HPS

Esther Hagenkort
Supervisor: Patrick Höhn

Influence of the file system on the performance of machine learning workloads

Project set up and preliminary results

Table of contents

Outline

Input/Output bottleneck is a problem for machine learning

- ML gaining more and more importance
  - Better and more efficient algorithms
  - Increasingly large datasets for training
- I/O takes up 90% of total training time
  - Costs resources such as time and money
  - Environmentally unsustainable

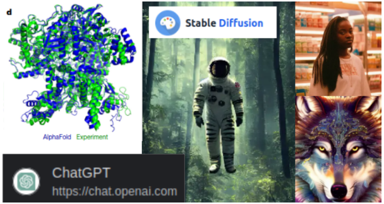


Figure: Examples of popular machine learning models

Pumma et al., "Scalable Deep Learning via I/O Analysis and Optimization"
Jumper et al., "Highly accurate protein structure prediction with AlphaFold"
https://chat.openai.com/
https://stablediffusionweb.com/

Storage file systems

- Distribution across multiple servers
- I/O parallelism
- Redundancy for host failure security
- Scalability of performance and capacity
- Separation of functionality
  - Object Data Storage (ODS)
  - Metadata Server (MDS)
  - Access Server

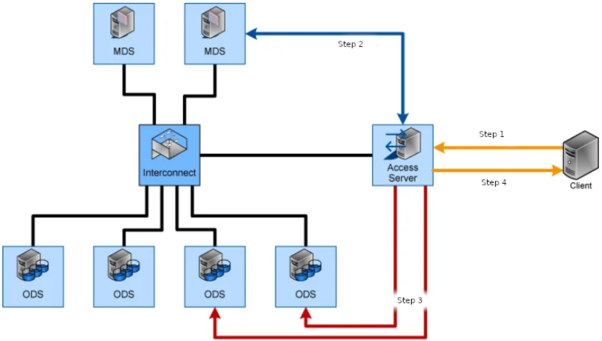


Figure: Exemplary storage file system architecture

https://www.itwm.fraunhofer.de/en/departments/hpc/fraunhofer-parallel-file-system-beegfs.html
https://www.comconsult.com/hochleistungs-dateisysteme/

Lustre

- Open source parallel file system
- By researchers of Carnegie Mellon University of the US
- NHR@Göttingen: Lustre at GWDG
  - Emmy: CPU Cluster
  - Grete: GPU Cluster

Figure: Supercomputer Emmy tile

https://www.lustre.org/about/
https://gwdg.de/hpc/systems/emmy/
https://info.gwdg.de/news/how-to-use-our-new-gpu-cluster-grete-for-hlrn-users/
https://gwdg.de/hpc/systems/

BeeGFS

BeeGFS®

- Shared source parallel file system
- By Frauenhofer Institute
- BeeGFS at GWDG
  - Scientific Compute Cluster (SCC): CPU & GPU Cluster

Figure: Supercomputer SCC tile

https://www.itwm.fraunhofer.de/en/departments/hpc/fraunhofer-parallel-file-system-beegfs.html
https://www.beegfs.io/c/download/
https://gwdg.de/hpc/systems/scc/
https://gwdg.de/hpc/systems/

Outline

1. Introduction to storage file systems
2. Project set up
3. Preliminary results
4. Appendix

Dataset: Conceptual Captions

- ~3.3M image-caption pairs
- Harvested from web by Google AI
- Competition: image captioning task
- Project subset
  - 12 720 images
  - Size: ~8GB

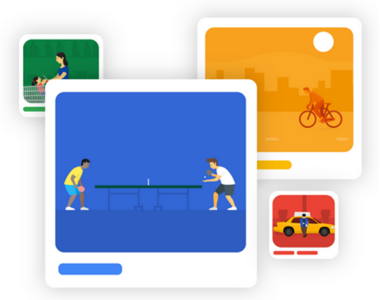


Figure: Google AI Conceptual Captions dataset logo

https://ai.google.com/research/ConceptualCaptions/
https://huggingface.co/datasets/conceptual_captions

HuggingFace provides

- In general
  - A collaboration platform for ML community
  - Open-source ML libraries, datasets and models

https://huggingface.co/brand
https://huggingface.co/datasets/conceptual_captions

HuggingFace provides

- In general
  - A collaboration platform for ML community
  - Open-source ML libraries, datasets and models
- For the conceptual captions dataset
  - Data including image_url, caption, labels and some other information
  - Code to load dataset and fetch the images
  - Note: save to `scratch` not `home` filesystem

```python
from datasets import load_dataset
dset = load_dataset("conceptual_captions")
```

https://huggingface.co/brand
https://huggingface.co/datasets/conceptual_captions

Emmy did not like HuggingFace

- Provided download code produced...
  - 76 lines of error traceback
  - File "/scratch/usr/nimestha/mambaforge/envs/scap_env/lib/python3.12/site-packages/requests/adapters.py", line 507, in send raise ConnectTimeout(e, request=request) requests.exceptions.ConnectTimeout: (**MaxRetryError**("HTTPSConnectionPool(host='huggingface.co', port=443): Max retries exceeded with url: /api/whoami-v2 (Caused by **ConnectTimeoutError**(<urllib3.connection.HTTPSConnection object at 0x2aaac1c42ab0>, 'Connection to huggingface.co timed out. (connect timeout=None)'))"), '(Request ID: bb81d713-1a76-4b53-b323-62811a84ec7a)'

Emmy did not like HuggingFace cont.

- Workaround
  - Use downloaded files on SCC
  - Upload with `scp` command for file transfer

```
$ scp directory_to_upload remote_username@glogin.hlrn.de:/remote/directory
```

Darshan

- Open source I/O characterisation tool
- Post mortem analysis
  - Elapsed time
  - Access sizes
  - Access pattern
  - File names for each file opened by an application

Figure: Darshan web logo

Kunkel et al., "Tools for analyzing parallel I/O"
https://www.mcs.anl.gov/research/projects/darshan/

Darshan installation

```
$ module load openmpi
$ wget https://ftp.mcs.anl.gov/pub/darshan/releases/darshan-3.4.4.tar.gz
$ tar -xvzf darshan-3.4.4.tar.gz
$ cd darshan-3.4.4/
$ ./prepare
$ cd darshan-runtime/
$ ./configure --with-log-path=/darshan-logs
    --with-jobid-env=SLURM_JOB_ID
    --prefix=/scratch/users/username/darshan/ CC=mpicc
$ make & make install
$ cd ../darshan-util/
$ configure --prefix=/scratch/users/username/darshan/
$ make & make install
```

https://www.mcs.anl.gov/research/projects/darshan/docs/darshan-runtime.html
https://www.mcs.anl.gov/research/projects/darshan/docs/darshan-util.html

Darshan log file usage

- Darshan-parser
  - Included in darshan-util
  - Creates complete, human-readable, text-format version of log files

https://www.mcs.anl.gov/research/projects/darshan/docs/darshan-util.html
https://pypi.org/project/darshan/3.4.0.0/#description

Darshan log file usage

- Darshan-parser
  - Included in darshan-util
  - Creates complete, human-readable, text-format version of log files
- Darshan-job-summary.pl
  - Also included in darshan-util
  - Creates graphical summary of the I/O activity as PDF

https://www.mcs.anl.gov/research/projects/darshan/docs/darshan-util.html
https://pypi.org/project/darshan/3.4.0.0/#description

Darshan log file usage

- Darshan-parser
  - Included in darshan-util
  - Creates complete, human-readable, text-format version of log files
- Darshan-job-summary.pl
  - Also included in darshan-util
  - Creates graphical summary of the I/O activity as PDF
- PyDarshan
  - Python utilities to interact with Darshan log files
  - Requires darshan-util
  - Install via pip

```
$ pip install darshan==3.4.0.0
```

https://www.mcs.anl.gov/research/projects/darshan/docs/darshan-util.html
https://pypi.org/project/darshan/3.4.0.0/#description

Darshan log file usage: PyDarshan code example

pydarshan_ex.py

```python
import darshan

# Open darshan log
report = darshan.DarshanReport('example.darshan', read_all=False)

# Load some report data
report.mod_read_all_records('MPI-IO')
# or fetch all
report.read_all_generic_records()

# Generate summaries for currently loaded data
# Note: aggregations are still experimental and have to be activated:
darshan.enable_experimental()
report.summarize()
```

https://pypi.org/project/darshan/3.4.0.0/#description

Define jobs for slurm via batch script

run.sbatch

```
#! /bin/bash

#SBATCH --mem 32G
#SBATCH -p medium
#SBATCH -t 01:00:00

export LD_PRELOAD=/scratch/users/username/darshan/lib/libdarshan.so

source /usr/users/username/.bashrc
source activate scap_env

module load openmpi

srun python test_run.py
```

How to choose partition

- Description of available partitions
  - SCC: https://gwdg.de/en/hpc/systems/scc/

Name	Number of nodes	CPU & GPU	Number of CPU-Cores	Memory [GB]	Partition
amp	95	ⓘ 2 x Xeon Platinum 9242	48	384	[medium]

  - Emmy: https://www.hlrn.de/doc/display/PUB/Compute+node+partitions

Partition (number holds cores per node)	Node name	Max. walltime	Nodes	Max. nodes per job	Max. jobs per user	Usable memory MB per node	CPU, GPU type	Shared	NPL per node hour	Remark
standard96	gcn#	12:00:00	996	256	unlimited	362 000	Cascade 9242	✘	96	default partition

Outline

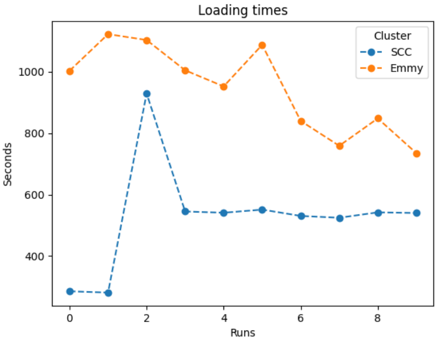
Some exploratory tests

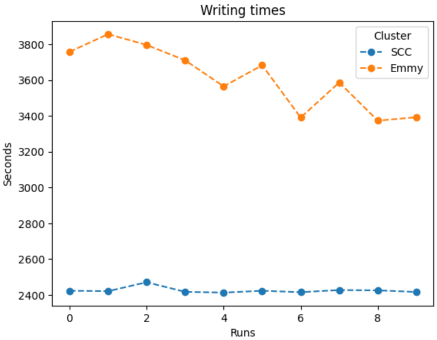
I/O type	SCC	Emmy
Reading	527.12 sec ≈ 8.79 min	945.18 sec ≈ 15.75 min
Writing	2425.50 sec ≈ 40.43 min	3611.15 sec ≈ 60.19 min

- Times averaged over 10 runs
- Used dataset
  - 12 720 images
  - ~8GB
- Used CPU partitions
  - SCC: medium (cascade lake)
  - Emmy: large40 (skylake)

Stopping time is not sufficient

- Standard deviation loading:
  - SCC: 167.84 sec ≈ 2.80 min
  - Emmy: 135.11 sec ≈ 2.25 min
- Standard deviation writing:
  - SCC: 15.79 sec ≈ 0.26 min
  - Emmy: 169.29 sec ≈ 2.82 min





More runs mean more variation

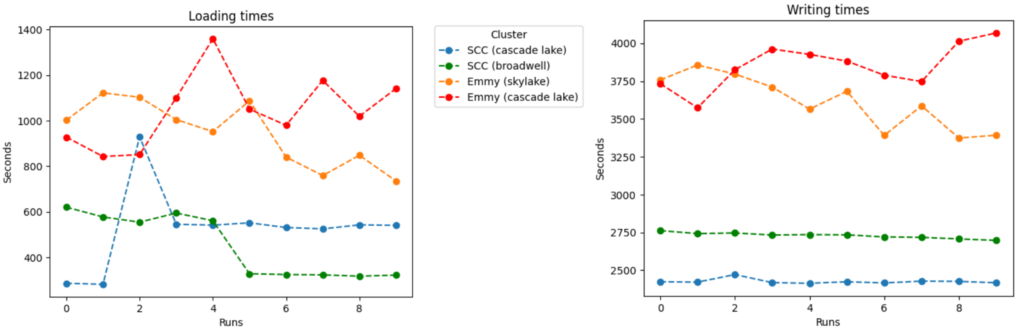
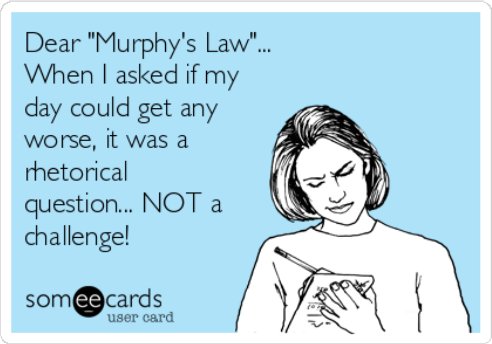


Table with avg. times and std in appendix

Preliminary results using Darshan

"If anything can go wrong, it will."
- Captain Edward A. Murphy



https://www.phrases.org.uk/meanings/murphys-law.html
https://medium.com/the-rookie-pm/on-murphys-law-agile-and-product-management-9158d3530a89

Summary

Project set-up

- Parallel file systems: BeeGFS and Lustre
- Dataset: Conceptual Captions 3M
- Measurement tool: Darshan

Summary

Project set-up

- Parallel file systems: BeeGFS and Lustre
- Dataset: Conceptual Captions 3M
- Measurement tool: Darshan

Preliminary results

- Stopping time not sufficient
- Trend: Emmy slower than SCC

Summary

Project set-up

- Parallel file systems: BeeGFS and Lustre
- Dataset: Conceptual Captions 3M
- Measurement tool: Darshan

Preliminary results

- Stopping time not sufficient
- Trend: Emmy slower than SCC

Future work

- Get Darshan running
- Analyse log files for I/O performance

References

Jumper, John et al. "Highly accurate protein structure prediction with AlphaFold". In: *Nature* 596.7873 (2021), pp. 583–589.

Kunkel, Julian Martin et al. "Tools for analyzing parallel I/O". In: *High Performance Computing: ISC High Performance 2018 International Workshops, Frankfurt/Main, Germany, June 28, 2018, Revised Selected Papers 33*. Springer. 2018, pp. 49–70.

Pumma, Sarunya et al. "Scalable Deep Learning via I/O Analysis and Optimization". In: *ACM Trans. Parallel Comput.* 6.2 (July 2019). ISSN: 2329-4949. DOI: `10.1145/3331526`. URL: `https://doi.org/10.1145/3331526`.

Outline

Result overview

Cluster		SCC		Emmy	
Processor gen.		broadwell	cascade lake	skylake	cascade lake
Reading	avg. times	7.53	8.79	15.75	17.41
	std	2.17	2.80	2.25	2.51
Writing	avg. times	45.48	40.43	60.19	64.20
	std	0.30	0.26	2.82	2.35

Table: Times for loading and writing roughly 8GB of image data on different partitions of the SCC and Emmy averaged over 10 runs and the standard deviations of those runs. All times are given in minutes and rounded to two decimals.

Useful slurm commands

- Submit batch script to start job

```
$ sbatch run.sbatch
```

- Review scheduled jobs

```
$ squeue -u username
  JOBID PARTITION       NAME     USER   STATE TIME NODES NODELIST(REASON)
5460973    medium run.sbatch username RUNNING 0:30     1 amp029
```

https://slurm.schedmd.com/documentation.html

Useful slurm commands cont.

- Cancel a job with `JOBID`

```
$ scancel 5460973
```

- Review available partitions and nodes

```
$ sinfo
PARTITION AVAIL  TIMELIMIT NODES STATE NODELIST
medium*      up 2-00:00:00     1  idle amp025
```

https://slurm.schedmd.com/documentation.html