Timeline-based I/O Behavior Assessment of Parallel Jobs

An Explorative Study on 10^6 Jobs

Analyzing Parallel I/O
November 20, 2019

Eugen Betke
Research Group
German Climate Computing Center

Julian Kunkel
Compute Science
University of Reading

Table Of Contents

Mistral, the HPC system for Earth system research (HLRE-3)

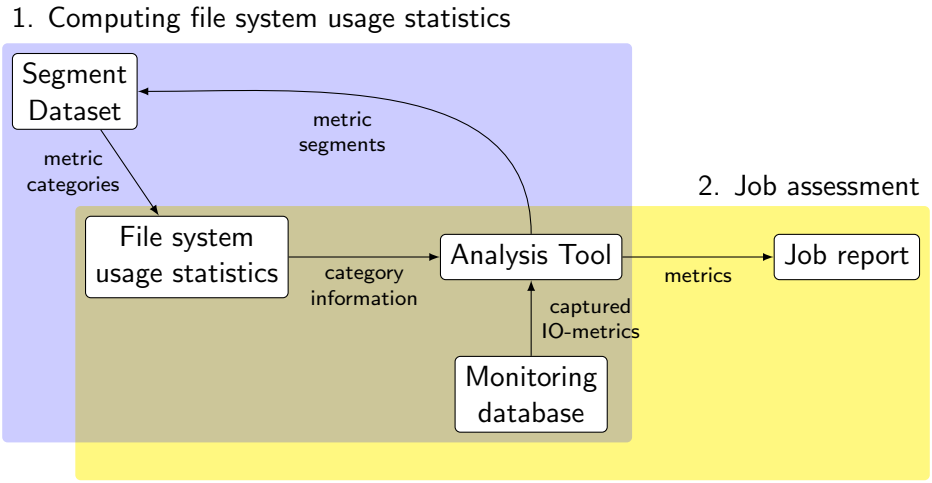
Peak performance	*3.14 PetaFLOPS*
Compute nodes	*3,300*
Compute cores	*100,000*
Memory	*266 Terabytes*
Storage (two file systems)	*54 Petabytes*

Motivation

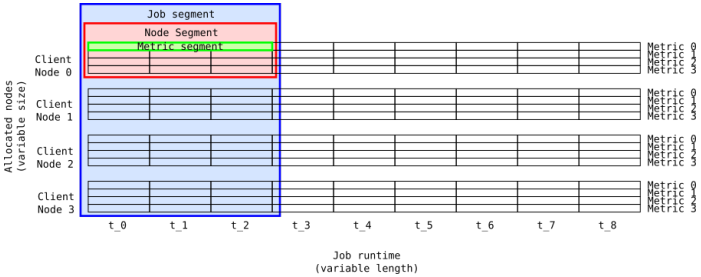
- Goals: Finding jobs with
  - high I/O load, but inefficient data access
    - e.g., for application optimization
  - critical I/O load, that can degrade file system performance
    - e.g., for better job scheduling
- Approach:
  - Define simple job metrics
  - Use them for ranking and comparison of jobs

Table Of Contents

Analysis Workflow



Segmentation and Scoring of Monitoring Data



1. Segmentation
   - Segment size = 3 time points (in this example only)
2. Categorization
   - Quantiles $q99$ and $q99.9$ define thresholds
3. Scoring
   - CriticalIO is at least 4x higher than HighIO

Category	Criteria	MScore
LowIO	smaller than q99	0
HighIO	between q99 and q99.9	1
CriticalIO	larger than q99.9	4

Categorization criteria and scores

Score name	Definition
MScore	0,1 or 4
NScore	$\sum$ MScore
JScore	$\sum$ NScore

Segment scores

Table Of Contents

Job-IO-Metrics [1/3]

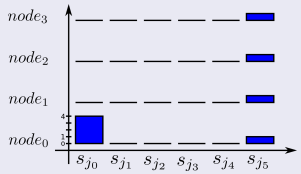
Job-IO-Balance $\quad \text{mean}\left(\left\{\frac{\text{mean\_score (j)}}{\text{max\_score (j)}}\right\}_{j \in \text{IOJS}}\right)$

Job-IO-Utilization $\quad \sum_{FS} \frac{\sum_{j \in IOJS} \text{max\_score}(j)}{N}$

Job-IO-Problem-Time $\quad \frac{\text{count (IOJS)}}{\text{count (JS)}}$

- FS: Filesystems
- JS: Job segments
- IOJS: IO-intensive job segments

Example



$$b_0 = \text{balance}(s_{j_0}) = 0.25$$
$$b_1 = \text{balance}(s_{j_5}) = 1$$
$$\text{Job-IO-Balance} = mean(\{b_0, b_1\}) = 0,625$$

Job-IO-Metrics [2/3]

Job-IO-Balance

$$\text{mean}\left(\left\{\frac{\text{mean\_score (j)}}{\text{max\_score (j)}}\right\}_{j\in\text{IOJS}}\right)$$

Job-IO-Utilization

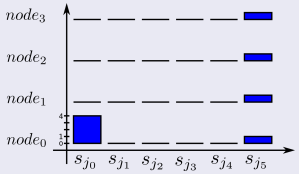
$$\sum_{FS}\frac{\sum_{j\in IOJS}\text{max\_score}(j)}{N}$$

Job-IO-Problem-Time

$$\frac{\text{count (IOJS)}}{\text{count (JS)}}$$

- FS: Filesystems
- JS: Job segments
- IOJS: IO-intensive job segments

Example



$$\begin{aligned}
\text{max}_0 &= \text{max_score}(s_{j_0}) &&= 4\\
\text{max}_1 &= \text{max_score}(s_{j_5}) &&= 1\\
U_{fs1} &= \mathit{mean}(\{\text{max}_0, \text{max}_1\}) &&= 2.5\\
U_{fs2} &= \mathit{mean}(\{\text{max}_0, \text{max}_1\}) &&= 2.5\\
\text{Job-IO-Utilization} &= U_{fs1} + U_{fs2} &&= 5
\end{aligned}$$

Job-IO-Metrics [3/3]

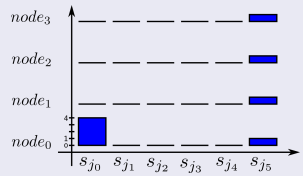
Job-IO-Balance $\quad \text{mean}\left(\left\{\frac{\text{mean\_score (j)}}{\text{max\_score (j)}}\right\}_{j\in\text{IOJS}}\right)$

Job-IO-Utilization $\quad \sum_{FS}\frac{\sum_{j\in IOJS}\text{max\_score}(j)}{N}$

Job-IO-Problem-Time $\quad \frac{\text{count (IOJS)}}{\text{count (JS)}}$

- FS: Filesystems
- JS: Job segments
- IOJS: IO-intensive job segments

Example

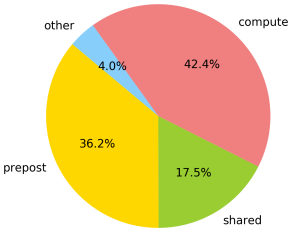


$$N_{IOJS} = 2$$
$$N_{JS} = 6$$
$$\text{Job-IO-Problem-Time} = \frac{N_{IOJS}}{N_{JS}} \approx 0.33$$

Table Of Contents

Monitoring Data of 10^6 jobs

- 1,000,000 jobs for time period of 99 days
  - from 2019–05–16 to 2019–08–23
- 323634 job data remain for analysis
  - Compute partition provides data for analysis
  - Only job with exit status COMPLETED are analysed



Slurm partitions

Captured Metrics

Metadata (Source: `/proc/fs/lustre/llite/lustre*-*/stats`)

1	`md_read`	getattr + getxattr + readdir + statfs + listxattr + open + close
2	`md_mod`	setattr + setxattr + mkdir + link + rename + symlink + rmdir
3	`md_file_create`	create
4	`md_file_delete`	unlink
5	`md_other`	truncate + mmap + ioctl + fsync + mknod

Data (Source: `/proc/fs/lustre/llite/lustre*-*/read_ahead_stats`)

6	`read_bytes`	Application I/O requests to Lustre.
7	`read_calls`	
8	`write_bytes`	
9	`write_calls`	
10	`osc_read_bytes`	Lustre object storage client requests to object storage.
11	`osc_read_calls`	
12	`osc_write_bytes`	
13	`osc_write_calls`	

File System Usage Statistics

Metric		Limits		Number of occurences		
Name	Unit	q99	q99.9	LowIO	HighIO	CriticalIO
md_file_create	Op/s	0.17	1.34	65,829K	622K	156K
md_file_delete	Op/s	0.00	0.41	65,824K	545K	172K
md_mod	Op/s	0.00	0.67	65,752K	642K	146K
md_other	Op/s	20.87	79.31	65,559K	763K	212K
md_read	Op/s	371.17	7084.16	65,281K	1,028K	225K
osc_read_bytes	MiB/s	1.98	93.58	17,317K	188K	30K
osc_read_calls	Op/s	5.65	32.23	17,215K	287K	33K
osc_write_bytes	MiB/s	8.17	64.64	16,935K	159K	26K
osc_write_calls	Op/s	2.77	17.37	16,926K	167K	27K
read_bytes	MiB/s	28.69	276.09	66,661K	865K	233K
read_calls	Op/s	348.91	1573.45	67,014K	360K	385K
write_bytes	MiB/s	9.84	80.10	61,938K	619K	155K
write_calls	Op/s	198.56	6149.64	61,860K	662K	174K

Job-IO-Metric Distributions

Job-IO-Balance

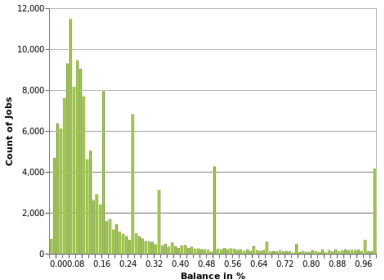
$$\text{mean}\left(\left\{\frac{\text{mean\_score (j)}}{\text{max\_score (j)}}\right\}_{j\in\text{IOJS}}\right)$$

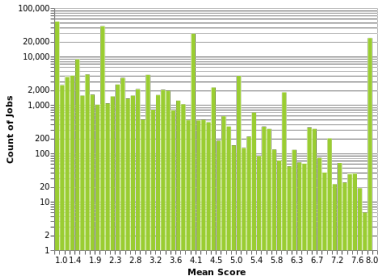
Job-IO-Utilization

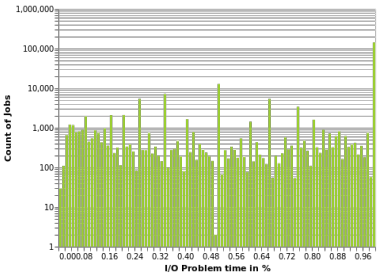
$$\sum_{FS}\frac{\sum_{j\in IOJS}\text{max\_score}(j)}{N}$$

Job-IO-Problem-Time

$$\frac{\text{count (IOJS)}}{\text{count (JS)}}$$

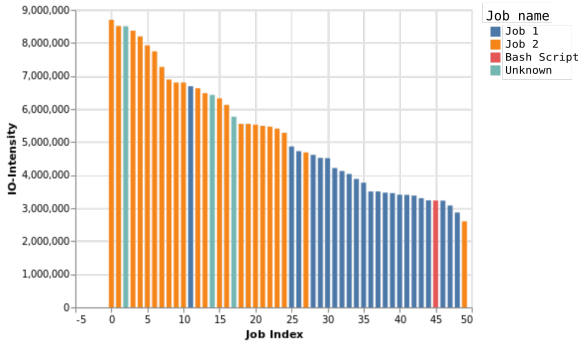




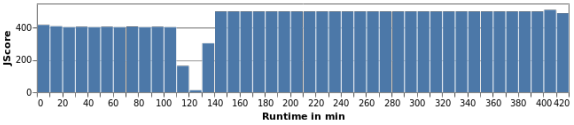


Jobs with high I/O-Intensity

$$\text{Job-IO-Intensity} = B \cdot PT \cdot U \cdot \text{total\_nodes}$$



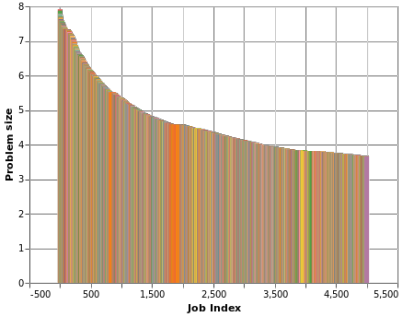
30 jobs ordered by IO-Intensity



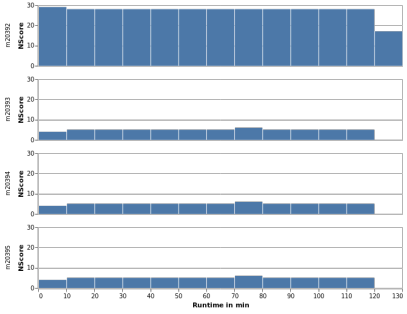
Nodes: 100; B: 0.88; PT:1.0; U: 4.0

Jobs with high Problem-Score

$$\text{Problem-Score} = (1 - \text{B}) \cdot \text{PT} \cdot \text{U}$$



5000 jobs ordered by Problem-Size, and runtime > 30min,



Example for high Problem-Size: Nodes: 4; B: 0.37; PT: 1.0; U: 8.0

Table Of Contents

Summary

- Applied methods
  - **Segmentation**: Preserves time line information
  - **Categorization**: Filters not significant I/O and make incompatible metrics compatible
  - **Scoring**: Allows mathematical computation
- Job-IO-Problem-Time, Job-IO-Balance and Job-IO-Utilization
  - Are **simple metrics**, that describe key properties of parallel jobs
- IO-Intensity and IO-Problem-Score
  - Are **penality functions**, used for job ranking

Thank you for your attention!