HPS

Julian Kunkel

Analyzing IO500 Data

Agenda

IO500 Benchmark

- covers various HPC relevant naive/optimized patterns
  - Reports a bounding box for observable performance
  - Includes metadata, Data, and search patterns
  - Utilizes IOR, MDTest, MDWorkbench tools from IOR repository
- aims to document execution and systems in detail for the community
- allows to monitor historic improvements
- demonstrated that it fosters innovation
- prevents cheating and caching
  - By using stonewalling - algorithm during write:
    1. execute independently for a minimum runtime (300s by default)
    2. synchronize on number of elements written
    3. all processes catch up on the data
  - This represents bulk synchronous I/O well and includes stragglers

https://io500.org

Data

- IO500 focus is naturally on performance numbers (score) stored in summary

```
IO500 version io500-isc21_v2-12 (standard)
[RESULT]       ior-easy-write       37.714715 GiB/s : time 330.013 seconds
[RESULT]    mdtest-easy-write     1102.720950 kIOPS : time 323.643 seconds
[      ]            timestamp        0.000000 kIOPS : time 0.003 seconds
[RESULT]       ior-hard-write       19.741594 GiB/s : time 321.884 seconds
[RESULT]    mdtest-hard-write      343.169370 kIOPS : time 325.427 seconds
[RESULT]                 find     1239.415980 kIOPS : time 323.647 seconds
[RESULT]        ior-easy-read       16.746869 GiB/s : time 718.238 seconds
[RESULT]     mdtest-easy-stat     2016.176058 kIOPS : time 186.107 seconds
[RESULT]        ior-hard-read       13.336159 GiB/s : time 466.821 seconds
[RESULT]     mdtest-hard-stat      983.473306 kIOPS : time 126.634 seconds
[RESULT]   mdtest-easy-delete      741.151972 kIOPS : time 484.635 seconds
[RESULT]     mdtest-hard-read      629.058439 kIOPS : time 186.672 seconds
[RESULT]   mdtest-hard-delete      714.156350 kIOPS : time 328.290 seconds
[SCORE ] Bandwidth 20.193628 GiB/s : IOPS 863.692659 kiops : TOTAL 132.064713
```

- But there is more data stored and reported...

The Data Treasure

- INI file with detailed output per phase

```
[ior-easy-write]
t_start         = 2021-06-07 21:09:20
exe             = ./ior -C -Q 1 -g -G 778777466 -k -e -o /tmp/dfuse/datafiles/2021.06.07-21.09.20/ior-easy/ior_file_easy -O s
throughput-stonewall = 37.78
score           = 37.714715
t_delta         = 330.0134
t_end           = 2021-06-07 21:14:50
```

  - The performance of doing independent I/O is included (at stonewall)
  - Actual runtime is with 330s higher than runtime of 300s
- Per-Benchmark outputs as TXT files
  - Contain extra information such as time for opening/closing file (IOR)
  - Load balancing in find
- Per-Benchmark and rank statistics in CSVs
  - Allows to identify latency and delay behavior

Example CSV file (IOR)

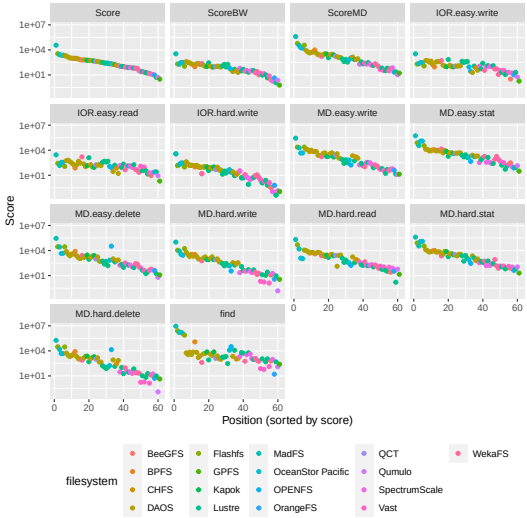
```
access,rank,runtime-with-openclose,runtime,throughput-withopenclose,throughput
read,0,4.1793304920e+01,4.1752141953e+01,5.6238672785e+07,5.6294117860e+07
read,1,4.1793386221e+01,4.1752182007e+01,5.6238563384e+07,5.6294063856e+07
read,2,4.1793395996e+01,4.1752168179e+01,5.6238550230e+07,5.6294082500e+07
read,3,4.1793278933e+01,4.1752177000e+01,5.6238707755e+07,5.6294070606e+07
read,4,4.1793433905e+01,4.1752176046e+01,5.6238499219e+07,5.6294071892e+07
read,5,4.1793218851e+01,4.1752174139e+01,5.6238788603e+07,5.6294074464e+07
read,6,4.1793239117e+01,4.1752176046e+01,5.6238761333e+07,5.6294071892e+07
read,7,4.1792804003e+01,4.1752181053e+01,5.6239346847e+07,5.6294065141e+07
read,8,4.1793321133e+01,4.1752182007e+01,5.6238650969e+07,5.6294063856e+07
```

Data Collection Process

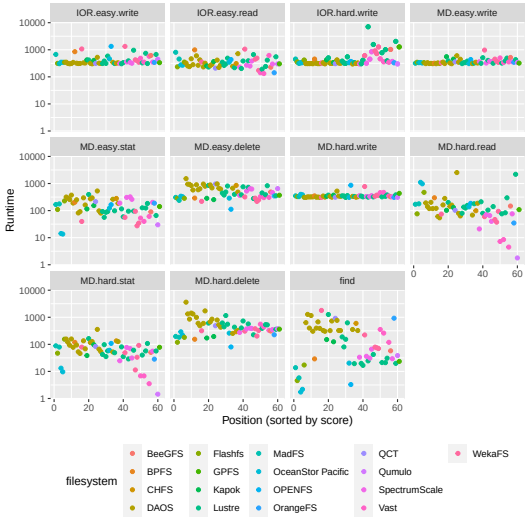
- Data between ISC21 and SC22 was used
  This used the C version and VI4IO page
- It is harder to retrieve the data with the new webpage
- Data was harvested using Python and analyzed using R
- Covers a total of 80 submissions

Outline

Overall Score of the Submissions

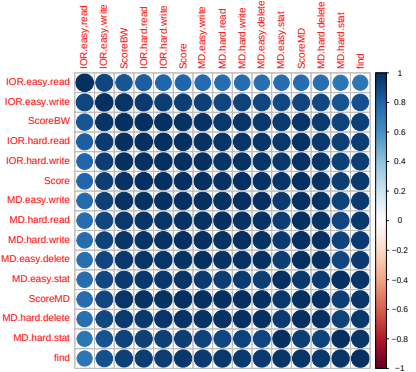


Overall Time of the Submissions



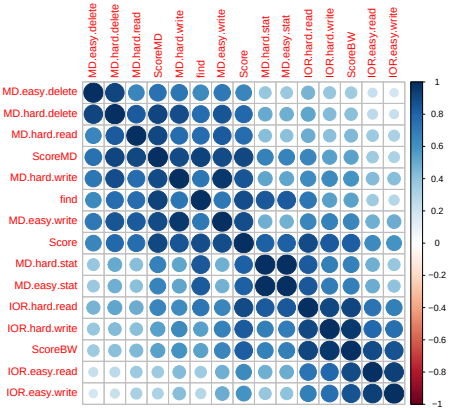
Simple Start: Correlation Between Scores

- Are different scores correlated?
- Sure, fast storage in one benchmark, usually means fast in the other
  Easy read-write, all others

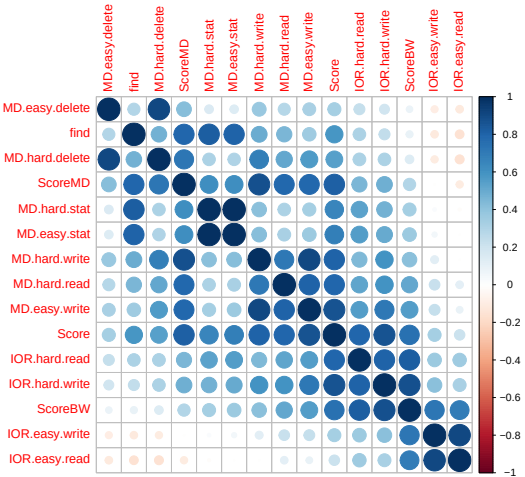


Correlation Between Scores - Normalized by Node Count

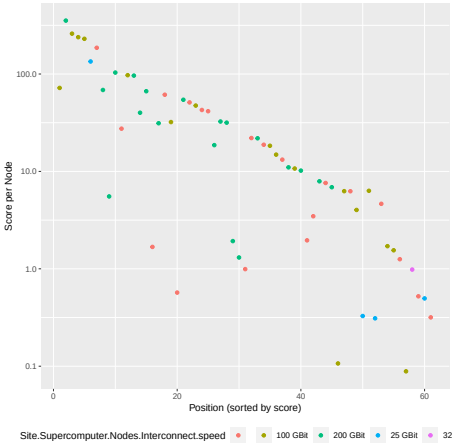
- Clusters: Metadata, score, stat, IOR



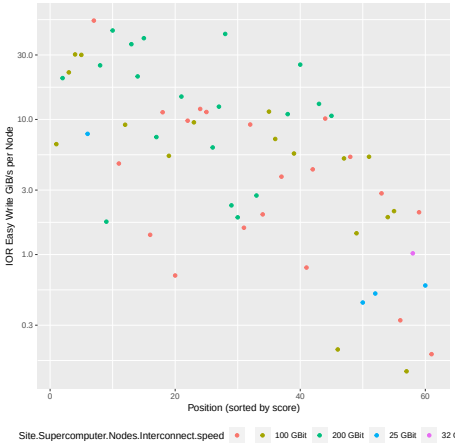
Correlation Between Scores - Normalized by Process Count



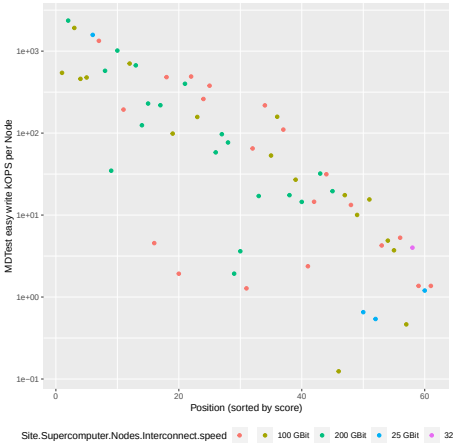
Score per Node and Network Speed



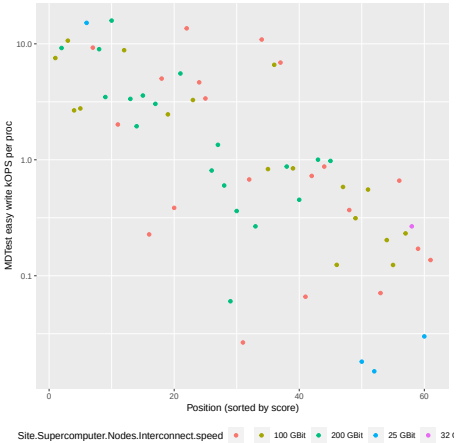
Score IOR Easy per Node



Score MDTest Easy per Node



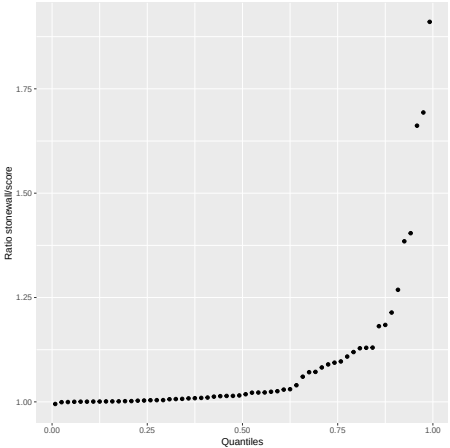
Score MDTest Easy per Proc



Outline

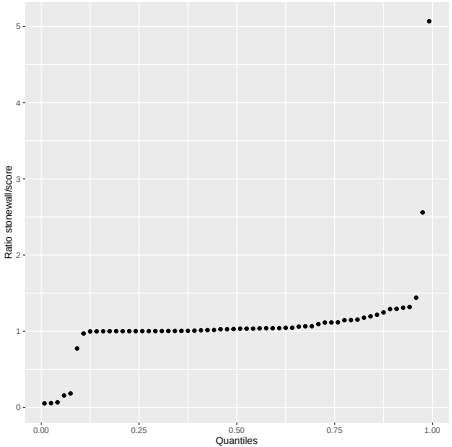
Stonewalling Performance: Stragglers - IOR Easy-Write

- Performance after 300s is usually higher due to parallelism
- Unbalanced writes lead to stragglers that delay storage



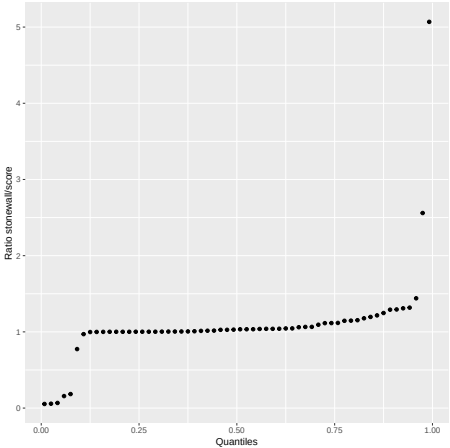
Stonewalling Performance: Stragglers - IOR Hard-Write

- Some file systems are seriously delayed
- Some are faster (how?)



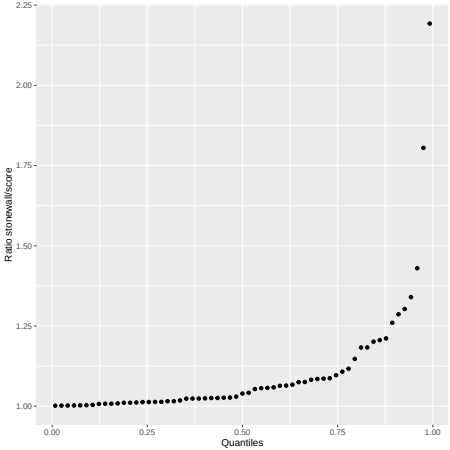
Stonewalling Performance: Stragglers - IOR Hard-Write

- Some file systems are seriously delayed
- All those are Weka, with $>>$ rank numbers, less congestion?

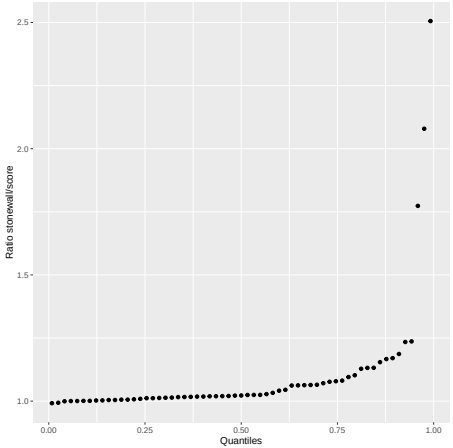


Stonewalling Performance: Stragglers - MDTest Easy Write

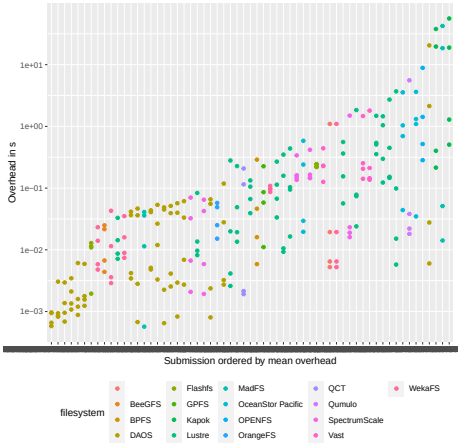
- Nothing unexpected here



Stonewalling Performance: Stragglers - MDTest Hard Write

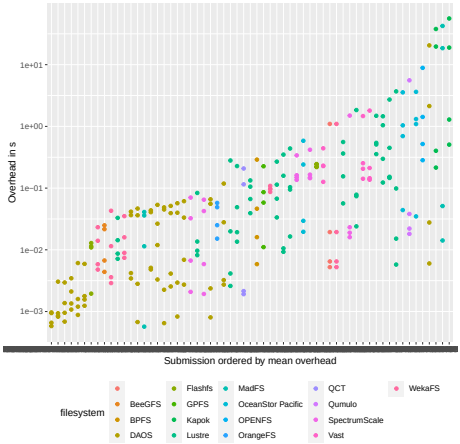


IOR Overhead Closing Files



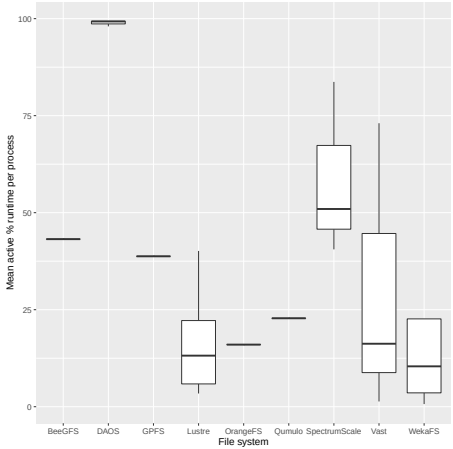
- DAOS has basically none
- Some fs has 10th of seconds (guess?)

IOR Overhead Closing Files

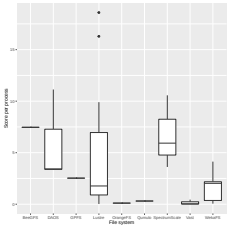


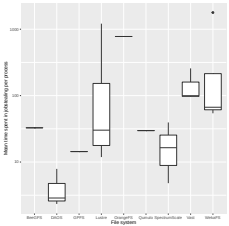
- DAOS has basically none
- Some fs has 10th of seconds
  MADFS, KAPOK, DAOS-FUSE

PFind Data - Meantime of Activity



- 100% means well balanced
- Some FS: one representative
- Kapok/Madfs not contained
  Uses own find...





Outline

Contribution

- This the first analysis of IO500 internals
- The IO500 contains more than just scores!
- Overhead of open/close
- Balancing of find/ior/mdtest (stonewalling)
  Unexpected result with Weka
- Corration shows that benchmarks are somewhat orthogonal
- Runtime of some file systems and phases is very slow