CLUSTERS @ HIM

HELMHOLTZ INSTITUT MAINZ

Dalibor Djukanovic
Helmholtz-Institut Mainz

12.12.2011
PANDA Collaboration Meeting
GSI, Darmstadt

HIMster - Intro

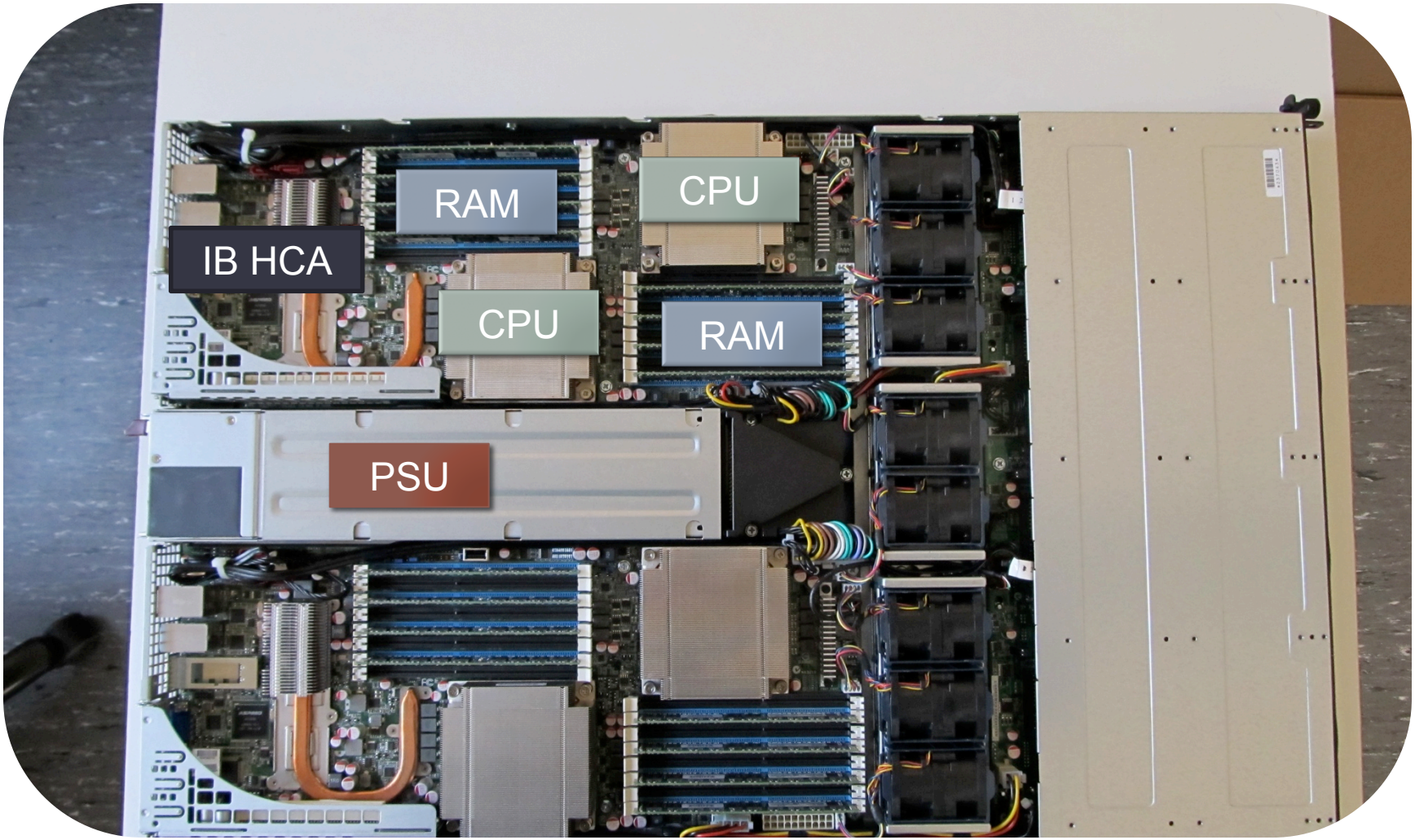
HIMster

- Cluster for experiment simulation = HIMster (HIM Cluster)
- Procurement started in late 2010
- Installation early 2011 by Megware
  - Delivery and initial setup = 2 days
- 3 Water cooled Racks of Hardware

Hardware

- 134 Compute Nodes, each node:
  - 2 AMD 6134 CPUs @ 2.3 GHz, 8 cores per CPU => 2144 cores
  - 2 Gbyte RAM / core, 14 Nodes with 4 Gbyte / core => 4.7 Tbyte
  - Infiniband Host Channel Adapter QDR 40 Gbit/s
  - No HDD (Diskless Setup)
- 1 Frontend (1 Database) Server
  - Same CPUs
  - 4 Gbyte Ram / core
  - Infiniband HCA, 10 Gbit-Nic
  - Redundant Power Supplies / Systemfans
  - Storage:
    - 15 k RPM SAS Drives (Frontend only approx. 1 TByte):
      - /home – systemwide NFS-mounted, quota 10 Gbyte
      - /cluster – systemwide NFS-mounted software dir
- Central Parallel File System
  - /data = 124 Tbyte disk space (7.2 k RPM)
  - Read/Write Performance 700 Mbyte/s (single client)
  - Read/Write Performance 1000 - 1500 Mbyte/s (aggregated multiclient)

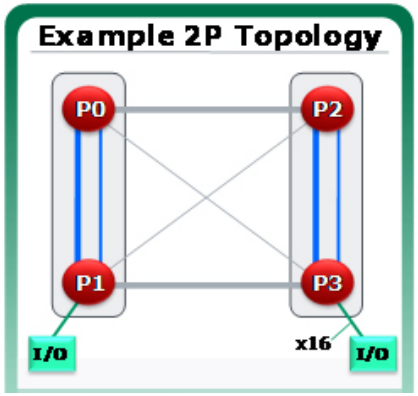
HIMster Node



HIMster Node - CPU

- 1 CPU (Package) = 2 Dies („Magny Cours“)
  - 4 cores @ 2.3 GHz per Die
  - 512 kb L2-cache, 5 MB shared L3-cache (per Die)
- Magny Cours = 4 NUMA (Non uniform memory access) zones

Mbyte/s	0	1	2	3
0	12483	7312	6928	6590
1	7605	12267	6629	7043
2	7016	6620	12260	7322
3	6708	7047	7286	12312

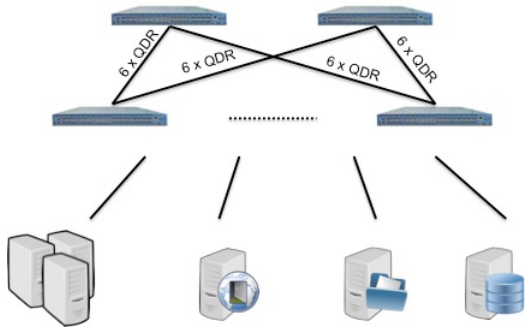


- „Scatter Jobs“ across Numa Nodes (avoid foreign memory)

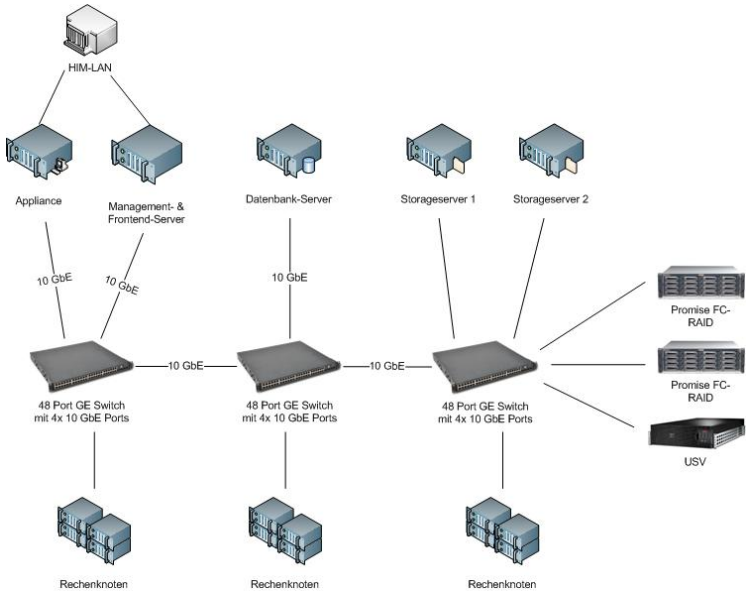
  => Nodeallocation policy, Process Pinning

HIMster - Network

1. Infiniband Network:
   - Fat Tree
   - Blocking Factor 2:1
   - 40 Gbit/s = 4 x QDR
   - 36-port managed switches (modular)



2. Gbit Network:
   - Administration
   - Interactive Usage
   - Switches 10 Gbit uplink



HIMster - Network

- High Bandwidth - 2.9 Gbyte/s (unidirectional)
- Low Latency - approx. 1.4 microseconds

```
[dalibor@frontend osu_benchmarks]$ mpirun -np 2 -hostfile ./
hostfile ./osu_bw
# OSU MPI Bandwidth Test v3.1.2
# Size        Bandwidth (MB/s)
1                    2.20
2                    4.39
4                    8.92
8                   17.81
16                  34.97
32                  65.83
64                 127.64
128                246.90
256                469.72
512                787.50
1024              1309.41
2048              1875.94
4096              2365.92
8192              2578.63
16384             2417.19
32768             2644.08
65536             2767.79
131072            2833.08
262144            2866.21
524288            2883.92
1048576           2893.15
2097152           2905.48
4194304           2904.55
```

```
[dalibor@frontend osu_benchmarks]$ mpirun -np 2 -hostfile ./
hosts ./osu_latency
# OSU MPI Latency Test v3.1.2
# Size            Latency (us)
0                    1.38
1                    1.40
2                    1.40
4                    1.40
8                    1.48
16                   1.49
32                   1.56
64                   1.61
128                  2.47
256                  2.66
512                  3.05
1024                 3.75
2048                 5.17
4096                 6.43
8192                 9.42
16384               13.17
32768               18.41
65536               29.76
131072               52.19
262144               97.74
524288              188.18
1048576              369.60
2097152              730.50
4194304             1463.14
```

Storage - Topology

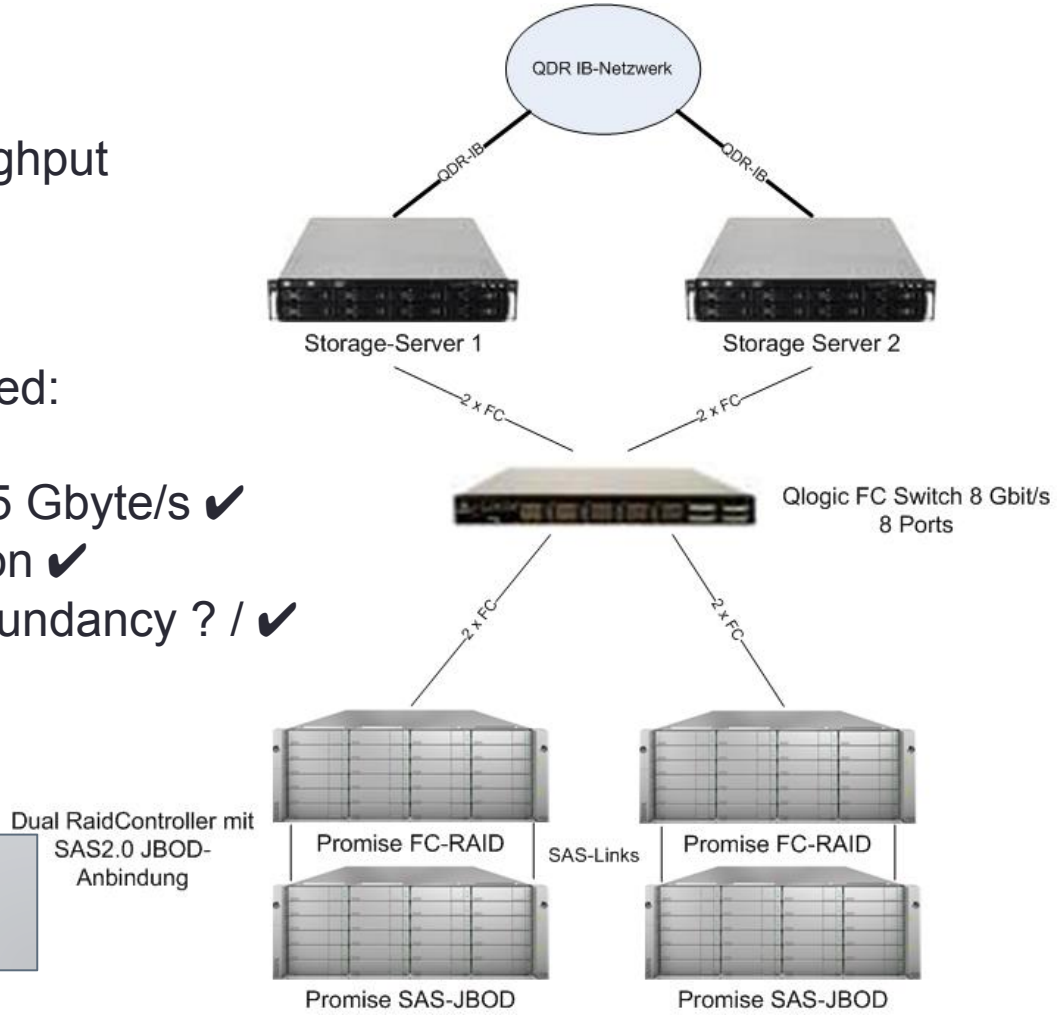
What we wanted:

1. Roughly 1 Gbyte/sec throughput
2. Easy setup / maintenance
3. Stable System

FraunhoferFS / Vendor promised:

1. Throughput in excess of 1.5 Gbyte/s ✔
2. Easy RPM based Installation ✔
3. High Availability / High Redundancy ? / ✔

Benefits of a parallel FS, while hiding comlpexity.



Storage - Basic Setup

- 2 Servers each running 1 instance of:
  - Object Storage Server (4 RAID6 sets, XFS)
  - Meta Data Server (RAID10, ext4)
- Server:
  - Dual Socket Intel E5630 @ 2.53GHz (Quadcore)
  - 12 Gbyte RAM (-> Upgrade to 24 Gbyte)
  - Intel SSD 40 GByte
  - Infiniband QDR (40 Gbit/s), Dual Port Fibre Channel (8 Gbit/s per port)
- Clients connect via Kernel module (no costly context switches)
- Native Infiniband Support (RDMA)

Storage - Performance (Single Client)

- Single Client Write Performance

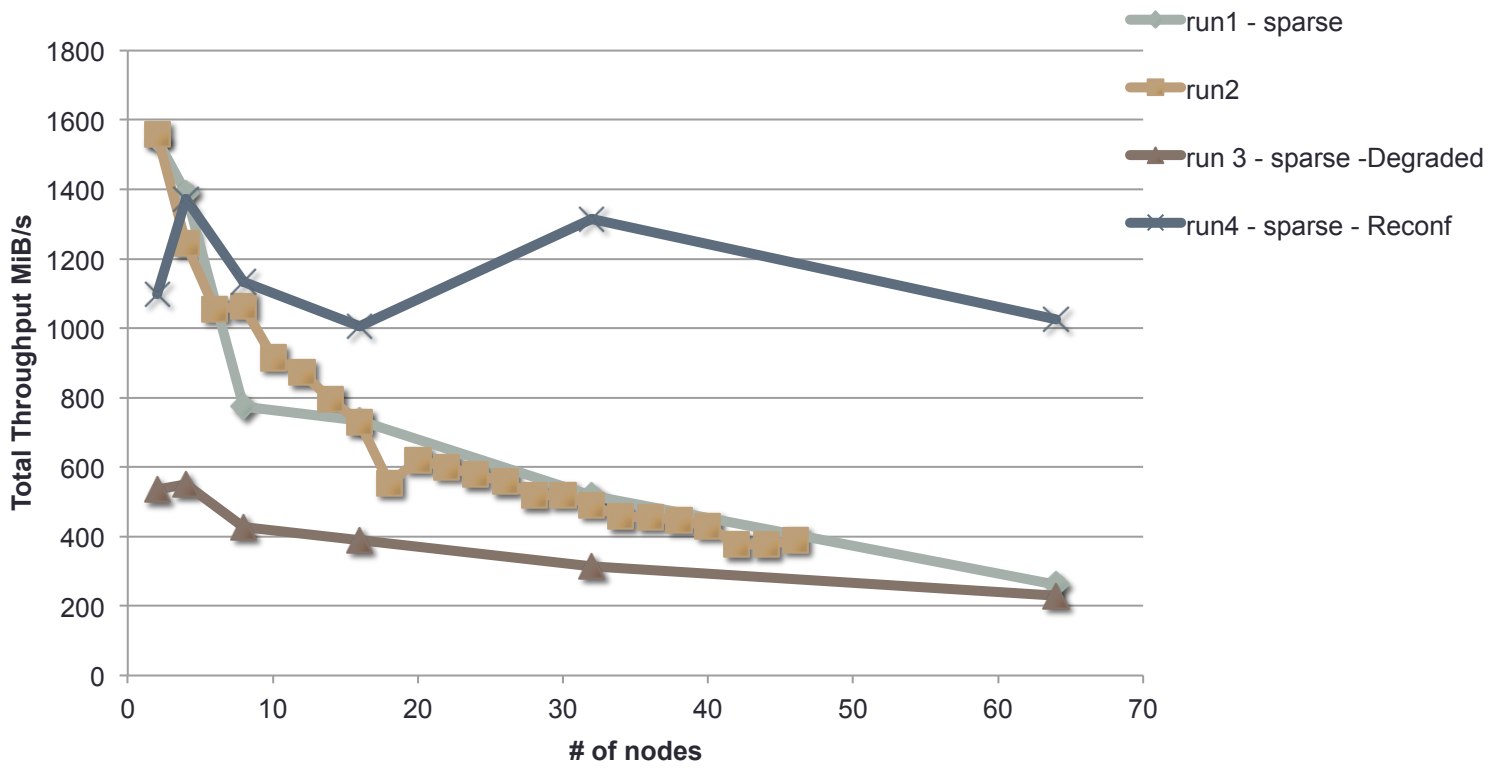
```
[dalibor@node130 ~]$ dd if=/dev/zero of=/data/work/kphth/dalibor/test.file
bs=1M count=100000
100000+0 records in
100000+0 records out
104857600000 bytes (105 GB) copied, 153.017 s, 685 MB/s
```

- Single Client Read Performance

```
[dalibor@node130 ~]$ dd if=/data/work/kphth/dalibor/test.file of=/dev/null
bs=1M
100000+0 records in
100000+0 records out
104857600000 bytes (105 GB) copied, 98.823 s, 1.1 GB/s
```

Storage - Performance (Multi Client Write)

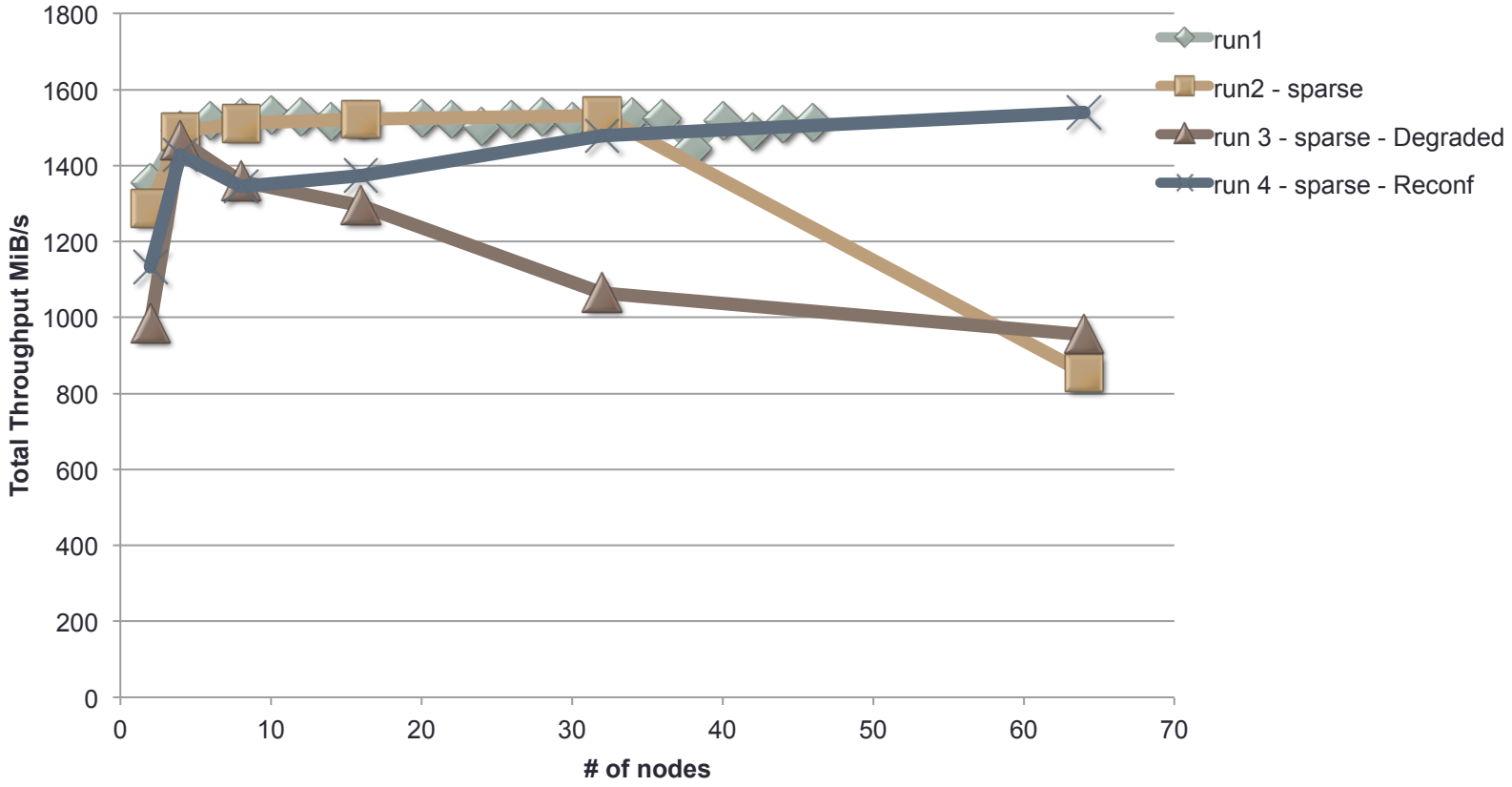
Write Average - 3 runs - Aggr. 97.66 GiB



./IOR -a POSIX -b (100 GByte) / (# of nodes) -o /data/tests/testScaleing –i 3 -w -r –t 1m -d 10 -F

Storage - Performance (Multi Client Read)

Read Average - 3 runs - Aggr. 97.66 GiB



./IOR -a POSIX -b (100 GByte) / (# of nodes) -o /data/tests/testScaleing –i 3 -w -r –t 1m -d 10 -F

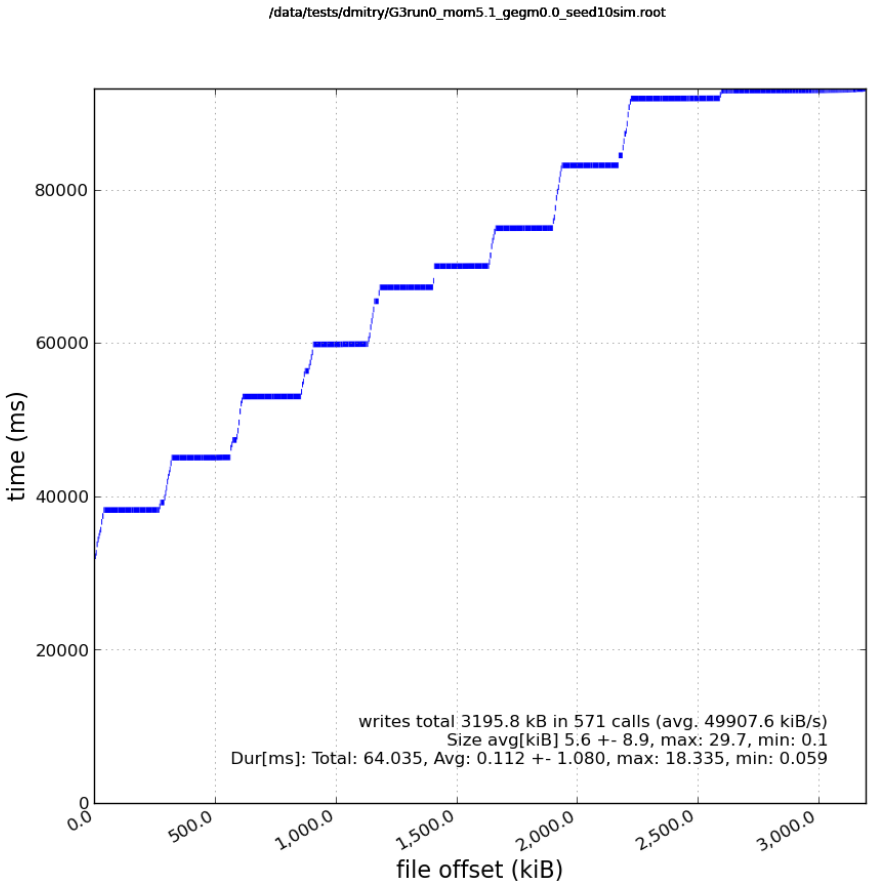
I/O - Profile - Generic Benchmarks

Application I/O Profile:

- Tracing I/O using Linux Standard strace-tool (e.g. ioapps)

```
strace -q -a1 -s0 -f -tttT -oOUT_FILE -e
trace=file,desc,process,socket
APPLICATION ARGUMENTS
```

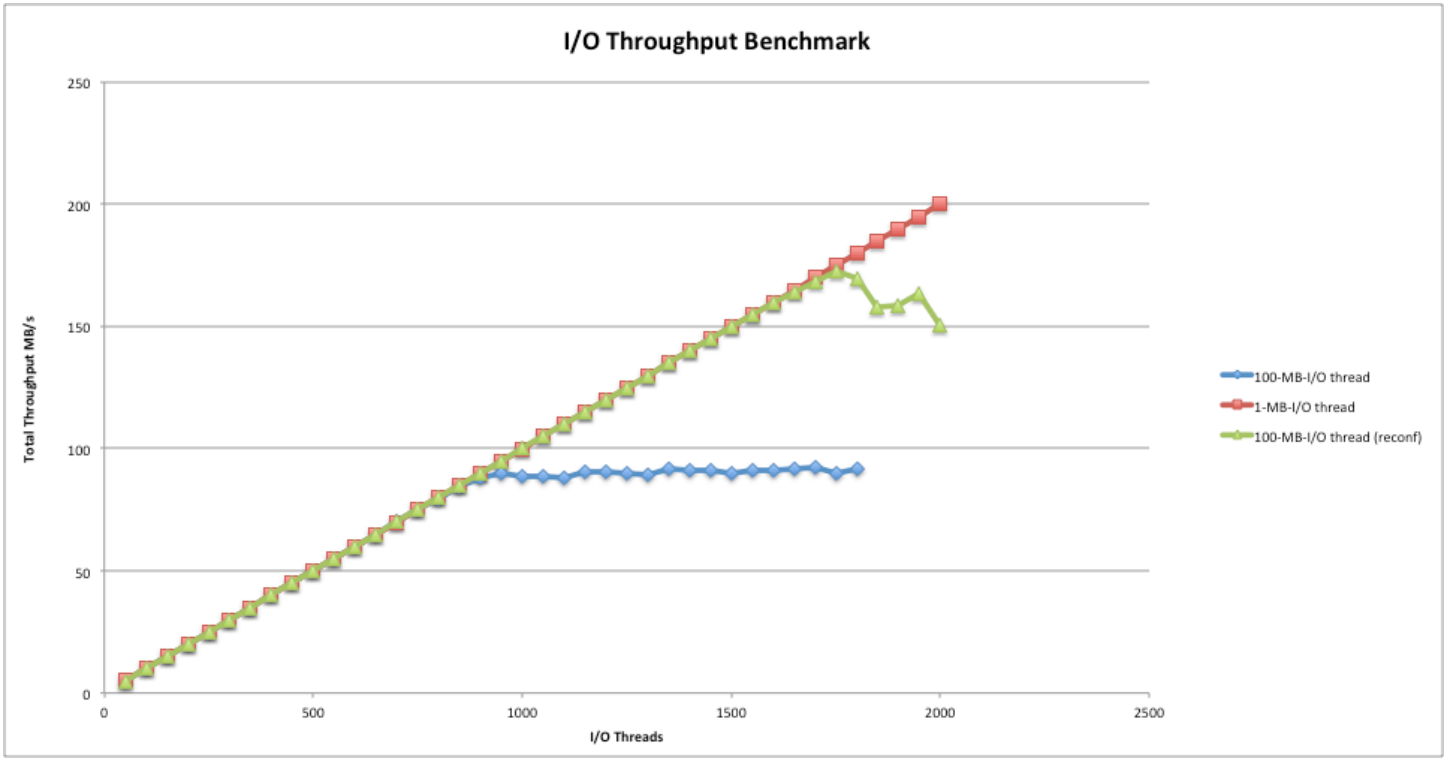
- 200 Events using Dmitry‘s Macro
- Access on file quite irregular
  ⇒ Not easy to find „generic“ Benchmark
- I/O Profiling done with:



- http://code.google.com/p/ioapps/
- https://twiki.cern.ch/twiki/bin/view/Main/CmsIOInstrumenting

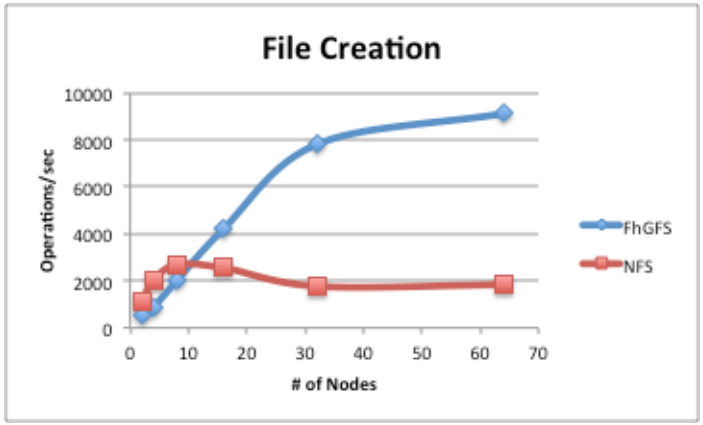
I/O - Simple Scaling Estimate (Write)

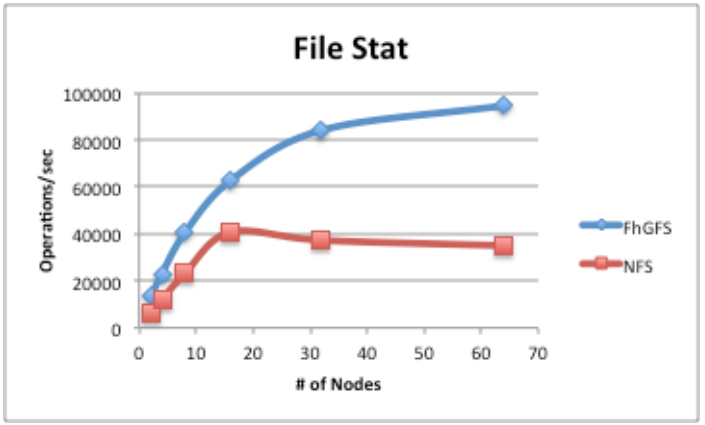
- Simple I/O run:
  - Prog writes 100 kbyte then sleeps for a second
  - Run multiple Threads



Meta Data

- Distributed Meta Data helps






```
./mdtest -n 10 -i 200 -u -t -d /data/work/kphth/dalibor/meta_test/
```

- Sometimes we see problems with ROOT Macros?

```
Process 16615 detached
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 96.23   13.599951          42    325252     26396 stat
  1.37    0.192971      192971         1           wait4
.
.
.
------ ----------- ----------- --------- --------- ----------------
100.00   14.133237                927007     27782 total
```

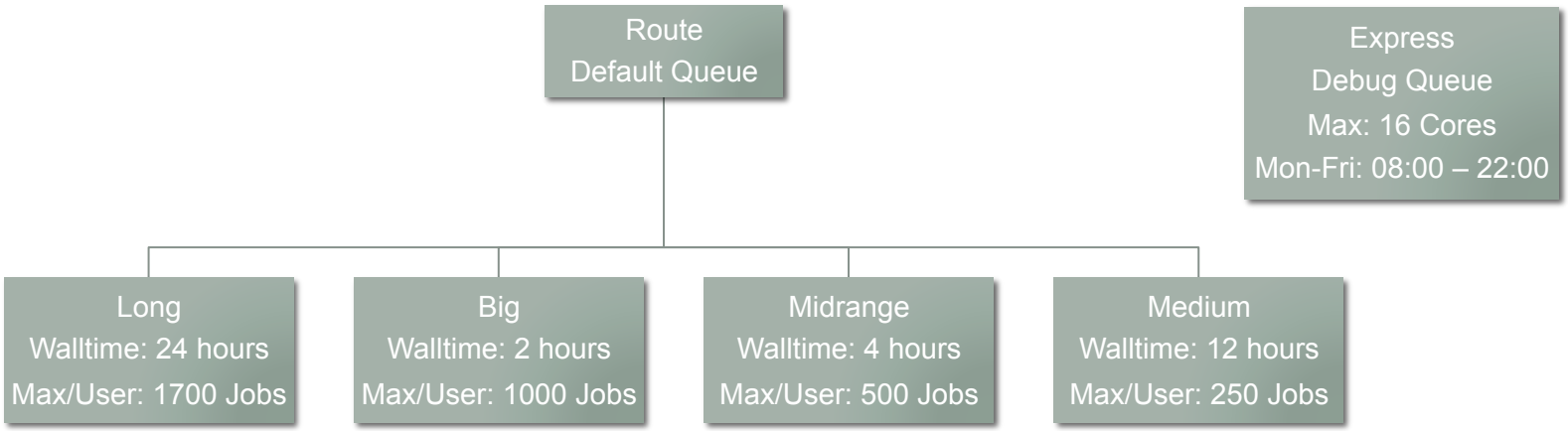
FhGFS update promises to improve stat performance ... Lets see.

Policies / Environment

- User Management (separate clusterwide NIS)
  - Default Resources:
    - 10 Gbyte quota on /home
    - 1 Tbyte scratch on /data (FhGFS)
    - Faishare 20 percent (+ Queue based restrictions)
  - So far account with the Inst. f. Kernphysik mandatory
- Compute nodes may only be used via batch system
- No access from outside (yet)
- PANDAGrid Access (work in progress)
  - Approx. 1000 cores with adjusted priority
  - 5 Tbyte scratch

Batch System

- Combination of Torque/Maui
- Queue assignment depends on walltime of job (routing queues)
- Queue limits depend on max walltime
- Rules revisited regularly to match Cluster usage pattern



Summary - Outlook

- Cluster is in stable production state ☺
- Users seem to be happy so far ☺
- No (too) big issues left ☺
  - Database server max out @ 20.000 queries / sec ☹
  - Metadata might be bottleneck under heavy load? ☹
  - Job output to stdout quite large (several Mbyte / job => several Gbytes)
- Integrate Cluster with Grid (on the way)

Questions?