HIMSTER II

Outline

1. Machine Specs and Basic Performance Numbers
2. Details Scheduler
3. Software
4. Questions

Machine is maintained by HPC department of JGU

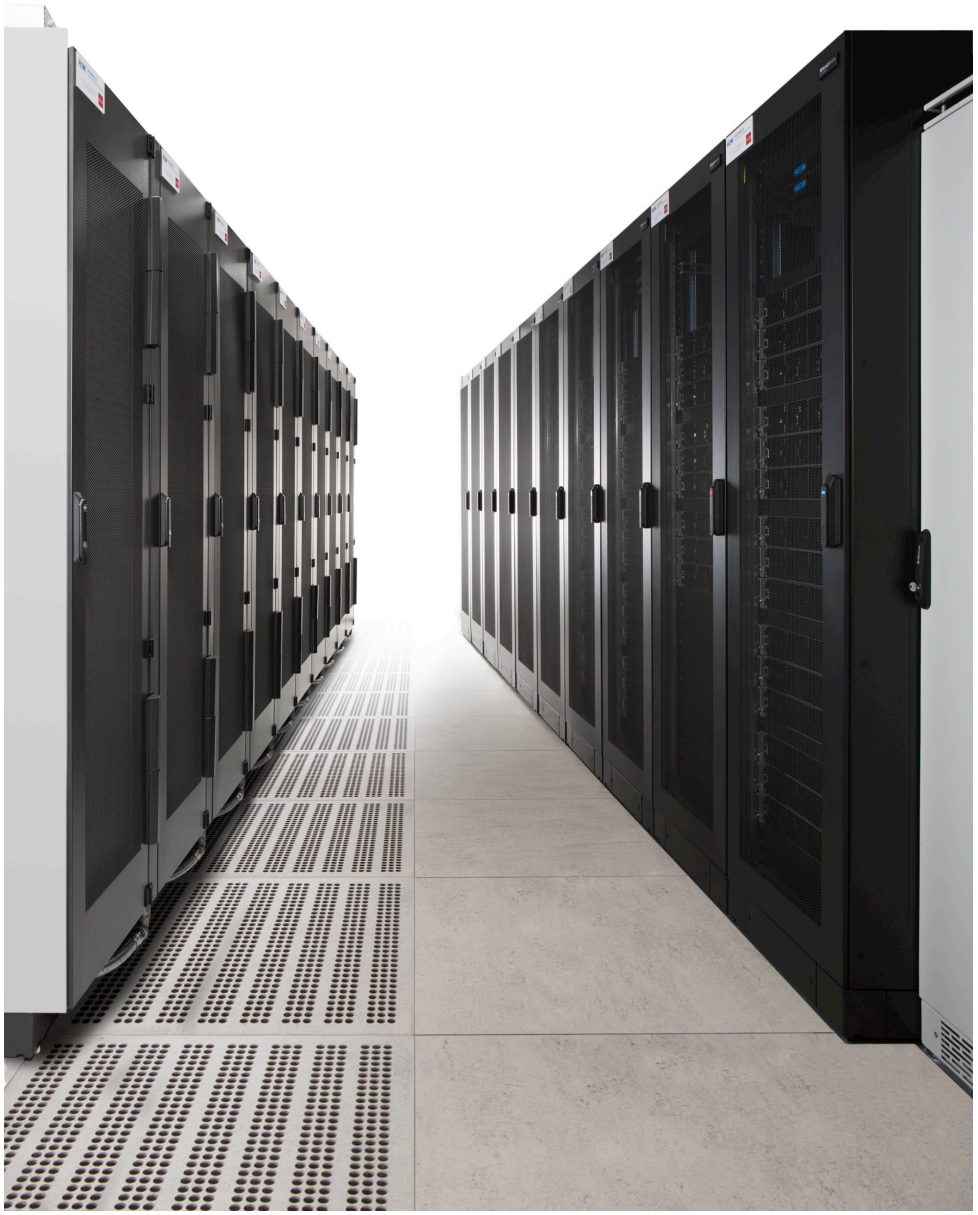
General Info:

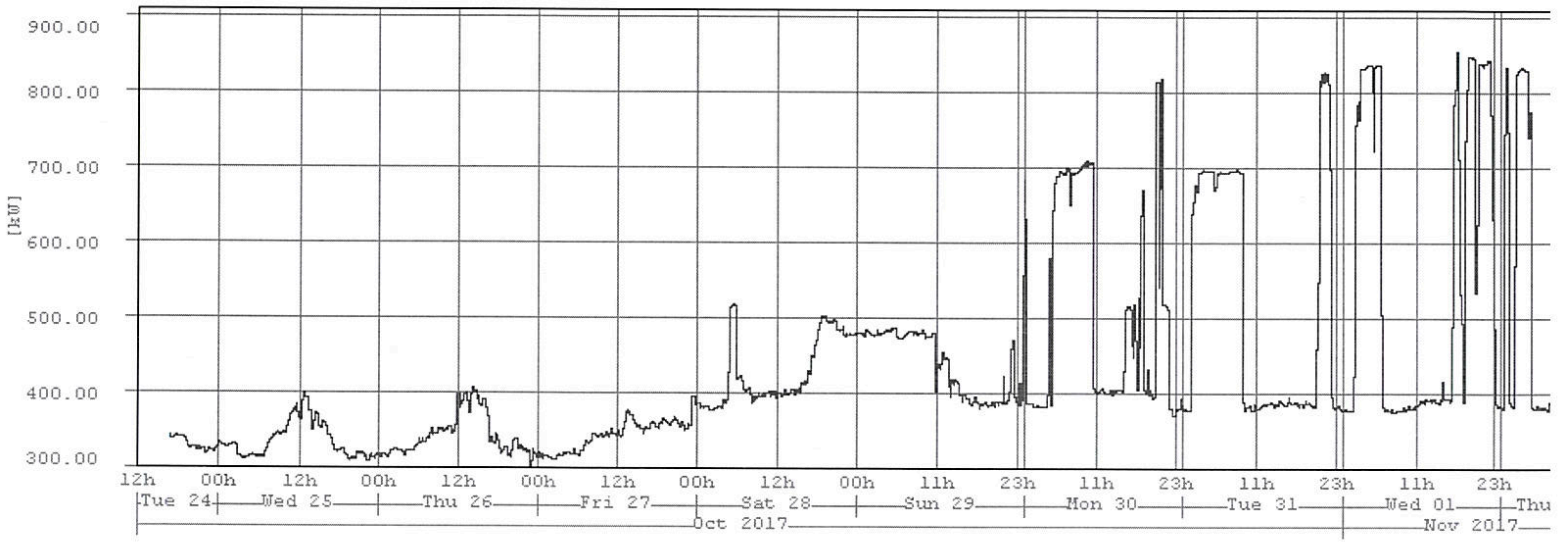
https://mogonwiki.zdv.uni-mainz.de/dokuwiki/

MACHINE SPECS

MogonIIa/b/HIMsterII

- Total cost of 10.6 M€ (HIM 1.9 M€)
- Two Vendors NEC/Megware
- MogonIIa Broadwell CPU MogonIIb/HIMsterII Skylake
- Roughly 2 PFlops Linpack
- Rank 65 Top 500 Fastest Computers (HimsterII makes up for 20 % of that)
- Total of 37 Racks
- Roughly 660 kW Power Consumption (Basically the whole buildings electrical power consumption)
- Installed directly below us





Wirkleistung Summe L1..L3 (10m) [HIM 1395 J04]

HIMster II Machine Specs

- 320 Compute Nodes (216 theory / 100 experiment + 4 development) = 6 Racks
  - Dual Socket Intel 6130 @ 2.1 GHz 32 real cores
  - 3 Gbyte RAM / Core, i.e. 96 Gbyte per node (roughly 1 GB less)
  - Omnipath Adapter 100 Gbit/sec (Currently Limited to 58 Gbit/sec)
  - Gbit ethernet
- Storage
  - 2 Lustre v 2.10 volumes
    - /lustre/miifs04 (Theory)
    - /lustre/miifs05 (Experiment, e.g. in principle more metadata performance)
  - No quotas
  - Usable size 747T each
- Software organized in modules (only per nfs mount so far, no cvmfs)
- Batch System: SLURM
- Part of TOP 500 run: Rank 65 worldwide

Access to the machine

- Access to HIMsterII is granted through a PI (principle investigator) of the HIM
- Every PI will be able to manage users themselves (or relay such a request to it@him.uni-mainz.de)
- JGU user is mandatory (also for external users)
- From inside the JGU
  - ssh miil01-miil04 (only ssh-key login possible)
- Same home directory as on MOGON
  - Quota 300 GByte
  - Mogon allows for login via JGU password (put your ssh key there)
- More info on logging in from the outside can be found here: https://mogonwiki.zdv.uni-mainz.de/dokuwiki/ssh_from_outside
- You have to follow the regualtions here: https://www.zdv.uni-mainz.de/benutzungsordnung/ https://www.en-zdv.uni-mainz.de/regulations-for-use-of-the-data-center/

SOME PERFORMANCE NUMBERS

Machine Performance - Memory

- On HIMsterII hyperthreading is enabled by default
- Processplacement is important
  - export KMP_AFFINITY=scatter; export OMP_NUM_THREADS=32
    - This gets you to roughly 180 Gbytes/sec
  - export KMP_AFFINITY=scatter; export OMP_NUM_THREADS=64
    - This gets you roughly 182 Gbytes/sec
  - Wihtout affinity setting maximum is 98 Gbytes/sec (i.e. 1 socket)
- Stream perf numbers using

icc -axCORE-AVX512,CORE-AVX2,AVX,SSE4.2 -march=native -O3 stream.c –fopenmp

STREAM

```
[djukanov@x2258 originalstream]$ ./a.out
-------------------------------------------------------------
STREAM version $Revision: 5.10 $
-------------------------------------------------------------
This system uses 8 bytes per array element.
-------------------------------------------------------------
Array size = 100000000 (elements), Offset = 0 (elements)
Memory per array = 762.9 MiB (= 0.7 GiB).
Total memory required = 2288.8 MiB (= 2.2 GiB).
Each kernel will be executed 10 times.
 The *best* time for each kernel (excluding the first iteration)
 will be used to compute the reported bandwidth.
-------------------------------------------------------------
Number of Threads requested = 32
Number of Threads counted = 32
-------------------------------------------------------------
Your clock granularity/precision appears to be 1 microseconds.
Each test below will take on the order of 8846 microseconds.
   (= 8846 clock ticks)
Increase the size of the arrays if this shows that
you are not getting at least 20 clock ticks per test.
-------------------------------------------------------------
WARNING -- The above is only a rough guideline.
For best results, please be sure you know the
precision of your system timer.
-------------------------------------------------------------
Function    Best Rate MB/s  Avg time     Min time     Max time
Copy:           173969.8     0.009267     0.009197     0.009381
Scale:          174785.4     0.009190     0.009154     0.009239
Add:            179948.7     0.013368     0.013337     0.013389
Triad:          179023.8     0.013431     0.013406     0.013462
-------------------------------------------------------------
Solution Validates: avg error less than 1.000000e-13 on all three
arrays
-------------------------------------------------------------
```

Triad Using Handwritten Assembler AVX512

[djukanov@x2258 mystream] $ ./test

Number of loops= 100

102402048 Elements

Alignment is 0x62800000 0x31a00000 0xc00000

Total bytes use is = 2457.65 [Mbytes]

TSC frequency is 2.10 GHz

Running 32 threads

Triad test using AVX512 intrinsics

==========

Avg : 180.51 [Gbytes/sec]

Best: 184.24 [Gbytes/sec]

```
#define load_scalar(c) \
__asm__ __volatile__ ("vbroadcastsd %0, %
%zmm2\n"\
                      :\
                      :\
                      "m"(c)\
                      :\
                      "xmm2");

#define triad(a, b, c) \
__asm__ __volatile__( "vmovapd %2, %
%zmm1\n"\
                      "vfmadd213pd %1, %%zmm2, %
%zmm1\n"\
                      "vmovntpd %%zmm1, %0\n"\
                       :\
                       "=m" (*(a))\
                       :\
                       "m"(*(b)),\
                       "m"(*(c)),\
                       "m"(d)\
                       :\
                       "xmm0","xmm1","xmm2")
```

Triad Using Handwritten Assembler AVX

[djukanov@x2258 mystream]$ ./test
Number of loops= 100
102402048 Elements
Alignment is 0x62800000 0x31a00000 0xc00000
Total bytes use is = 2457.65 [Mbytes]
TSC frequency is 2.10 GHz
Running 32 threads
Triad test using AVX intrinsics
==========
Avg : 173.06 [Gbytes/sec]
Best: 177.16 [Gbytes/sec]

Triad Using Handwritten Assembler

[dalibor@node001 mystream]$ ./test
Number of loops= 100
102402048 Elements
Alignment is 0x62800000 0x31a00000 0xc00000
Total bytes use is = 2457.65 [Mbytes]
TSC frequency is 2.60 GHz
Running 16 threads
Triad test using AVX intrinsics
==========

Avg : 88.77 [Gbytes/sec]
Best: 89.67 [Gbytes/sec]

Memory

- Per core mem bandwith roughly same as on Clover (5.6 Gbyte/sec)
- Per core mem bandwidth 1.7x increase over HIMster (3.8 Gbyte/sec)
- If your application is membandwidth limited transition
  - From HIMster to HIMsterII will help you a lot
  - Form Clover to HIMsterII not so much
- In any case more mem per core
- If you run OpenMP jobs be aware of the CPU affinity settings
  OMP_PROC_BIND=spread
  KMP_AFFINITY=scatter

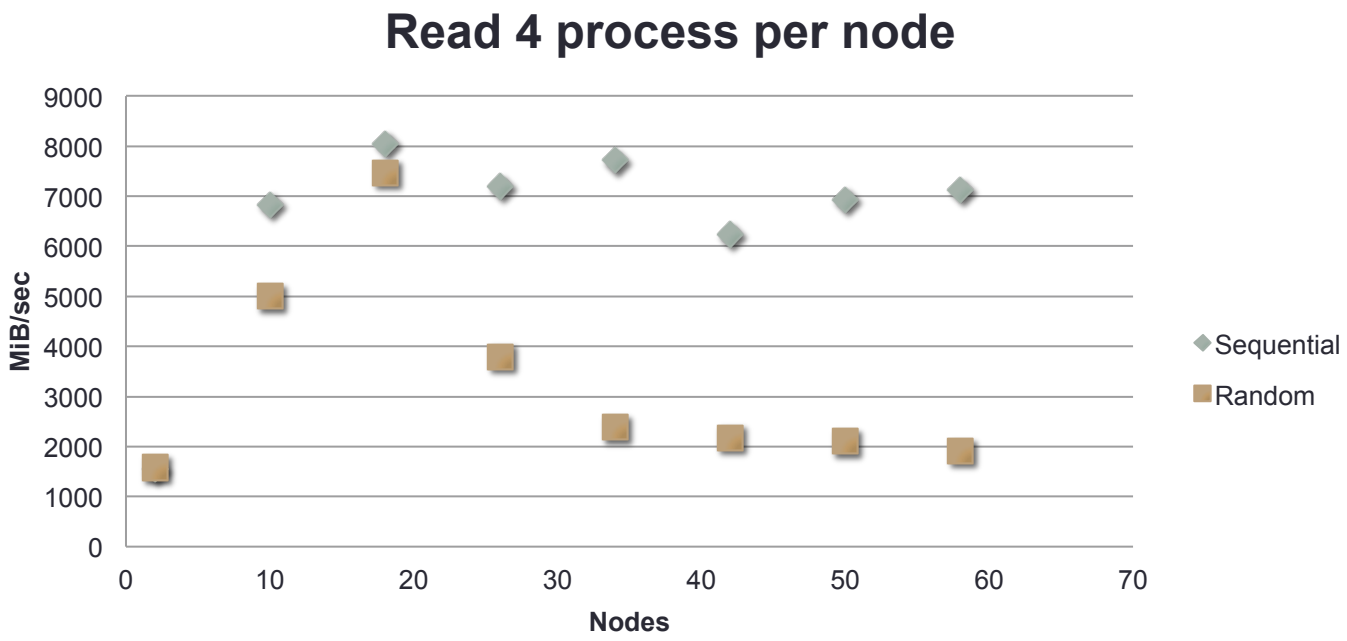
CPU

- HIMsterII have Skylake CPUs, i.e. AVX512 instruction set is available.
- However frequencies for the VU change
  - 2.1 (2.8) GHz scalar
  - 1.6 (2.4) GHz AVX2 (256 Bit Register)
  - 1.3 (1.9) GHz AVX512 (512 Bit Register)
  - Number in parenthesis is max turbo frequency when fully utilized
- Not so (super)clear going (full) vector helps all too much

Storage

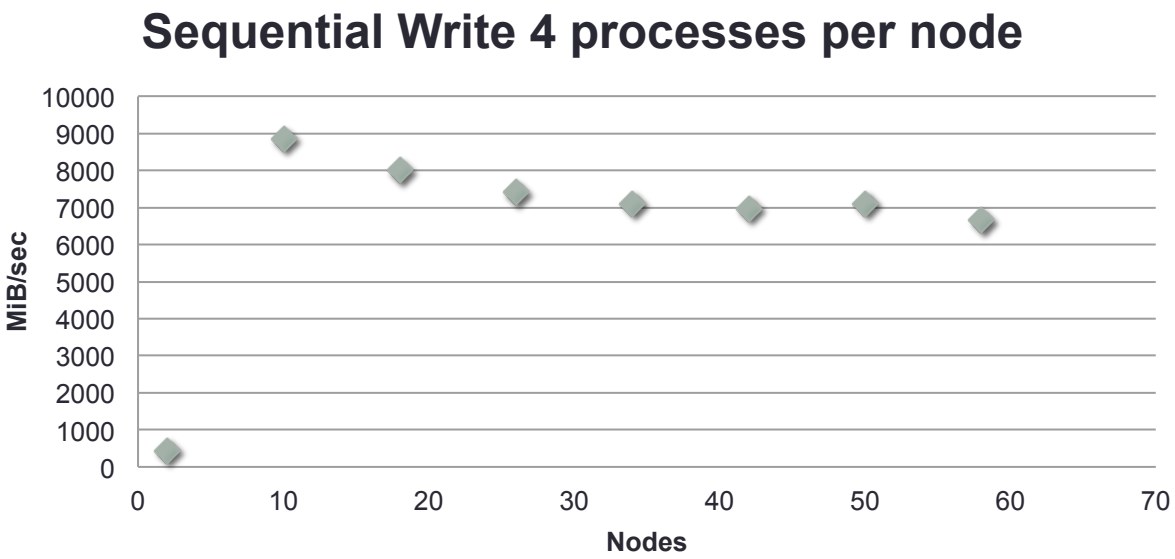
- We use Lustre, with some performance numbers on the next slide
- NO BACKUP of data
- Dedicated storage systems for experiment and theory
  - /lustre/miifs04 (Theory)
  - /lustre/miifs05 (Experiment, e.g. in principle more metadata performance)
- Try to use large files:
  Source code should be in /home/
- Try not to put too many files into one directory (less than 1k)
- Try to avoid too much metadata
  DO NOT DO ls –l unless you really need it
- In your scripts avoid excessive tests of file existence (put in a sleep statement betwenn two tests say 30 secs)
- Use lfs find rather than GNU find
- Use O_RDONLY | O_NOATIME

Performance using IOR

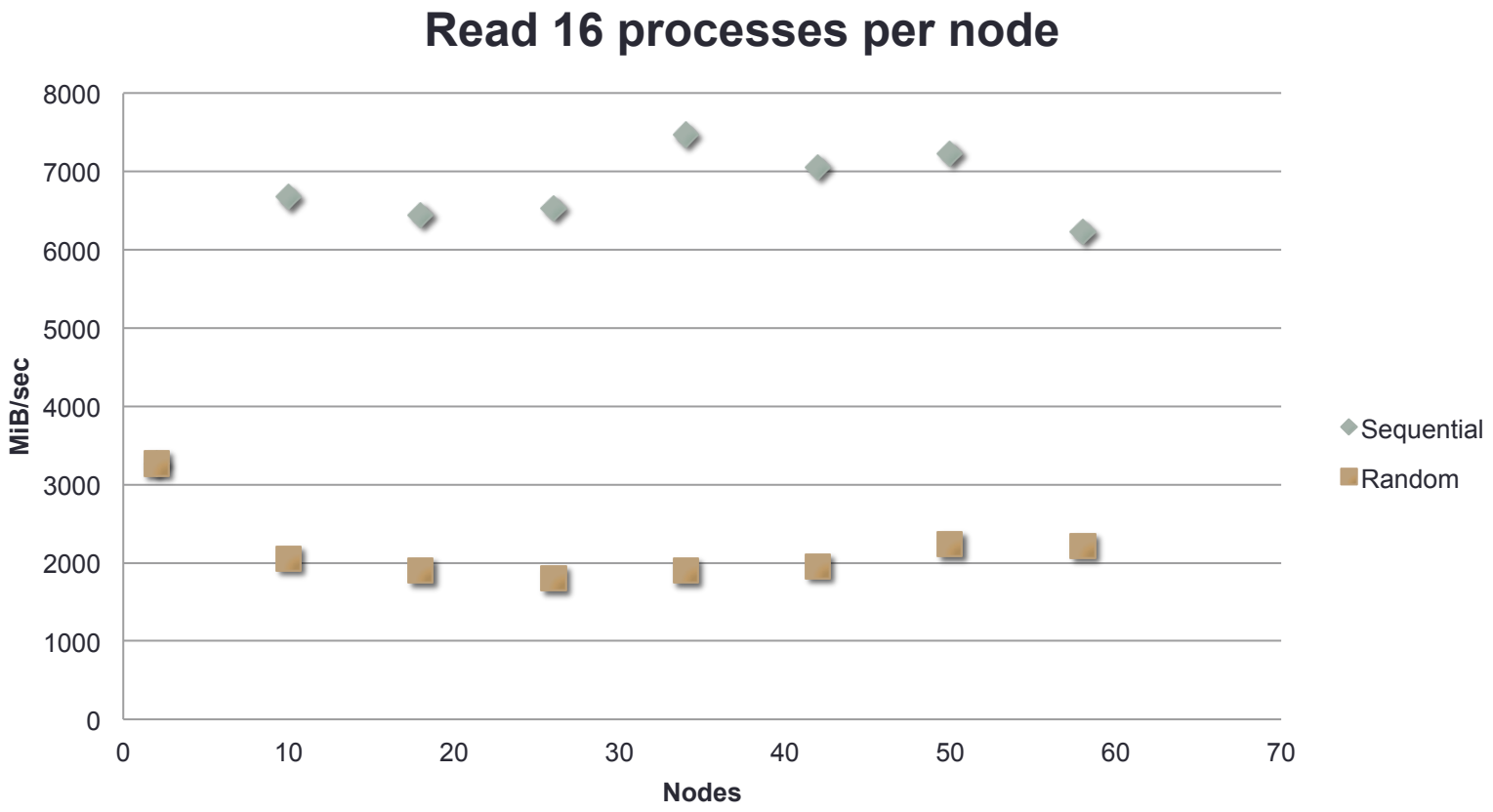


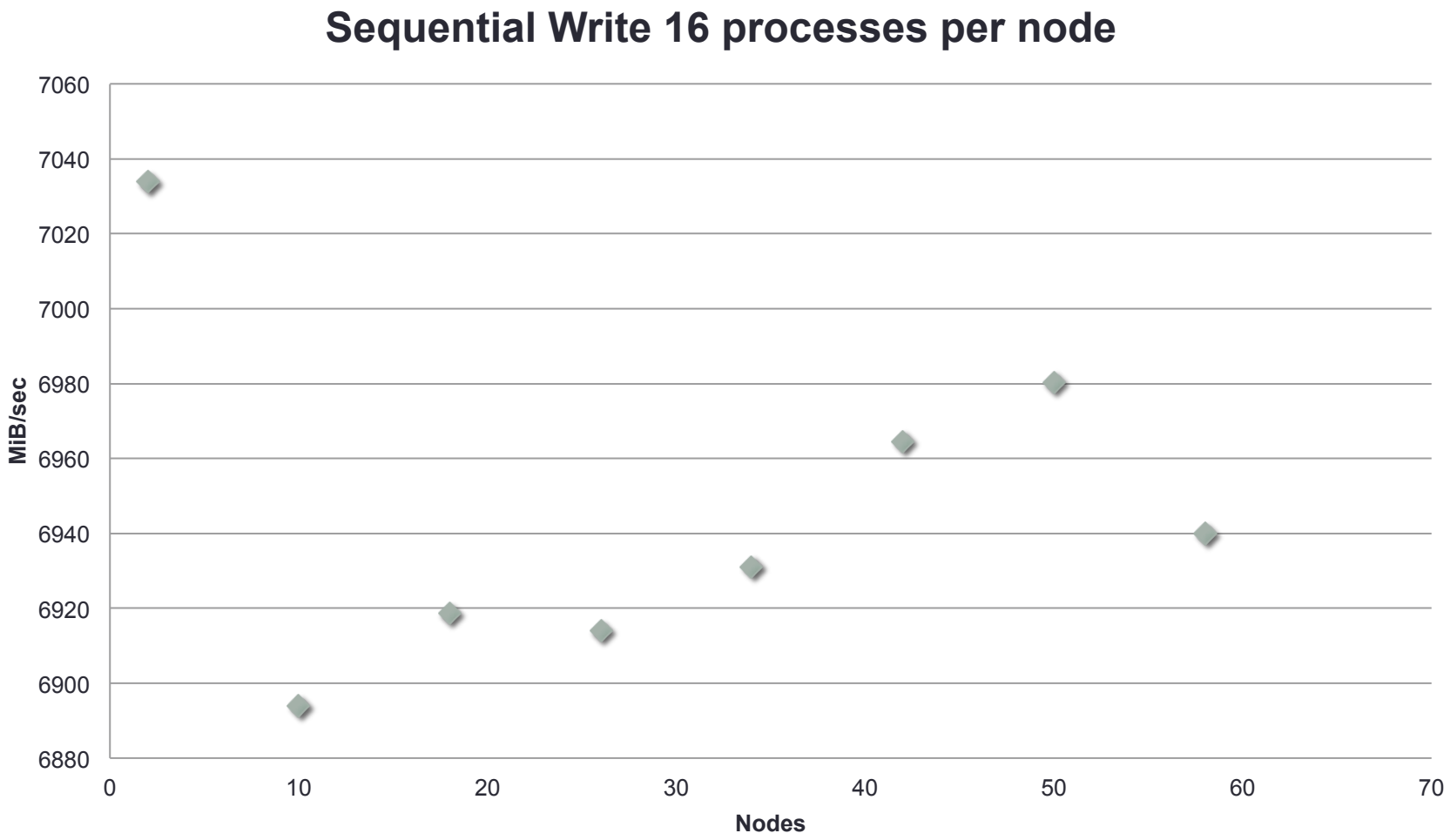
Every process write a 1 Gbyte file (1 MB block size)

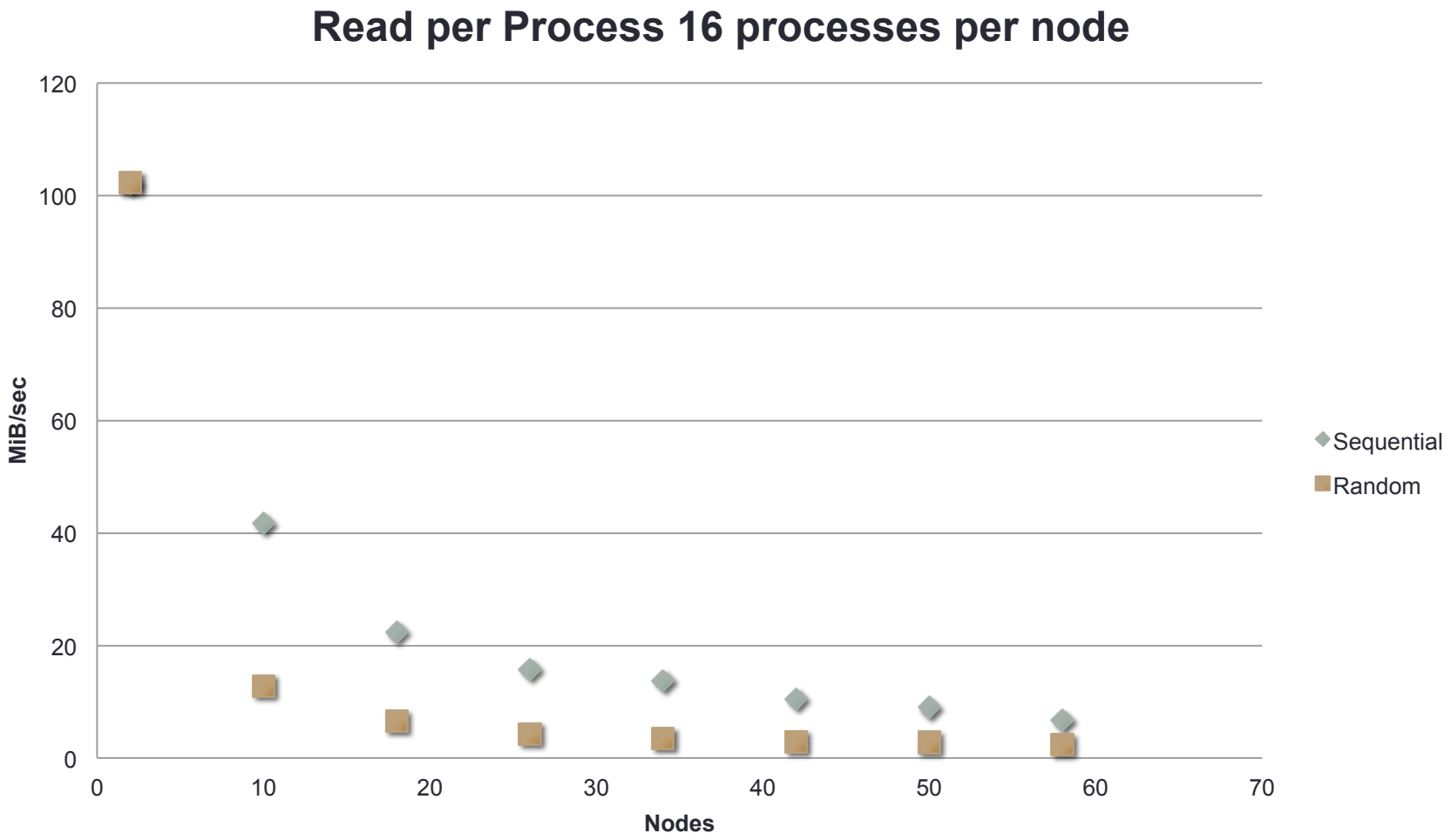
Performance using IOR



Every process write a 1 Gbyte file (1 MB block size)


Read 16 processes per node
8000
7000
6000
5000
4000
3000
2000
1000
0
MiB/sec
0
10
20
30
40
50
60
70
Nodes
Sequential
Random


Sequential Write 16 processes per node
MiB/sec
7060
7040
7020
7000
6980
6960
6940
6920
6900
6880
0
10
20
30
40
50
60
70
Nodes


Read per Process 16 processes per node
MiB/sec
120
100
80
60
40
20
0
0
10
20
30
40
50
60
70
Nodes
Sequential
Random

SCHEDULER

Scheduler Slurm

- Running using slurm you have to give an account. Accounts associated with you can be found

  sacctmgr -s list user $USER

- For the HIMster II basically there are 2 accounts:
  - m2_him_exp
  - m2_him_th
- You submit to a partition (sort of name of the machine)
  - himster2_exp
  - himster2_th
  - himster2_devel (this one is shared)
- Submission is similar to HIMster/Clover

Slurm – Sample Submission Script

- #!/bin/bash
- #SBATCH -o /home/djukanov/latency/myjob.%j.%N.out
- #SBATCH -D /home/djukanov/latency
- #SBATCH -J bench_baseline
- #SBATCH -A m2_him_th
- #SBATCH –partition=himster2_th
- #SBATCH -N 2
- #SBATCH --mem-per-cpu=2400
- #SBATCH --mail-type=end
- #SBATCH --mail-user=djukanov@uni-mainz.de
- #SBATCH --time=01:00:00
- #SBATCH --exclude=x0328,x0326,x0327

- module load mpi/impi/2018.1.163-iccifort-2018.1.163-GCC-6.3.0
- srun -n 2 --tasks-per-node=1 ./lat

Scheduler Slurm Priority

Siehe hier:
https://slurm.schedmd.com/priority_multifactor.html#mfjppintro

Job_priority = (PriorityWeightAge) * (age_factor) + (PriorityWeightFairshare) * (fair-share_factor) + (PriorityWeightJobSize) * (job_size_factor) + (PriorityWeightPartition) * (partition_factor) + (PriorityWeightQOS) * (QOS_factor) + SUM(TRES_weight_cpu * TRES_factor_cpu, TRES_weight_<type> * TRES_factor_<type>, ...)

```
[djukanov@login22 src]$ sprio  -w
          JOBID   PRIORITY
  AGE  FAIRSHARE    JOBSIZE        QOS
    2000    1000000      40000      10000
```

Slurm Priority

- Age:
  Jobage with respect to MaxAge (7 days)
- Jobsize:
  Zur Zeit im Status Small_Relative_To_Time, Berechnung:
  ((#CPUS für Job)/(Walltime in Minuten))/(Gesamtzahl CPUS)
  -> BigJob with a one minute Walltime Factor 1
- Fairshare
  Currenlty in Modus FAIR_TREE
  FAIR_TREE If set, priority will be calculated in such a way that if accounts A and B are siblings and A has a higher fairshare factor than B, all children of A will have higher fairshare factors than all children of B.
  PriorityDecayHalfLife = 10 Days
  FairShareTarget=?
- Tres not used currently

Slurm

- It is intended that your account usage on HimsterII does not count against MogonIIa or MogonIIb, i.e.
  - Restrictions on CPU hours should not apply
  - Walltime will not be charged against other accounts
  - Fairshare „should be machine dependent“

SOFTWARE

Software – Module

Module avail will show you the available sofwtare:
[djukanov@login22 mystream]$ module avail

```
------------------------------------------------------------ /usr/share/Modules/modulefiles -------------------------------------------------------------
dot         module-git  module-info modules     null        use.own

------------------------------------------------------------ /cluster/easybuild/broadwell/modules/all ------------------------------------------------------------
bio/FastTree/2.1.10-foss-2017a                                  math/libxsmm/1.6.4-intel-2017.02
bio/IQ-TREE/1.5.6-foss-2017a-omp-mpi                            math/Maple/17
bio/MAFFT/7.305-foss-2017a-with-extensions                      math/MATLAB/2017a
bio/MIGRATE-N/3.6.11-foss-2017a                                 math/MPFR/3.1.6-GCCcore-6.3.0
bio/PhyloBayes-MPI/1.7b-intel-2018.01                           mpi/impi/2017.2.174-iccifort-2017.2.174-GCC-6.3.0
bio/PhyloBayes-MPI/20161021-intel-2018.01                       mpi/impi/2018.0.128-iccifort-2018.0.128-GCC-6.3.0
bio/RAxML/8.2.11-foss-2017a-hybrid-avx2                         mpi/impi/2018.1.163-iccifort-2018.1.163-GCC-6.3.0
chem/CP2K/2.5.1-intel-2017.02                                   mpi/MVAPICH2/2.2-GCC-6.3.0
chem/CP2K/4.1-intel-2017.02                                     mpi/OpenMPI/1.10.4-GCC-6.3.0
chem/Libint/1.1.4-intel-2017.02                                 mpi/OpenMPI/2.0.2-GCC-6.3.0
chem/libxc/2.2.3-intel-2017.02                                  mpi/OpenMPI/2.1.2-GCC-6.3.0
chem/ORCA/4.0.1-OpenMPI-2.0.2                                   mpi/OpenMPI/3.0.0-GCC-6.3.0
chem/PLUMED/2.3.0-intel-2017.02                                 numlib/FFTW/3.3.6-gmvapich2-2017a
compiler/GCC/6.3.0                                              numlib/FFTW/3.3.6-gompi-2017a
compiler/GCC/7.2.0                                              numlib/FFTW/3.3.7
compiler/GCCcore/5.4.0                                          numlib/GSL/2.3-intel-2017.02
compiler/GCCcore/6.3.0                                          numlib/imkl/2017.2.174-iimpi-2017.02-GCC-6.3.0
compiler/GCCcore/7.1.0                                          numlib/imkl/2018.0.128-iimpi-2018.00-GCC-6.3.0
compiler/GCCcore/7.2.0                                          numlib/imkl/2018.1.163-iimpi-2018.01-GCC-6.3.0
compiler/icc/2017.2.174-GCC-6.3.0                               numlib/OpenBLAS/0.2.19-GCC-6.3.0-LAPACK-3.7.0
compiler/icc/2018.0.128-GCC-6.3.0                               numlib/PETSc/3.7.6-intel-2017.02-downloaded-deps
compiler/icc/2018.1.163-GCC-6.3.0                               numlib/ScaLAPACK/2.0.2-gmvapich2-2017a-OpenBLAS-0.2.19-LAPACK-3.7.0
compiler/ifort/2017.2.174-GCC-6.3.0                             numlib/ScaLAPACK/2.0.2-gompi-2017a-OpenBLAS-0.2.19-LAPACK-3.7.0
compiler/ifort/2018.0.128-GCC-6.3.0                             perf/Advisor/2018.1
compiler/ifort/2018.1.163-GCC-6.3.0                             perf/IMB/4.1-intel-2017.02
data/grib_api/1.21.0-foss-2017a                                 perf/OSU-Micro-Benchmarks/5.3.2-foss-2017a
data/grib_api/1.21.0-intel-2017.02                              system/CUDA/8.0.61
data/HDF5/1.8.18-foss-2017a                                     toolchain/foss/2017a
data/HDF5/1.8.18-intel-2017.02                                  toolchain/gmvapich2/2017a
data/netCDF/4.4.1.1-foss-2017a-HDF5-1.8.18                      toolchain/gmvolf/2017a
data/netCDF/4.4.1.1-intel-2017.02-HDF5-1.8.18                   toolchain/gompi/2017a
data/netCDF-C++4/4.3.0-intel-2017.02-HDF5-1.8.18                toolchain/iccifort/2017.2.174-GCC-6.3.0
data/netCDF-Fortran/4.4.4-foss-2017a-HDF5-1.8.18                toolchain/iccifort/2018.0.128-GCC-6.3.0
data/netCDF-Fortran/4.4.4-intel-2017.02-HDF5-1.8.18             toolchain/iccifort/2018.1.163-GCC-6.3.0
debugger/Valgrind/3.13.0-foss-2017a                             toolchain/iimpi/2017.02-GCC-6.3.0
devel/Bazel/0.5.4                                               toolchain/iimpi/2018.00-GCC-6.3.0
devel/Boost/1.60.0-intel-2018.01                                toolchain/iimpi/2018.01-GCC-6.3.0
devel/Boost/1.63.0-intel-2017.02-Python-2.7.13                  toolchain/intel/2017.02
devel/Boost/1.63.0-intel-2018.01-Python-2.7.14                  toolchain/intel/2018.00
devel/Boost/1.65.1-foss-2017a                                   toolchain/intel/2018.01
devel/Boost/1.65.1-foss-2017a-Python-2.7.13                     tools/Advisor/2018.1
devel/Boost/1.65.1-intel-2017.02                                tools/APR/1.6.2
devel/CMake/3.7.2                                               tools/APR-util/1.6.0
devel/likwid/4.3.1-foss-2017a                                   tools/EasyBuild/3.4.0
devel/MariaDB/5.5.52-clientonly                                 tools/EasyBuild/3.5.0
devel/protobuf/3.3.0-intel-2017.02                              tools/GC3Pie/2.4.2
devel/protobuf-python/3.3.0-intel-2017.02-Python-3.6.1          tools/IntelClusterChecker/2018.0.002
devel/SCons/2.5.1                                               tools/IntelClusterChecker/2018.1
EasyBuild/3.1.1                                                 tools/itac/2018.1.017
EasyBuild/3.2.0                                                 tools/parallel/20170622
```

Software

- Usually you'd need to compule certain things
  module load compiler/GCCcore/6.3.0
- Or say for mpi

  module load mpi/OpenMPI/3.0

which mpicc

/cluster/easybuild/broadwell/software/mpi/OpenMPI/3.0.0-GCC-6.3.0/bin/mpicc.0-GCC-6.3.0

Make it easy to change

module load mpi/impi/2018.1.163-iccifort-2018.1.163-GCC-6.3.0

which mpicc /cluster/easybuild/broadwell/software/mpi/impi/2018.1.163-iccifort-2018.1.163-GCC-6.3.0/bin64/mpicc

Software – HIM specific

- Currently software goes to

[djukanov@x2258 mystream]$ ls /cluster/him/

bes3 fairroot fairsoft modulefiles

- This is an nfs mount
- NO CVMFS so far
- We will provide additional modules
- What is needed? Send a request to it@him.uni-mainz.de

DATAMIGRATION

HIMster-Shutdown Data Migration

- HIMster is running on unreliable storage hardware
- Need a process to migrate data until the end of June (will be 8th year of running)
- Proposal:
  - Every group assigns a data officer who will be responsible to sort out which data to migrate and what can die
  - Will report a list to us with directory name to copy and discuss the best strategy to either move the data to the new storage (or possibly archive on tape)