg. size</th>
<th>max size</th>
</tr>
</thead>
<tbody>
<tr>
<td>total opened</td>
<td>1382400</td>
<td>3.9K</td>
<td>3.9K</td>
</tr>
<tr>
<td>read-only files</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>write-only files</td>
<td>1382400</td>
<td>3.9K</td>
<td>3.9K</td>
</tr>
<tr>
<td>read/write files</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>created files</td>
<td>1382400</td>
<td>3.9K</td>
<td>3.9K</td>
</tr>
</tbody>
</table>
Presenting data in radar chart

The best storage I/O system should be represented in a full diamond graph.
Conclusions

• Tuning the parameters, could take some time depending on the system and the experience
• The good news is that as community we can solve many issues
• Till now the IOR easy is considered the normal approach for procurement, however, this does not correspond to the real application
• We need a better way to understand the procurement of storage and IO500 seems to be in the right direction
• We plan some future additions, such as mix workload
• More submissions we have, the better to understand the various filesystems