Outline

- SuperMUC supercomputer
- User Projects
- Monitoring Tool
- Darshan Tool
- Persyst and Darshan
- Conclusions
SuperMUC supercomputer - LRZ

- Member of the Gauss Centre for Supercomputing (GCS). Tier-0 centre for PRACE, the Partnership for Advanced Computing in Europe.
- 2012 SuperMUC Phase 1 and 2015 SuperMUC Phase 2. Total Peak Performance 6.4 PFlop/s.
SuperMUC supercomputer - LRZ

SuperMUC supercomputer - LRZ

Phase 2 Compute Nodes

I/O System

Phase 1 Compute Nodes
The I/O PATH on SuperMUC - Parallel Storage (WORK and SCRATCH file space)

10 x (8NSD + 1 DDN)
1 DDN (1 SFA12K):
- 564 SATA drives (HDD size is 3TB, NL SATA)
- 56 RAID6 (LUNS 8+2P)
- Stripe width is 2MiB
- Stripe size 256kiB
- 4 hot Spare drives

12 PB (WORK)

80 NSD

16 NSD

5 PB (SCRATCH)
The Global Filesystem HOME (NFS)

- Available on all HPC cluster systems (environment variable $HOME)
- Shared area for all user accounts in a project

Very reliable
- user-restorable snapshots (last 10 days)
- automatic data protection by LRZ
Outline

▪ SuperMUC supercomputer
▪ User Projects
▪ Monitoring Tool
▪ Darshan Tool
▪ Persyst and Darshan
▪ Conclusions
Projects by Research Area

- Computational-Fluid-Dynamics (CFD) 26%
- Astrophysics-Cosmology (APH) 6%
- Informatics-ComputerSciences (INF) 10%
- Chemistry (CHE) 13%
- Biophysics-Biology-Bioinformatics (BIO) 5%
- Physics-High-EnergyPhysics (HEP) 17%
- Physics-Solid-State (FKP) 9%
- Geophysics (GEO) 13%
- Engineering-others (ENG) 3%
- Meteorology-Climatology-Oceanography (CLI) 3%
- Other 2%
I/O Requirements

I/O Libraries
- HDF5 15%, NetCDF or PnetCDF 10%; POSIX, MPI-IO, or an I/O library locally installed 75%.

Storage Parallel
- WORK (70% Capacity) -> 5 fold increase
- SCRATCH (80% Capacity) -> 8 fold increase

Checkpointing and large scale output with a connection to a visualization cluster.
Spanning (for the Large-Scale Projects):
Periods: 5 min to 8 hours
Size: 100 GB -> 38%
1TB -> 10%
5TB -> 7%
10TB -> 1%
35TB -> 2%
70TB -> 1%
< 100GB -> 41%
Outline

- SuperMUC supercomputer
- User Projects
- Monitoring Tool
- Darshan Tool
- Persyst and Darshan
- Conclusions
GPFS Monitor on SuperMUC: SCRATCH

One Year (06.06.15 19:01 - 20.06.16 19:01)

Datasource Throughput

![Graph showing GPFS throughput over a year with throughput values for read and write operations.]

- **Read**: 0.00 B/s Last 47.66 GB/s Average 172.43 GB/s Max
- **Write**: 0.00 B/s Last 50.14 GB/s Average 166.23 GB/s Max
One Year (06.06.15 19:00 - 20.06.16 19:00)

Datasource Throughput

- **read**: 0.00 B/s, Last 53.18 GB/s, Average 341.75 GB/s, Max
- **write**: 0.00 B/s, Last 30.75 GB/s, Average 166.48 GB/s, Max
One Year (06.06.15 18:59 - 20.06.16 18:59)
Datasource Throughput

NFS throughput

- read: 0.00 B/s Last 535.09 MB/s Average 2.36 GB/s Max
- write: 0.00 B/s Last 158.11 MB/s Average 1.48 GB/s Max
PerSyst Tool collects performance properties of all running jobs every 10 minutes. No instrumentation is needed nor modifications to the user codes.

https://www.lrz.de/services/compute/supermuc/tuning/persystreport/
Outline

- SuperMUC supercomputer
- User Projects
- Monitoring Tool
- Darshan Tool
- Persyst and Darshan
- Conclusions
To make use of Darshan in its version 2.3 and 3.x, the module appropriate must be loaded.

```
module load darshan
```

- Set up the variable `FORTRAN_PROG` in “true” if the program is a Fortran program and false if it’s not.

```
FORTRAN_PROG=true
```

- Load the appropriate library.

```
export LD_PRELOAD=`darshan-user.sh $FORTRAN_PROG`
```

- Set up Darshan job identifier with loadleveler job identifier.

```
export JOBID_LL=`darshan-JOBID.sh $LOADL_STEP_ID`
```

- Set up environment variable `DARSHAN_JOBID` to environment variable name that contain the job identifier of loadleveler.

```
export DARSHAN_JOBID=JOBID_LL
```

- Set up Darshan log path

```
export LOGPATH_DARSHAN_LRZ=`darshan-logpath.sh`
```
I/O performance estimate (at the MPI-IO layer): transferred 4048187.3 MiB at 1176.54 MiB/s
Outline

- SuperMUC supercomputer
- User Projects
- Monitoring Tool
- Darshan Tool
- Persyst and Darshan
- Conclusions
Analysing I/O data of Darshan and PerSyst

Two cases:

- Simple pattern: BT-IO Class E and 1296 MPI processes. 1 Shared File of 2 TB. Total Data Transferred 4 TB (Write 2 TB and read 2 TB). Similar Request Size for read and write operations.

- Complex Pattern: ECHO parallel application. Three HDF5 shared files and four POSIX small files. Total I/O near to 18 GiB. Different request sizes.
**BT-IO Class E - PerSyst**

Explanation: Bytes read divided by the number of read operations (Scope: Node Unit: ratio)

Hint: A low ratio means too many separate requests with little data sent. Try to use fewer IO requests which send more data to the filesystem together.

Avg Request Size ~ 16 MB
Darshan reports read operations with more I/O cost and PerSyst with more severity
PerSyst detects this request size

### Most Common Access Sizes (POSIX or MPI-IO)

<table>
<thead>
<tr>
<th>access size</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>POSIX</td>
<td></td>
</tr>
<tr>
<td>16777216</td>
<td>251100</td>
</tr>
<tr>
<td>3959638</td>
<td>8000</td>
</tr>
<tr>
<td>3959584</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>access size</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPI-IO ‡</td>
<td></td>
</tr>
<tr>
<td>32752160</td>
<td>8900</td>
</tr>
<tr>
<td>32751040</td>
<td>8000</td>
</tr>
<tr>
<td>32753280</td>
<td>4500</td>
</tr>
<tr>
<td>32757840</td>
<td>1400</td>
</tr>
</tbody>
</table>

### File Count Summary (estimated by POSIX 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>1</td>
<td>2.0T</td>
<td>2.0T</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>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>read/write files</td>
<td>1</td>
<td>2.0T</td>
<td>2.0T</td>
</tr>
<tr>
<td>created files</td>
<td>1</td>
<td>2.0T</td>
<td>2.0T</td>
</tr>
</tbody>
</table>

‡ NOTE: MPI-IO accesses are given in terms of aggregate datatype size.
Two cases:

- Simple pattern: BT-IO Class E and 1296 MPI processes. 1 Shared File of 2 TB. Total Data Transferred 4 TB (Write 2 TB and read 2TB). Similar Request Size for read and write operations.

- Complex Pattern: ECHO parallel application. Three HDF5 shared files and four POSIX small files. Total I/O near to 18 GiB. Different request sizes.
Persyst detects small request sizes.
Darshan reports counters per file (the plot corresponds to a HDF5 file)

PerSyst does not detect number of files. Severity corresponds to all I/O activities.

<table>
<thead>
<tr>
<th>Job String ID</th>
<th>IO_BytesRead</th>
<th>IO_BytesReadPer</th>
<th>IO_Closes</th>
<th>IOOpens</th>
<th>IO_WrittenBytes</th>
<th>IOBytesWritten</th>
<th>IOBytes Read</th>
<th>IOBytes read p</th>
</tr>
</thead>
<tbody>
<tr>
<td>avg Severity</td>
<td>0e+0</td>
<td>388.66e-3</td>
<td>0e+0</td>
<td>0e+0</td>
<td>0e+0</td>
<td>261.05e-3</td>
<td>0e+0</td>
<td>0e+0</td>
</tr>
</tbody>
</table>
PerSyst should detect this request per file HDF5 and very small request for the other files.

This request corresponds to 16 (MPI processes per compute node) x MPIIO access sizes (collective buffering technique)

**Most Common Access Sizes**

<table>
<thead>
<tr>
<th>POSIX Access Sizes</th>
<th>POSIX access size</th>
<th>POSIX count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1012143</td>
<td>6132</td>
<td></td>
</tr>
<tr>
<td>1011791</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>272</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>544</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>63488</td>
<td>37440</td>
<td></td>
</tr>
<tr>
<td>65536</td>
<td>31200</td>
<td></td>
</tr>
<tr>
<td>61504</td>
<td>11232</td>
<td></td>
</tr>
<tr>
<td>61440</td>
<td>7200</td>
<td></td>
</tr>
</tbody>
</table>

**MPI-IO Access Sizes**

<table>
<thead>
<tr>
<th>MPI-IO access size</th>
<th>MPI-IO count</th>
</tr>
</thead>
<tbody>
<tr>
<td>63488</td>
<td>37440</td>
</tr>
<tr>
<td>65536</td>
<td>31200</td>
</tr>
<tr>
<td>61504</td>
<td>11232</td>
</tr>
<tr>
<td>61440</td>
<td>7200</td>
</tr>
</tbody>
</table>

**File Count Summary**

<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>1</td>
<td>5.8G</td>
<td>5.8G</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>1</td>
<td>5.8G</td>
<td>5.8G</td>
</tr>
<tr>
<td>read/write files</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>created files</td>
<td>1</td>
<td>5.8G</td>
<td>5.8G</td>
</tr>
</tbody>
</table>

**NOTE:** MPI-IO accesses are given in terms of aggregate datatype size.
Conclusions

• Darshan provides detailed information about the I/O characteristics. Only total counters per file at POSIX and MPI-IO level. Temporal pattern is not provided.

• PerSyst provides some specific I/O counters per interval (10 min) per job. It is not possible to know the total files, total I/O and I/O processes (and other I/O counters of Darshan).

• Darshan allows us to know the counts of files, file size, number of I/O processes and other important counters.

• We should consider the buffer size of the MPI-IO library for analyzing the request sizes (Collective Buffering, Data Sieving).

• We can use both tools to obtain knowledge about I/O performance, but we need select temporal fields of PerSyst and associate them with the I/O profiling provided by Darshan to learn more about I/O activities.
Thank you for your attention!