Future Generation Computer Systems 142 (2023) 117–130

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Less is not more: We need rich datasets to explore

Laurens Versluis [a,*], Mehmet Cetin [a], Caspar Greeven [b], Kristian Laursen [a,b], Damian Podareanu [b], Valeriu Codreanu [b], Alexandru Uta [c], Alexandru Iosup [a]

[a] VU Amsterdam, The Netherlands
[b] Surf Amsterdam, The Netherlands
[c] Leiden University, The Netherlands

ARTICLE INFO

Article history:
Received 14 May 2022
Received in revised form 10 December 2022
Accepted 12 December 2022
Available online 23 December 2022

Keywords:
Statistical analysis
Methodology
Dataset
Open-access
Datacenter
Holistic analysis

ABSTRACT

Traditional datacenter analysis is based on high-level, coarse-grained metrics. This obscures our vision of datacenter behavior, as we do not observe the full picture nor subtleties that might make up these high-level, coarse metrics. There is room for operational improvement based on fine-grained temporal and spatial, low-level metric data. We leverage in this work one of the (rare) public datasets providing fine-grained information on datacenter operations, with over 60 billion measurements captured in 15-second intervals. We show evidence that fine-grained information reveals new operational aspects, that the different metrics cannot be derived from one another (and thus need to be captured), and that many low-level metrics, gathered frequently are key to understanding datacenter operations. We propose a holistic analysis for datacenter operations, providing statistical characterization of node and workload aspects. Our analysis reveals both generic and machine learning-specific aspects, summarized in over 30 observations, providing deep insight into this dataset and the originating cluster. We give actionable insights, surprising findings, and exemplify how our observations support performance-engineering tasks such as workload prediction and long-term datacenter design.

1. Introduction

Datacenters have become the main computing infrastructure for the digital society [1]. Because datacenters are supporting increasingly more users and more sophisticated demands, their workloads are changing rapidly. Although our community has much data and knowledge about HPC and super computing workloads [2,3], we have relatively much less information on emerging workloads such as machine-learning, which seem to differ significantly from past workloads [4,5]. We posit in this work that, to design the efficient datacenters of tomorrow, we need comprehensive yet low-level machine metrics from datacenters. Such metrics could be key to optimization [6], performance analysis [7,8], and uncovering important phenomena [9]. Yet, comprehensive datasets of low-level datacenter metrics are rare [10–13]. Commercial providers are reluctant to publish such datasets, for reasons that include the need for commercial secrecy, adherence to privacy legislation, and lack of strong incentives to compensate for the additional effort. Often, published datasets are collected over short periods of time, with coarse time-granularity, not including low-level machine metrics. Some datasets have hardware specifications and rack topologies omitted and values obfuscated through normalization or other processes; only coarse, narrowly focused analysis can result from them. In contrast, in this work we propose a method that can be used to analyze rich, fine-grained datasets and apply it to an open-access dataset [14,15] that is currently available in the community. This dataset contains low-level and fine-grained operational server metrics gathered from a scientific computing infrastructure over a period of nearly 8 months at 15-s intervals. Low-level metrics are metrics that are not aggregated and at the level (or close to) the lowest units at which they can be reported, e.g., bytes transferred, pages accessed, rotations per minute in storage devices, Watt-hours consumed by a component, etc. Fine-grained refers to the temporal and spatial aspects of the dataset. The fine-grained temporal aspect originates from the low interval between measurements, typically comparable with task execution. The spatial aspect refers to the *physical* granularity (the origins) of the metrics, e.g., per component, per machine, per rack, etc. We show that such data are key in understanding datacenter behavior and we encourage datacenter providers to release such data and join our effort.

We focus on addressing three challenges related to a deeper understanding of datacenter operations. First, the *lack of work on diverse operational metrics.* For decades, the community has successfully been optimizing computer systems only for the metrics

* Corresponding author.
E-mail addresses: l.f.d.versluis@vu.nl (L. Versluis), a.uta@liacs.leidenuniv.nl (A. Uta), a.iosup@vu.nl (A. Iosup).

https://doi.org/10.1016/j.future.2022.12.022

we measured—e.g., throughput [16], job completion time [17], latency [18], fairness [19]—and biased toward the workloads and behavior that have been open-sourced [2,3,10–13,20]. This suggests that capturing only a few metrics is insufficient to get a comprehensive view on the datacenter operation, as most metrics cannot be reliably derived from another. Instead, to capture possibly vital information, we should aim to include as much data as possible, from hardware sensors, to operating systems, and further to the application level.

We identify as a second challenge the *lack of holistic analysis methods*, able to combine and then work on diverse operational metrics such as workload and machine metrics. Previous research already points out that large bodies of modern research might be biased toward the available datasets [3,21], and that effort to measure "one level deeper" is still missing [22]. High-level metrics are typically composed of various low-level metrics. These low-level metrics are instrumental in improving the performance of a system, yet including all low-level metrics in a model is likely infeasible to optimize; a balance must be struck. One of the findings in this work is that most metrics cannot be reliably derived from one another. Thus, one needs to carefully consider which metrics to include and capture. Next to operational bias, this also results in understudying other metrics and limits our ability to fully understand large-scale computer systems. For example, only since the 2000s and more intensively only after the mid-2010s, has energy consumption become a focus point [23,24]. In pioneering work in operational data analytics in the late-2010s, Bourassa et al. [25] propose to conduct extensive data collection and feed the results back into running datacenters for improving operations. Pioneering software infrastructures such as GUIDE [26] and DCDB Wintermute [27] take first steps in this direction. However, much more research is needed to understand the kinds of data and analysis feasible (and necessary) in this field. Similarly, many studies and available datasets focus only on computational aspects, e.g., [2,3,20,28], but details on the operation of machine-learning workloads on infrastructure equipped with GPUs (and, further, TPUs, FPGAs, and ASICs) are still scarce.

As a third challenge, we consider the relative *lack of relevant, fine-grained, and public datasets*. In practice, collecting holistic data has been feasible at the scale of datacenters for nearly a decade, with distributed monitoring [29], tracing [30], and profiling [31] tools already being used in large-scale datacenters. Unfortunately, such data rarely leaves the premises of the datacenter operator. From the relatively few traces that are shared publicly, many are focused on important but specific kinds of workloads, such as tightly-coupled parallel jobs [2], bags of tasks [32], and workflows [20]. Other datasets only include a limited subset of metrics such as power consumption [33], or high-level job information [34]. Only a handful of datasets include low-level server metrics, such as the Microsoft Azure serverless traces [13] or the Solvinity business-critical traces [35]. Recently, in 2020, the largest public infrastructure for scientific computing in the Netherlands has released as Findable, Accessible, Interoperable, Reusable (FAIR) [36], open-access data a long-term, fine-grained dataset about their operations [14,15]. In this work, *we conduct a deep analysis of the datacenter operations captured by this dataset*.

Addressing these challenges, we advocate for a more holistic view of datacenter operations, with a four-fold contribution:

1. Motivated by the need for diverse operational metrics, we propose a method for the analysis of datacenter operations (Section 3) using rich datasets. Our method considers information about both the workload and machine, and provides comprehensive statistical results.
2. We show the benefits of our method in understanding the long-term operations of a large public provider of scientific infrastructure (Sections 4–5). We provide deep insights into a large-scale, fine-grained and, most importantly, public dataset [14,15]. Unique features of our analysis include a comparison of generic and machine-learning workloads and nodes, per-node analysis of power consumption and temperature. We also note that the working from home due to COVID-19 did not significantly impact datacenter usage.
3. We analyze whether diverse operational metrics are actually needed (Section 6). We conduct a pair-wise correlation study across hundreds of server metrics, and analyze whether correlations are enough to capture datacenter behavior. Our results show strong evidence about the need for a more diverse set of metrics, to capture existing operational aspects.
4. We explore ways to leverage holistic-analysis results to improve datacenter operations (Section 7). We propose actionable insights, assessing overheads of collecting more data and metric correlations. We also exemplify long-term use for design and tuning.

2. The LISA system model

In this section we present the system model of LISA, the datacenter the data originates from and present the outline of its operations.

Infrastructure: In total, the datacenter contains 349 nodes heterogeneous spread across 20 racks. The details of the nodes are listed in Table 1. Racks are either *generic*, including nodes only with CPUs, or for machine learning (*ML*), with all nodes having both CPUs and GPUs. To classify nodes as ML-nodes, the datacenter operators investigated the workloads executed on these nodes. Over 90% of the workload on each of the GPU nodes is from the ML domain, a determination based on the libraries used by each job. These nodes can only be reserved and used with special privileges assigned by the datacenter administrators. Each rack includes up to 32 *generic nodes* or up to 7 *ML nodes*; the number of generic and ML nodes per rack depend on the CPU and GPU models used, combined with the power-consumption limitations imposed by the cooling system.

In the datacenter, several nodes are used as entry, administrator, and/or compilation nodes; here, users can compile libraries or programs, process data, generate, etc. without interfering with jobs running on the other nodes. These nodes were omitted from our analyses.

Workload: The datacenter acts as infrastructure for over 800 users, who have submitted in the period captured by the dataset over 1 million jobs. The majority of these jobs originate from the bioinformatics, physics, computer science, chemistry, and machine learning domains. Jobs are exclusive per user; there are no multi-user jobs or workflows at the moment. SLURM is the cluster manager used to allow users to queue jobs for these different types of nodes. Machines can either be reserved for a certain amount of time, or jobs can be submitted to a job queue where they are executed on a resource that at least meets the requirements specified when submitting the job. All jobs are scheduled using FIFO per stakeholder, with fairsharing across stakeholders. The datacenter offers both nodes with co-allocation of jobs or exclusive use through the use of distinct queues. The operator uses `cgroups` to enforce CPU and memory limits on multi-tenant nodes.

Monitoring: The usage of nodes and the state of the machine and job queues are monitored by SLURM and output to log files which can be queried. Across the majority of LISA, Prometheus has been employed to capture the fine-grained temporal and spatial data of each node at specified intervals (15 s). This data is further complemented by libraries from Intel to capture additional CPU metrics and by libraries from NVIDIA to capture GPU-related metrics.

Table 1
Breakdown of all nodes and their characteristics within LISA.

# Nodes	Processor type	Clock	Scratch	Memory	Sockets	Cache	Cores	Accelerator(s)	Interconnect
23	Bronze 3104	1.70	1.5 TB NVMe	256 GB	2	8.25 MB	12	4 x GeForce 1080Ti, 11 GB GDDR5X	40 Gb/s
2	Bronze 3104	1.70	1.5 TB NVMe	256 GB	2	8.25 MB	12	4 x Titan V, 12 GB HBM2	40 Gb/s
29	Gold 5118	2.30	1.5 TB NVMe	192 GB	2	16.5 MB	24	4 x Titan RTX, 24 GB GDDR6	40 Gb/s
192	Gold 6130	2.10	1.7 TB	96 GB	1	22 MB	16		10 Gb/s
96	Silver 4110	2.10	1.8 TB	96 GB	2	11 MB	16		10 Gb/s
1	Gold 6126	2.60	11 TB	2 TB	4	19.25 MB	48		40 Gb/s
6	Gold 6230R	2.10	3 TB	376 GB	2	35.75 MB	52		2 x 25 Gb/s

3. A method for analyzing rich operational datasets

We propose in this section a holistic analytical method for analyzing rich operational datasets, which capture datacenter operations with many and diverse metrics, over long periods of time, and with fine granularity. Our method is data-driven, and thus we detail the input data (Section 3.1) and the method we use for data cleanup (Section 3.2). The highlight of this section is the data analysis method, for which we describe the main research questions and how we address them (Section 3.3). We highlight in Section 3.3 the research questions not addressed by prior work. We also cover the main limitations we see to our method (Section 3.4). To enable reproducibility, we validate and open-source all the software used in this work, see Section 3.3.

3.1. Input data

Although our method does not depend on specific metrics, we are mindful of the information currently available as public datasets. We take as model a public dataset originating from LISA. This dataset has the finest temporal and spatial granularity current available in any open-sourced datacenter metric dataset. This temporal granularity originates from the sampling rate used to capture the data, which is 15 s, over a time period of nearly 8 months. The spatial granularity originates from the rich number of high- and low-level server and rack metrics collected and made available by the datacenter operators of Surf [14,15]. These temporal and spatial granularities may enable more detailed and novel insights into the workings of datacenters, which we investigate in this work.

Overall: Table 2 summarizes the public dataset: up to 1,26 million samples per metric per node, and in total 66 billion individual, high- and low-level metric measurements. The *low-level metrics* include server-level (e.g., power consumption), hardware-sensor (e.g., fan speeds, temperature), and OS-level metrics (e.g., system load).

Table 2
Generic outline of the machine metric dataset.

Dataset	Item	Value
Public data (see Section 3.1)	Start date	2019-12-29
	End date	2020-08-07
	Sampling frequency [s]	15
	Max. samples per metric per node	1,258,646
	Number of metrics	327
	Number of measurements	66,541,895,243
Clean data (see Section 3.2)	Number of valid racks	15
	Number of valid nodes	315
	Number of valid measurements	63,978,689,791

3.2. Data cleanup

We first clean the dataset. The clean dataset is unprecedentedly rich (see Table 2). It includes:

Clean node- and rack-data: We include only the 315 nodes in 15 racks that are used for computation and for which data is available. Together, these nodes contain 5352 CPU cores, 41.6 TB of CPU memory, 128 GPUs, and 1.8 TB of GPU memory. Most nodes (283) only contain CPUs; the others (32) also have GPUs attached.

Clean job-data: For the workload, we filter out the jobs based on their start time if they are outside the start and end time range of the dataset. Additionally, all jobs that are not related to the racks in the machine dataset are filtered out, as they run in the 5 racks used as public gateways, and as compile, debug, and test farms.

Clean metric-data: When performing numerical analyses, we removed the NaN values or set them to, e.g., zero when summing. Overall, the original dataset contains over 66 billion measurements, with close to 2.6 billion NaN values (3,85%). For some metrics, the dataset contains some gaps where the monitoring system was down; for some others, data collection stopped halfway into May 2020.

Clean time-series: We filter out all missing measurements (not-a-numbers, NaNs). In visual overviews, we mark missing data using special coloring.

Clean correlation-data: When computing correlations between pairs of metrics, we omit pairs where one or both metrics' measurements never change, because such data are unfit for the ranking step required to compute the Spearman and Kendall correlations.

3.3. Data analysis

Our method for holistic analysis proposes diverse research questions, answered using a combination of machine and workload data.

Machine and workload data: As main input dataset, we use the clean dataset introduced in Section 3.2. For the workload analysis, the datacenter cannot publish the workload data due to privacy constraints (the EU GDPR law); instead, we contacted the datacenter operators and worked with them to run the analysis we need on the data. For the COVID-19 analysis, we record that the Dutch government declared the start of the (ongoing) pandemic on Feb 27, 2020 [37]; we thus consider all data before this date to be "non-covid" data.

Method FAIRness [36]**:** The scientific community is a powerful advocate for FAIR data. The dataset used in this work is FAIRly stewarded by Zenodo, and comes with a full specification and a data-schema that allow sharing and using the data with low effort [15].

We open-source all the software (scripts) used in this work at https://github.com/sara-nl/SURFace. All scripts are validated for correctness by at least two persons.

Novelty of our method: Previous work [3,26,27,34] has performed individual analyses that align and overlap with our holistic analysis. However, we show that analyzing the combination of machine and workload (e.g., jobs) data, and combining prior methods of analysis with new analysis, leads to novel insights.

A. Analysis of machine operations (results in Section 4): To analyze how the datacenter machines behave over a long period of time, we use a variety of low-level metrics as input for answering the following research questions:

RQ1: What is the general resource usage? We aim to understand the usage of each server: the average system load; RAM, disk I/O, and GPU usage. We further study the average power consumption, the temperature, and the fan speed.

RQ2: What is the specific memory and network usage? The answer should include common ranges and modes in the distribution of memory consumption, etc., per node-measurement; linked when possible to known workload.

RQ3: What is the power consumption, per node and per rack? What is the rack temperature? We seek the (instantaneous) power consumption, including common ranges and modes. We want to further understand how the heat dissipates and if the cooling system is overwhelmed.

RQ4: How does the system load vary over time? We focus here on diurnal and longer-term patterns. (The current dataset does not enable seasonality analysis, but data keeps accumulating.)

RQ5: How do generic and ML nodes and racks differ? –orthogonal concern, applies to all other machine-related question.

RQ6: What is the impact of the COVID-19 pandemic?, especially how operations responded to workload changes that may result from the large social disruption associated with a pandemic.

B. Analysis of datacenter workload (results in Section 5): To understand if the *workload* exhibits similar properties to other traces known in the community, especially traces from scientific and Big Tech clusters, we formulate the following questions:

RQ7: What are the job characteristics?—job size in CPU-cores, job length, and variability across these features.

RQ8: What are the job arrival patterns? This question focuses on the basic statistics and time-patterns of job submissions.

RQ9: What is the peak demand?—explains the intensity of the peak demand, and contrasts it to normal operation.

RQ10: What are the patterns of job-failure?—fraction of jobs that fail to complete and their resource-waste.

RQ11: How do long jobs behave? We consider this orthogonal concern for each of the other workload-related questions.

C. Generating insights from data (results in Section 7):

RQ12: How can we leverage fine-grained temporal and spatial data?, focusing on using this data to perform better predictions.

RQ13: What are the implications of storing fine-grained temporal and spatial data? This question focuses on the feasibility of storage for fine-grained metric data as well as how scalable its analysis is.

RQ14: How do metrics correlate? This question focuses on insights into low-level metrics correlation and the implication for data collection and analysis.

RQ15: What are the implications of holistic analysis for datacenter operation and design? This focuses on leveraging fine-grained data to tune and design efficient datacenters.

3.4. Known limitations

We discuss here four known limitations to our method:

The most important limitation to our method derives from its *holistic nature*, which is also its strength. This nature is reflected in the broad analysis of several hundred metrics, which, as we show in the next three sections, helps understand how the whole works and gives actionable insights. However, datacenters can expose thousands of signals, so even our broad selection imposes a bias. Finding a *complete and general*, holistic method of analysis is beyond the scope of this work—a goal which we envision for the entire community, for the next decade, which already includes award-winning work that focuses on selecting meaningful signals [38] and large-scale data collection [26,27,33]. Furthermore, the method proposed here can be contrasted with methods from the other end of the holistic-reductionist spectrum; *compared with focused work* on even one of the questions we address, our method cannot produce the same depth for the same effort. Without rehashing the broad and as-of-yet inconclusive debate of the entire scientific world about holism vs. reductionism, we draw attention to its current stand-off: both add value and should not be discarded, lest the community that does so fail in producing scientific discoveries, long-term.

A second limitation derives from the *statistical methods* used in this work and from the libraries that compute them. For example, we use linear regression, because this is a common well-understood form of fitting. However, we envision that expert-level models could be developed, e.g., leveraging machine learning or higher order polynomials, giving better accuracy and precision. An example here could be to develop non-linear models where failures and even performance anomalies [39] are causally linked to signals from many metrics in the system, such as high load, extreme temperature, or unusual [40] and/or fail-slow hardware failures [9]. As discussed in Section 6, most metrics are not uniformly distributed, which is required for the Pearson correlation; nonetheless, the three correlation coefficients sketch a better picture together.

Another limitation is the *vantage point*, in that we look at data from a specific datacenter. This could affect especially the workload-level, where machine learning is emerging. However, more datasets as fine-grained as this work analyses are currently not available publicly—we encourage datacenter operators for all kinds of organizations to help!

Last, the dataset we analyze is much more fine-grained than others, but there is still much room for additional data and further analysis of it. For example, datasets could further include details on (i) the *operational policies*, e.g., detailed scheduling queues and thresholds (e.g., in the Parallel Workloads Archive, as defined by the community since the late-1990s [41]); (ii) the *trade-offs* considered during the capacity planning and datacenter design phases (e.g., of capability and cost); and (iii) the *energy sourcing and flows* (e.g., how the datacenter operations link with the energy markets and renewable energy-generation processes).

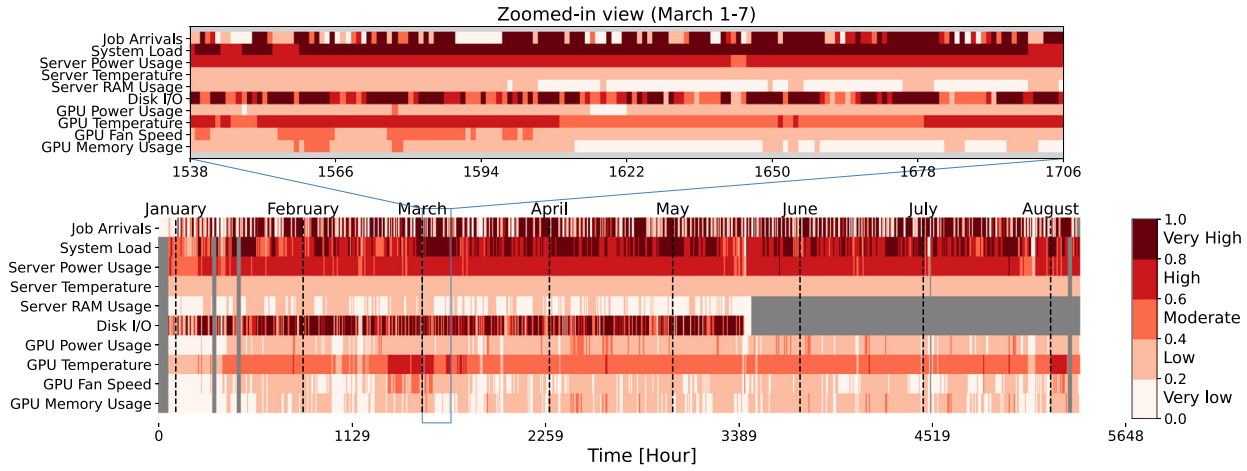


Fig. 1. Resource usage for various metrics. For this plot, we normalize the metrics and color them accordingly (see text). Each slice of a bar depicts one hour. We color each slice to the maximum value observed within that hour. Vertical dashed lines depict the start of a month. Grey depicts lack of valid data (see Section 3.2).

4. Datacenter machine operations

We present in this section a comprehensive characterization of machine operations in datacenters, with the method from Section 3.3.

4.1. General resource usage (RQ1, RQ4)

Observation-1: Job arrivals do not consistently overlap with machine metrics, including load, disk I/O. Jobs get queued.

O-2: Average system load is high (44.6%) or very high (20.2%).
O-3: Average RAM usage is low (33.3%) or very low (66.7%).
O-4: Average Disk I/O activity spikes to high or very high levels most hours.
O-5: GPU metrics indicate low to moderate average GPU usage.

To obtain a holistic view of the workload and how resources are being used, we plot the number of jobs arriving and various resource-related metrics in Fig. 1. Each slice of a bar in the figure depicts an hour, where the color of the given slice is set to the maximum normalized value observed within that hour. For the arrival of jobs, we count how many jobs arrive per 15 s interval (aligned with the metric samples) and then normalize the data using the 99th percentile clipped to 1, to avoid that a few outliers skew the normalization. We label five intensity classes—very low, low, moderate, high, and very high—spread equally in the normal range.

Setup: To depict how the overall datacenter is utilized, due to the absence of CPU utilization metrics, we use the UNIX *load1* as system load metric. UNIX load captures the "number of threads that are working or waiting to work" [42]. The load is an aggregate metric over time, e.g., load1 uses a 1-min rolling window. Values can exceed the number of available server cores, likely indicating system overload. Across all nodes, we sum load1 and divide it by the sum of their cores, clipped to 1.

The average *server power usage* is normalized to 5500 W—the maximum the cooling system can handle per rack. The *server temperature* is normalized to 77 °C—the minimum, across all the CPU models, of the maximum allowed temperature [43].

The *Server RAM usage* shows the utilization of all the RAM in the datacenter. To obtain *disk I/O* usage, we sum the bytes read and written from both local storage and NFS mounts and divide this number by the peak bandwidth achievable by a server. The datacenter does not contain burst buffers or a distributed file system. The peak bandwidth of 1.8 GB/s, obtained from benchmarks run in the datacenter, fits high-speed NVMe setups, or RAID-0 over multiple disks or SSDs.

The *GPU power usage*, *temperature*, and *fan speed* serve as proxy-metrics for GPU load, for which there is no direct utilization metric. The GPU power usage and temperature are normalized by the Thermal Design Point (TDP) and Thermal Threshold of each GPU [44–46]. The *GPU memory usage* captures the GPU memory consumed across the datacenter; per model, 11 GB (GTX 1080ti), 12 GB (Titan V), and 24 GB (RTX Titan).

Observations: From Fig. 1, we gain several interesting insights that would not have been possible only with high-level performance metrics. We observe that the number of jobs incoming does not always overlap with any other metric (**O-1**). Intuitively, one would assume that the load would increase based on an increased number of incoming jobs, but as can be observed and further discussed in our technical report [47], one or more nodes peak continuously to high levels—the average system load is typically moderate (18.2% of the measurements), high (44.6%), or very high (20.2%) (**O-2**). This matches observations in similar HPC infrastructures [33,34]. We also observe that the power consumption reaches high levels most of the time. This suggests that, combined with the observed CPU load, that there is little to no room to deploy energy saving techniques such as dynamic voltage scaling, which contrasts the findings of Patel et al. [33]. Next, we observe that some resources are barely used to their full potential, most notable RAM (**O-3**) and GPU memory. This aligns with other HPC infrastructures [48], and could lead to new designs of both applications and datacenter nodes. Overall, the disk I/O is spiky (**O-4**): 1.3% of all samples are at high disk I/O levels and 0.8% at very high levels. However, looking at Fig. 1, it shows for most hours these levels are reached at least once, indicating bursty behavior of disk I/O. Interestingly, the load on the GPUs is mainly low (**O-5**), although periods with moderate to heavy load exist as mentioned earlier. The zoomed-in view shows a small period of increased GPU usage, based on the GPU metrics. Server temperatures are low, even with a high rack power consumption, whereas GPU temperatures are moderate to high most of the time. We also observe periods with heavy disk I/O and sub-moderate CPU load, indicating the system is not used to its full potential; pipelining approaches or other parallel methods could help.

Table 3
RAM usage in the datacenter.

Percentile	1%	25%	50%	75%	90%	99%	100%
RAM [GB]	0.64	1.46	3.65	8.07	20.99	58.06	2000

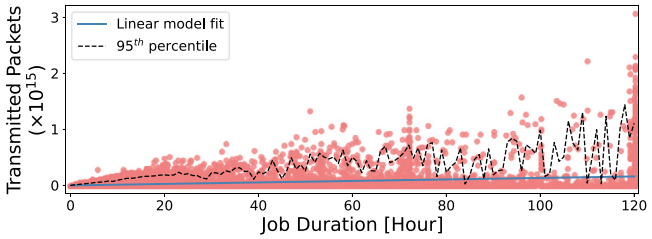


Fig. 2. Transmitted packets versus job duration.

4.2. Memory and network (RQ2)

O-6: 99% (75%) of RAM measurements fit within 64 GB (8 GB).
O-7: RAM usage has a very long tail, going up to 2 TB.
O-8: Longer jobs transmit more network packets, albeit not proportionally with the job duration.
O-9: The longer the job duration, the higher the probability of high outliers for the number of transmitted packets.

We characterize RAM usage for the entire dataset; Table 3 summarizes the basic statistics. We observe the RAM usage is low to moderate (**O-6**). Over 99% of all RAM measurements are below 60 GB of RAM, significantly less than the lowest RAM model (96 GB) in the datacenter. Three-quarters of all RAM measurements fit in 8 GB, matching other reports of memory underutilization in HPC environments [48]. Yet, RAM usage is long-tailed, up to 2 TB (**O-7**).

In Fig. 2 we plot the number of transmitted packets versus job time. We observe that shorter jobs seldom send more packets than longer running jobs, i.e., there are no extreme network-heavy yet short-running jobs (**O-8**). This could indicate that the majority of the network traffic is in the initial setup, e.g., downloading data; avoiding overloads across jobs could require advanced resource management techniques. Both the number of transmitted packets and the outliers generally increase over time, but only marginally. Outliers appear more likely for long-running jobs (**O-9**). We plot the increase in number of transmitted packets vs. job duration as the blue curve found by the linear regression model fit; the small increase matches MPI jobs generating typical TCP traffic. Further analysis that includes more sophisticated network models, e.g., traffic-congestion analysis, is outside the scope of this work but would be possible because the dataset also includes metrics such as TCP retransmission [15].

4.3. Power consumption (RQ3, RQ5, RQ6)

O-10: Generic nodes (racks) have more stable power consumption than ML nodes (racks).
O-11: Generic nodes consume 143 W on average, and up to 1300 W. ML nodes consume 467 W, and up to ≈1500 W.
O-12: Most racks, both generic and ML, exceed the threshold of the cooling system from time to time.
O-13: The covid pandemic did not alter datacenter operation.

Energy consumption is becoming increasingly important [49]. To better understand the power consumption within the datacenter, we observe power consumption using two different levels. First we show the distribution of power consumption per *rack*,

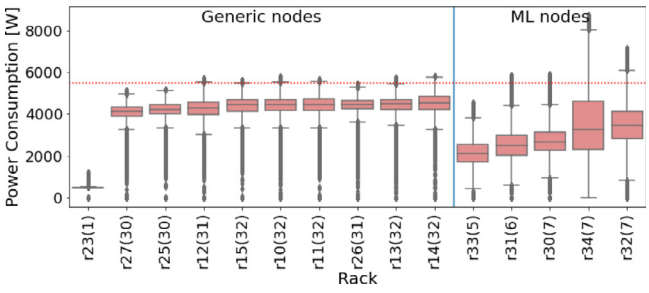


Fig. 3. Distributions of *rack* power consumption grouped by generic nodes and ML nodes. The labels show between parenthesis how many nodes each rack contains. The distributions are sorted by median per group. The dotted red line depicts the limit of the rack cooling system.

Table 4
Power consumption (Watt) of generic and ML *nodes*.

	1%	25%	50%	Mean	75%	99%	100%
Generic	80.00	100.00	148.00	143.01	176.00	260.00	1300.00
ML	130.00	260.00	364.00	467.16	624.00	1274.00	1508.00

in Fig. 3. We additionally group together generic nodes and ML nodes as the latter contain accelerators (GPUs).

Fig. 3 shows that there is little to moderate variation in generic node rack power consumption, except for rack 23. 86.4% of the time, the rack power consumption is within $0.6 - 0.8\times$ of the cooling system's capacity, with 0.1% of time it being in the range $0.8 - 1.0$. This matches the findings of Patel et al. [33]. Furthermore, the IQR ranges of the box plots show that most generic racks consume more power compared to ML racks. The ML racks show more variation and have higher extremes even though they contain fewer nodes (**O-10**), see Table 4. The fluctuations are due to the power profile of GPUs: idle they consume as little as 1 W, yet at full load their power consumption goes up as high as 416 W. As ML nodes have up to four GPUs, the power consumption can go significantly higher than generic nodes. The reason for rack 23 being an outlier is that it only hosts one node vs. 30–32 for the other generic node racks. Hence, this causes a lower power consumption profile for the rack.

Next, we investigate the power consumption of the *nodes* within each rack. From Table 4 we observe that generic nodes feature a small range, typically between 80–260 W. This range is somewhat smaller than reported by Netti et al. [27], likely due to a difference in hardware. Interestingly, the average power consumption for generic nodes is 143 W, which is well above the TDP of the most common CPU in our dataset (125 W TDP), contrasting the findings of Patel et al. [33].

Comparing the generic nodes with the ML nodes, we observe the generic nodes power consumption range is constrained, which in turn limit the ranges of the racks. As the generic node racks pack more nodes, they consume more energy, leading to the higher average seen in the previous discussed image. We also wondered if the lower number of nodes per ML rack is due to power supply unit or cooling system limitations. After inquiring the datacenter operators, the cooling system is indeed the limiting factor, only handling loads up to 5.5 kW per rack. We observe these are occasionally exceeded (**O-12**). Datacenter designs that include accelerators or aim for upgradeability have to consider this power-limiting aspect, underlined by the recently announced GPUs by Nvidia whose power consumption increased significantly (e.g., [50]) over older versions.

Finally, we assess the operation of the datacenter before and during the first months of the COVID-19 pandemic. We analyze various metrics; for brevity, we plot here only the average power

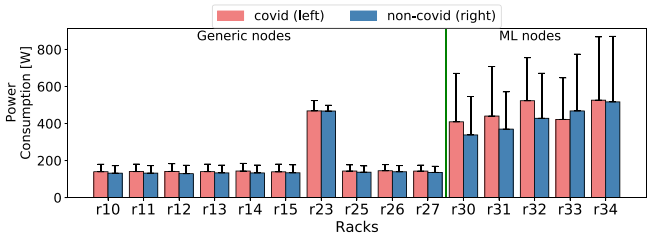


Fig. 4. Average power consumption for the covid and non-covid periods across all the racks.

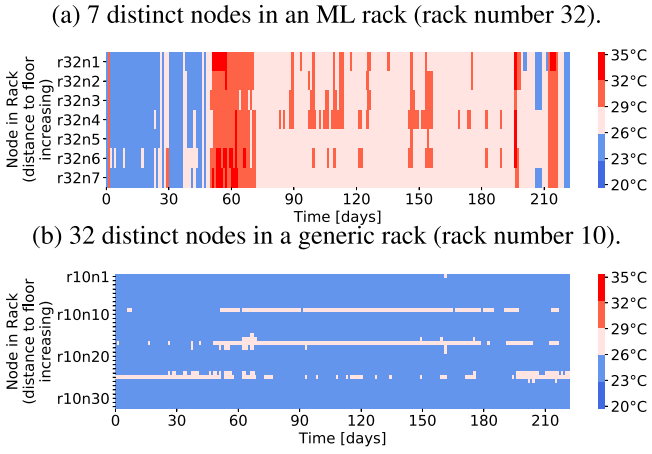


Fig. 5. Max daily temperature, ML vs. generic nodes.

consumption per rack, in Fig. 4. Overall, the COVID-19 pandemic did not significantly alter datacenter operations. This finding was surprising to the datacenter operation team, and could have positive implications for workload procurement. Only the ML-racks experience a moderate increase in power consumption (**O-13**). We observe similar stability for other metrics, cf. our technical report [47].

4.4. Rack temperature (RQ3, RQ5)

O-14: Temperature is correlated between nodes in the same rack.
O-15: Temperature and node position in rack are not correlated.
O-16: ML-racks run hotter than generic racks, by ≈3 °C.

The dataset we analyze in this paper contains multiple types of temperature-related metrics: GPU temperature, as well as server ambient temperature. While the former is the chip temperature, which is highly correlated with GPU workload, the latter is the temperature inside the server enclosure, which is influenced by many other factors: CPU workload, cooler (mal-)functioning, as well as warmer nearby nodes and distance from the datacenter floor. According to the datacenter operator, all nodes in this study are air cooled.

We find that nodes in ML racks tend to be correlated in terms of temperature (**O-14**). They are either mostly warmer, or mostly cooler. Fig. 5 plots this behavior. The graph shows the maximum temperature registered by servers in rack 32 for the entire period the dataset was collected. The graph also maintains the server ordering in the rack, with the smallest node ID at the top (see vertical axis). We notice that the node positioning in the rack does not influence its temperature (**O-15**). This finding matches the type of cooling used, i.e., air. Based on the experience of the datacenter operator, water cooling would not change the conclusion, because water cooling has superior heat dissipation. For the entire period, the lowest node temperature is around 20 °C, while the highest temperature is 35 °C. This range is significantly lower than reported by Netti et al. [27] where a range of 47–54 °C is reported. The difference could be caused by a different node hardware and cooling system combination. The figure depicts clearly that hotter periods are correlated over the entire rack. This type of behavior holds for all ML racks. If water cooling is used, it is likely that these temperatures would remain low and thus will not correlate as observed, due to the efficiency of these systems [51]. The generic racks are cooler: most nodes operate at 23–25 °C, ≈3 °C lower than most ML-rack nodes (**O-16**).

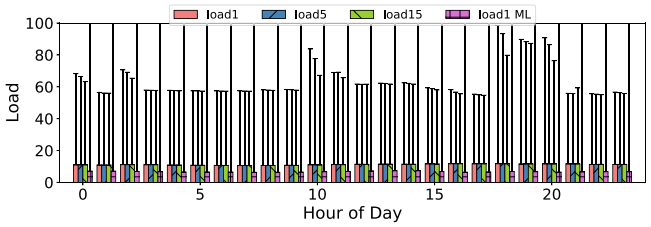


Fig. 6. The average UNIX load1, load5, and load15 metrics per hour of day. The error bars depict the standard deviation.

4.5. CPU diurnal load

O-17: The average system loads are stable. (See also **O-22**.)
O-18: Across all hours, ML nodes have an average load1 metric ≈40% lower than generic nodes.

To investigate the daily and weekly trends that may appear in the datacenter, we depict in Fig. 6 the load1, load5, and load15 UNIX metrics across the entire datacenter. We notice that the average load is very stable within the datacenter (**O-17**). The averages range between 10.6 and 11.8 for load1. Interestingly, this does not match the arrival pattern of jobs visible in Fig. 10. This might be due to the loads being regularly above 16, depicted by the error bars. This behavior indicates that processes are getting delayed as the most common node within the datacenter features 16 cores. In our technical report [47], we additionally show that load grouped per day of week also is stable, with a minimal elevation on Fridays and a small decrease in the weekend. Similarly to hour of day, the arrival of jobs does not correlate with the load.

When considering the load1 of ML nodes, we notice that it is stable, yet significantly lower than the cluster average. The average load1 per hour ranges between 6.3 and 7.4, which is around 40% lower than the average load across all machines (**O-18**). This indicates that these machines are utilized less. In Section 5, where we characterize the workload in-depth, we notice in Fig. 12 that indeed fewer users submit ML jobs.

5. Workload characterization

We characterize here the datacenter workload, with RQs from Section 3.3.

5.1. Job characteristics (RQ7)

O-19: Most jobs are small. Most jobs request less than 100 CPU cores, with a mode of 16 cores and max >500.
O-20: Most jobs are short: ≈90% of all completed jobs have a runtime ≤300 s.

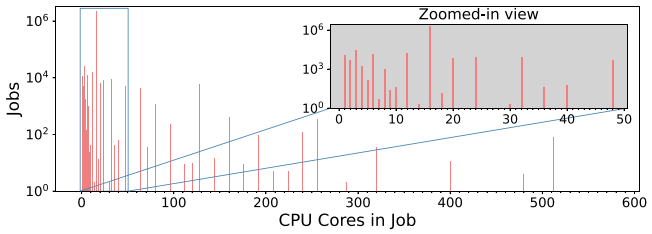


Fig. 7. Frequency distribution of allocated CPU-cores.

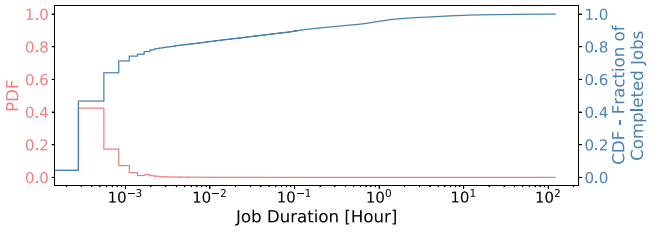


Fig. 8. Duration of completed jobs, CDF-PDF plot.

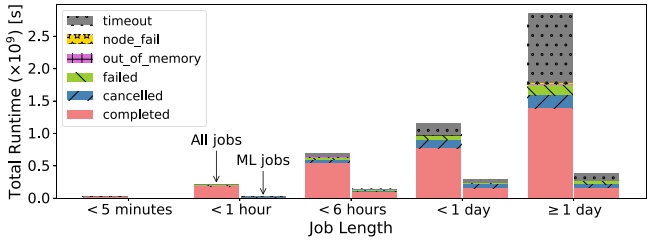


Fig. 9. Total job runtime grouped by job length. Per bar, we stack the runtimes per job state.

For job sizes, we depict the frequency of allocated CPU-cores in Fig. 7. Most jobs are small (**O-19**). Considering the number of requested cores (equal to the number of allocated cores in this system), Fig. 7 features a peak for 16 cores. This is equal, for example, to the number of requested cores in the Google trace [3]. As the most common node in the system has 16 cores, we believe most users simply request one full node using SLURM (the scheduler used within this datacenter); the default queue in SLURM enables this behavior. Most submitted jobs request less than 100 CPU cores, with extremes using over 500 CPU cores (**O-19**). The extremes are smaller than reported by other HPC clusters [34], yet small jobs being the majority of submitted jobs does hold.

We inspect the runtime of jobs within the datacenter. Fig. 8 shows the CDF of job durations. Most jobs are short: 88.9% of all completed jobs have a runtime of 5 min or less (**O-20**). Fig. 9 shows short-jobs also consume less, cumulatively, than long-running jobs. The cumulative runtime of short jobs is more than 177× smaller than for jobs running for a day or longer. Interestingly, jobs lasting up and until one hour take up a noticeably larger share when compared to other publicly available cluster traces [3,32,35]. Additionally, in our technical report we show that the daily footprint of submitted jobs in used CPU-hours seems to vary less [47].

5.2. Arrival patterns (RQ8), peak demand (RQ9)

O-21: Arrival and demand are highly variable. The number of submitted jobs per day varies by up to four orders of magnitude. (Also, the number of consumed CPU-hours varies by at most two orders of magnitude [47].)

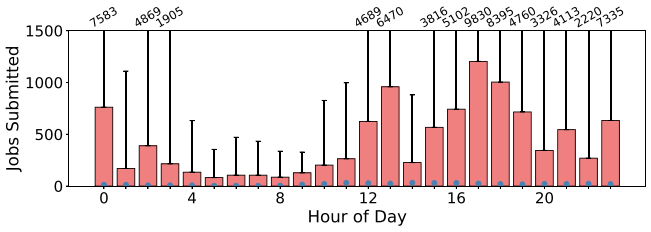


Fig. 10. Number of submitted jobs per hour of day. The blue dots depict the number of ML jobs.

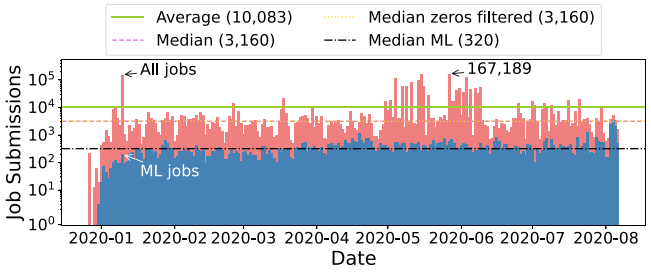


Fig. 11. Overview of the daily number of submitted jobs. The maximum is annotated. Logarithmic vertical axis.

O-22: Job submissions have a diurnal (office-like, e.g., 9 to 5) pattern.
O-23: The datacenter has a high job-arrival rate, with several days experiencing over 100,000 job submissions, each.
O-24: Significantly fewer ML jobs are submitted.
O-25: There are periods with high, sub-second job arrivals.
O-26: Low variability in the number of requested CPU-cores.

Combining data depicted in Figs. 10 and 11, we observe a highly variable job-arrival process (**O-21**). Unlike the Mustang and OpenTrinity traces analyzed by Amvrosiadis et al. in [3], the trace we analyze does feature a clear diurnal pattern in job submissions, depicted in Fig. 10. We observe an office-like daily pattern (**O-22**), with job submissions ramping up in the morning after 9am and lasting until office closing time. This confirms the expectations of the datacenter operational team. However, we also observe job-submissions still occur, until 4 am.

Following the method of Amvrosiadis et al. [3], we classify arrival rates exceeding 10,000 jobs per day as a “high arrival rate”. Fig. 11 shows the maximum number of submitted jobs on a single day is 167,189, and the average rate is above 10,000 (**O-23**). The peak demand of the datacenter exhibits many periods with high, sub-second job-arrival rates (**O-25**); these appear in Fig. 11 as daily peaks larger than 10^5. These translate to resource over-commitment; although the allocation of CPU-cores and RAM is limited using cgroups, other resources such as network and disk I/O are not rate-limited. We draw attention to the need to develop and deploy new, scalable technology for this problem.

We observe significantly fewer ML jobs arrive on average, compared to generic jobs (**O-24**). Fig. 11 depicts this phenomenon. The median number of ML-job arrivals per day is only 320, an order of magnitude lower than the median for all jobs. We link this to the system—users require additional permission to use ML nodes.

Following the approach of Amvrosiadis et al. [3], we compute the coefficient of variation (CoV) of CPU cores requested per user. We observe in Fig. 12 the CoV is at most 2, with a rapid decrease below 1, low values (**O-26**) similar to those observed at Google.

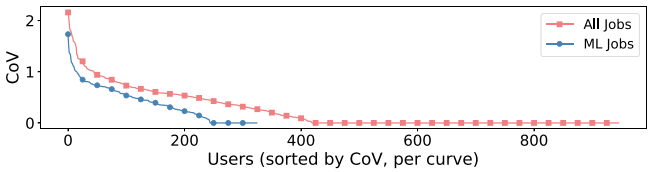


Fig. 12. CoV of the number of CPU cores requested per user. We show a marker for every 25th user.

Table 5
Fraction of jobs per job state.

Ok	Failed	Canceled	Timeout	Out of memory	Requeued	Node failure
91.7%	6.0%	1.2%	1.1%	0.05%	7×10^{-4}%	3×10^{-4}%

5.3. Failure analysis (RQ10, RQ11)

O-27: Most (91.7%) jobs complete successfully.
O-28: Longer-running jobs terminate unsuccessfully more often.
O-29: Unsuccessful jobs consume a significant amount of resources, and at worst they do so until they time out.
O-30: Among all classes of runtimes, ML jobs terminate unsuccessfully more often than other jobs.

Complementing works on machine failures [26,52], we investigate job failures. Relatively few jobs have outcomes other than OK and end unsuccessfully, see Table 5. As we observe, about 91.7% of jobs complete successfully (**O-27**), significantly more than the highest fraction at Google [3] or in production datacenters [53].

In contrast, we observe that longer jobs and jobs that consume more resources tend to fail more frequently (**O-28**), see Fig. 9. For the latter category, for all (ML) jobs, a high fraction of 51.2% (55.8%) of the runtime is spent on non-completed jobs. For long-running jobs, (ML) jobs that do not complete consume 13.8% (51.9%) (**O-29**).

Across all job durations, between 32.3% and 55.8% of the ML jobs complete unsuccessfully; this is more often than all jobs (12.9–51.2%) (**O-30**). We depict the total sum of job runtimes and their fraction per job state. The behavior of longer jobs failing more frequently is mainly due to timeouts, as there is a 5-day limit in the datacenter, as the operators reported. The data shows clearly that larger jobs fail more often and consume more time than smaller jobs.

We have presented an in-depth analysis of several of the metrics listed in the archive we consider. Next, we investigate if other metrics in the dataset can be derived from others.

6. Are just a few metrics enough?

In this section we show that more metrics are needed when analyzing datacenter behavior, and thus also that more metrics should be recorded and shared. Although the dataset includes low-level metrics collected by servers, OS, and applications, we focus in this section on metrics mostly as context-agnostic information, that is, without a structure or ontology that attaches them to specific datacenter components or processes. This allows us to understand whether more metrics can provide new information.

Method overview: Correlations can lead to improvements in system monitoring and find interesting relationships for, e.g., predictions [54]. In particular, we are interested if all metrics in the dataset we consider are necessary or can be obtained from others through correlation, and if these correlations are persistent or workload-dependent. First, we compute all valid correlation-pairs during a day and inspect if pairs which are considered "very strong" by literature are persistent, as these pairs are the most likely candidates to reduce the size of the dataset through derivation and likely most robust. Second, we analyze visually the distribution and correlations of several commonly used high-level node metrics.

Conclusion: A small set of metrics cannot capture the information provided by diverse metrics. This means that if any analysis is to be done in the future on different metrics, past data cannot be leveraged if these metrics are not included. We urge datacenter practitioners to *collect as much fine-grained data as possible for enabling valuable analyses*, and to *open-source such data for the benefit of all*; we analyze some associated overheads in Section 7.1.

6.1. More metrics needed

To observe if metrics are workload dependent, we compute the Spearman and Kendall correlations for all pairs of metrics. Although common, we do not use the Pearson correlation as it is unlikely that metric pairs have a linear relationship. This results in over 14,000 valid correlation pairs per day.[1] Next, we compute per day the number of pairs with a "very strong" monotonic relationship, i.e., with Spearman coefficient $\geq$0.9 [55]. We verify that all p-values of the pairs depicted in the figure are equal to 0, that is, all coefficients are significant.

Fig. 13 depicts the number of such metric-pairs across 50 consecutive days. From this figure, we observe that across all 14,000+ pairs only 40 pairs of low-level metrics show a *strong monotonic relationship* (correlation coefficient $\geq$0.9), for the Spearman correlation coefficient. The same is observed, yet to a lesser extent, when computing the Kendall tau correlation coefficient for all metric pairs. Kendall's tau correlation is sometimes favored due to it being less sensitive to outliers and errors in the data [56]. We observe across the same 50 days that only 12 pairs consistently have a *strong monotonic relationship*, i.e., $\tau \geq 0.9$, whereas several days feature 40+ of such pairs.

This indicates that correlations, even very strong ones, change daily. Because workloads are the most variable aspect of a datacenter, we conjecture correlations are workload-dependent. This suggests metric information should be collected across many metrics, and over long periods of time. Second, this shows, combined with observations from Section 6.2, that we cannot (significantly) reduce the amount of metrics, as many of these metrics cannot be reliably derived nor predicted from others.

6.2. Correlations are not enough

To understand what correlations can reveal about the correlated metric-pairs, we depict in Fig. 14 a correlation matrix of power usage, ambient temperature, number of processes running, amount of memory used, and UNIX load1 as substitute for CPU load. These metrics are coarse, high-level metrics which can be used as indicators for system utilization and are often captured by monitoring systems. For all these metrics, the input dataset has valid data, so we are able to accurately compute all correlations. We only summarize here the results; more details appear in our technical report [47].

The correlation matrix in Fig. 14 includes: (i) normalized histograms on the diagonal, (ii) pair-wise scatter plots and linear regressions in its sub-diagonal elements; (iii) mirrored in the diagonal, the Pearson, Spearman, and Kendall correlations of each pair in (ii). From (i), we observe all metrics except for temperature have a long tail, which makes linear relationships less likely. Indeed, from (ii), we observe that most metric-pairs lack a linear

[1] For some metrics, no valid data are available or all values are the same.

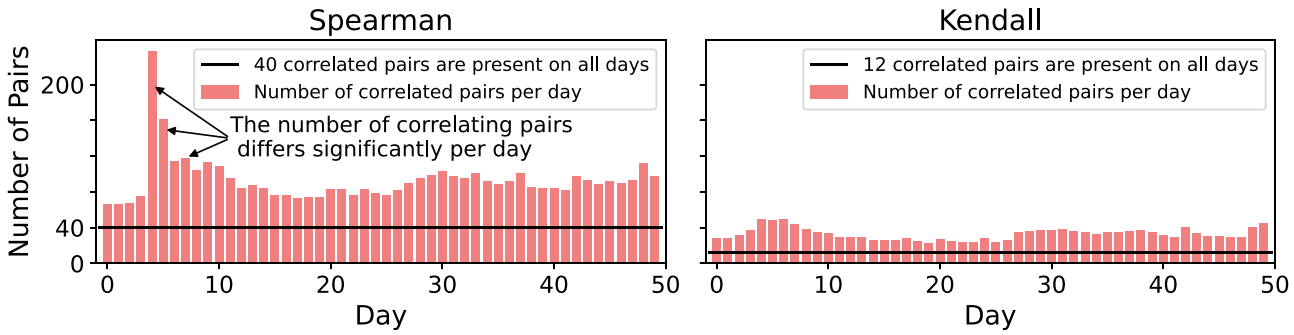


Fig. 13. Number of correlated metric pairs with a Spearman (left) and Kendall (right) coefficient ≥ 0.9 across 50 days. The black line in each figure depicts the number of pairs that are consistently present on all days. Because only 40, respectively 12 pairs are consistently correlated, we need to consider the other hundreds of metrics, taken individually.

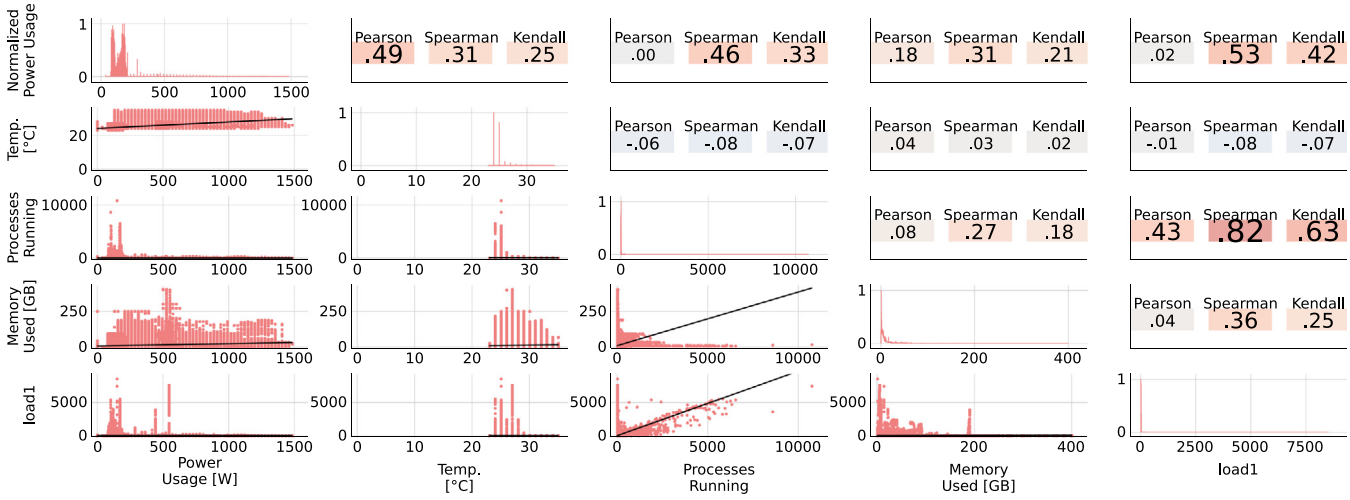


Fig. 14. The correlation matrix of node metrics. See text for an explanation of the sub-plots.

relationship. Exceptions are power usage and temperature, and load1 and (number of) running processes; both are expected from practice.

The plots type (iii) indicate some relationships with moderate, yet sufficiently strong correlations exist [55]. However, most cannot be captured by a linear relationship which the Pearson correlation confirms. (All p-values are $<10^{-13}$, so the results are meaningful.) Interestingly, although the Pearson correlation indicates a moderate correlation for Normalized Power Usage and Temperature, there does not seem to be a linear correlation visually. Thus, we conclude that *most common metrics do not seem to correlate. Additionally, mere correlations between a pair of metrics does not imply that any one of the metrics can be recomputed (linearly) from the other. Thus, we cannot discard these common metrics from the set of metrics to collect; in fact, it advocates for more data.*

7. Implications of our results

For the principle of holistic analysis to gain traction, the community needs to find useful guidelines and applications. Toward this end, but limited in scope, this section presents several examples.

7.1. Actionable insights (RQ13, RQ14)

Computation and Storage Overheads. The amount of storage required for the fine-grained data is non-linear with the number of samples due to compression. Intuitively, storing a 2× larger dataset would require 2× storage. However, with modern storage formats, that leverage compression and columnar formats, this is not the case. Using *snappy* compression (the default codec for *parquet* in Spark), the data representing a 10-min granularity snapshot for the two metrics used in this example requires 32.77 MB of storage. In turn, only 277.56 MB is required to store data with a granularity of 15 s, so increasing the volume of uncompressed data 40× increases the actual storage by only 8.47×. Recent formats such as Zstd (https://facebook.github.io/zstd/) look even more promising in compressing ratio and compression/decompression speed over snappy. Therefore, leveraging modern data storage techniques enables storing high-frequency data with sublinear overheads. To conclude, *higher-frequency metric data incurs both computational and storage overheads, but these seem worthwhile when compared with the benefits they enable.*

Metric Correlations. The analysis we depict in Fig. 13 shows a novel insight. Having so many pairs that correlate infrequently shows evidence that correlations are workload dependent, therefore *as many metrics must be captured as frequently as possible*. Our guideline is to *only eliminate the metrics that are strongly correlated over long periods of time.*

We conclude this section with anecdotal insight from our correlation analysis. We find many metrics that, intuitively, correlate persistently: *load1* with *load15*, *netstat TCP data* with *netstat IP data*, and *swap memory* with *free memory*. By manually inspecting the correlations that are not persistent over time, we find other, more interesting correlations that would be difficult to predict even by experts. Table 6 presents three metrics linking IO and GPU processing, corroborating recent ML benchmarks [57].

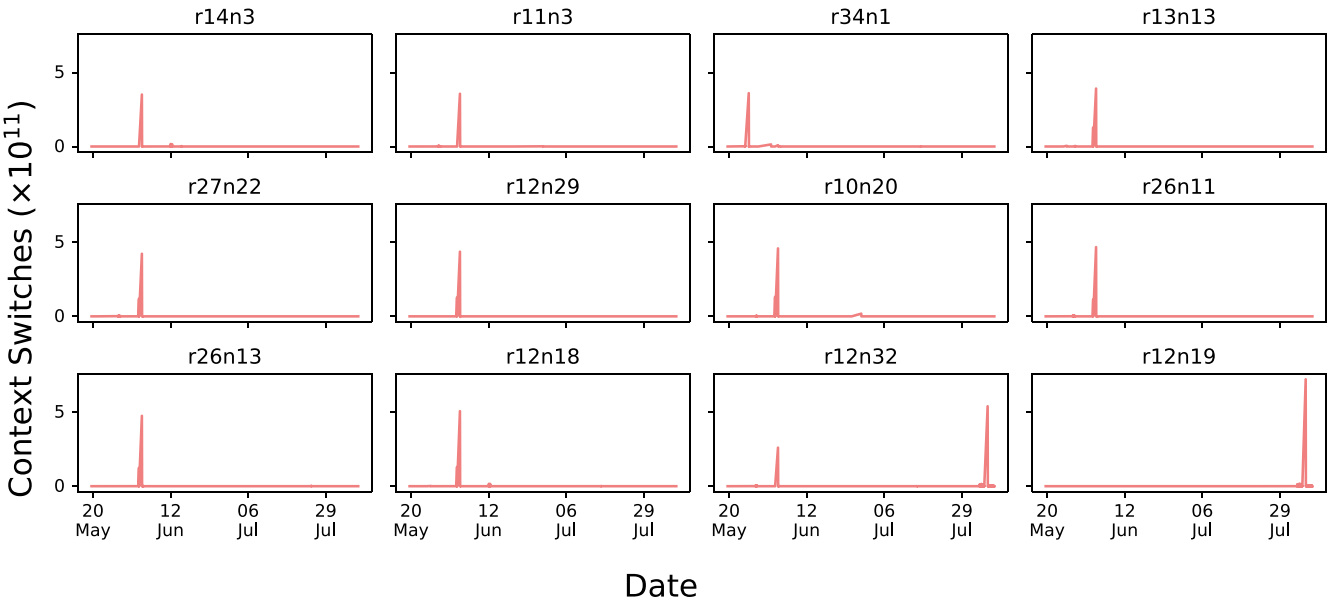


Fig. 15. Number of context switches for the 12 nodes with the highest peak observed. Each figure is labeled with the rack and node number.

Table 6
Correlated metrics identified by analyzing the dataset generated by analysis in Fig. 13.

Metric 1	Metric 2
Server swap memory	GPU temperature
Network receive fifo	GPU temperature
TCP open sockets	GPU temperature

Systematic insight into (multi-)metric correlations remains an open challenge.

7.2. Enable low-level, detailed performance analysis (RQ15)

Another application for fine-grained and low-level data is low-level, detailed performance analysis. Since single-CPU core-speed is stalling [58], while memory and network speeds are continuing to improve, data analysis pipelines are becoming CPU-bound [59]. To alleviate this, parallelism has been introduced via frameworks such as MapReduce and Spark. However, as dataset sizes grow, the number of CPU-cores required to scale with them introduced new issues related to cost and overhead. Accelerators offer an alternative for data processing; examples of these are the use of TPUs for AI and FPGAs and GPUs for more generic purposes. Leveraging long-term, fine-grained operational data can reveal (1) whether the current generation of applications leverages efficiently the available infrastructure, e.g., Software as a Service (SaaS) applications—where the user has no control of—leveraging available accelerators, which can lead to new design and tuning, and (2) whether the current generation of *users* leverages the available resources effectively, i.e., investigating if users of an Infrastructure as a Service (IaaS) environment—where the datacenter operator has less control over what software is being executed—make (efficient) use of the resources available to them. Examples are users not deploying software that can take advantage of accelerators, e.g., artists using CPU-based rendering engines rather than GPU-accelerated engines when working with rendering software, or software developers (not) making use of libraries to leverage available accelerators in their software. These investigations could lead to new designs of incentives for high-performance, energy-efficient infrastructures.

A more recent development is the emergence of system designs that are “CPU-less” [60], which aims to alleviate the significant performance overhead caused by coordination and communication between accelerators and the CPU. An example of such an approach can be found with NVIDIA’s Spark-rapids library [61]. By leveraging the Unified Communication X (UCX) framework, Remote Direct Memory Access (RDMA) becomes available. RDMA enables the capability of GPUs directly transferring memory between GPUs or to host memory, completely bypassing the CPU, avoiding PCI-e bus traffic and context switches, enabling significant speedups. A talk of Robert Evans and Jason Lowe at the Data + AI Summit 2020 elaborates on this architecture [59]. Having available low-level system metrics allows to (1) analyze if RDMA approaches are interesting to explore, and (2) observe and further tune CPU-less use-cases. To illustrate this, Fig. 15 shows the number of context switches for the 12 nodes with the highest observed peak. For 9 of these 12 nodes, the peak occurs at the same time, possibly caused by a single job spanning multiple nodes or the system itself. Investigating such peaks to trace its cause can yield valuable performance, especially when the behavior is observed for a class of jobs and/or originates from the system itself, as context switches are expensive [62].

Wang et al. [63] demonstrate another exemplary use-case of RDMA. Traditional monitoring systems such as Netdata and Prometheus lead to spikes in CPU utilization, resulting in network latency spikes. These latency spikes can be costly for cloud businesses such as Google, Alibaba, etc., where extreme consolidation can lead to high potential for performance interference [7,8]. They introduce a monitoring system based on RDMA. The value of their system is that, by avoiding these CPU spikes, the latency is more predictable and lower, resulting in more revenue and higher customer satisfaction. The data we explore in this work includes statistics on CPU load, cache hits, and networks per protocol. This data enables operators of (large-scale) computing environments such as clusters and datacenters to detect if RDMA can improve performance. Further, having available these low-level, fine-grained metrics allows for analyzing the behavior of such systems and observe and further tune these approaches [64]. As most public datasets and methods do not contain detailed communication traffic, e.g., PCI-e, Ethernet, InfiniBand, etc., next to job-level information, these datasets cannot be used for such purposes, supporting our notion why low-level metrics should be captured.

7.3. Designing and tuning (RQ15)

Our final guideline is to *use fine-grained data for designing and tuning datacenters, from individual chips to full-system procurement.* We support this guideline with qualitative analysis.

Datacenters are often acquiring homogeneous batches of hardware. Often, for datacenters for scientific computing and engineering, nodes pack a large x86 CPU and large amounts of memory. Clusters equipped for HPC and machine learning often also add GPUs and high-speed interconnects. The power envelope of datacenters has constantly increased, and modern large-scale datacenters approach the limits of what our society can leverage in terms of power while being mindful of carbon emissions [1,65]. Others have considered power savings by means of reducing cooling [66], but that is only one example of the many aspects that could be considered. In this paper, we have analyzed many metrics, all with potential impact on how datacenters could be tuned and designed. We posit that using such data for customizing datacenters suited to their user's needs is key for efficiency. Using the resource usage profiles uncovered in this work one could, for example, build machine-learning clusters more efficiently by leveraging lower-power CPUs (e.g., ARM and RISC-V) next to power-hungry GPUs. In GPU-based ML workloads, power-hungry x86 CPUs are underutilized, being mostly used in data pre-processing and data movement. Moreover, as memory usage is low in our traces, for similar workloads the designer does not need to purchase large amounts of RAM. For inadvertent peak-loads, designers could leverage software disaggregation methods [67,68], instead of hardware acquisition.

Uncovering inefficiencies in datacenters by holistic performance analysis approaches can also lead to improved chip design. In the post-Moore era, this is an avenue beginning to be explored by large tech companies and hardware manufacturers. Google pioneered optimizing ML training with TPUs [69]. This trend continues at organizations like Amazon or Nvidia, who are building inference-tailored chips [70,71], or even deep-learning programmable engines [72]. Only with such analysis, practitioners could tackle both performance, power consumption and other important metrics at the same time. Similar trends have already started at the network level, where in-network computing is already a reality [73]. Significantly improving network performance, and releasing pressure from CPUs is something that our data already supports (see Figs. 1 and 2). Already, the analysis we have conducted in this work has helped the datacenter operator improve the design of their next monitoring system.

8. Related work

Datacenter operations: Several articles provide a holistic view of datacenter operations, including job allocation [74], cloud services [75], physical network [76], etc. Different from related work, our article provides a view of the effect of the workload on machine metrics. This complements prior work and aids in understanding the operations within modern datacenters. Other monitoring systems, e.g., GUIDE [26] and DCDB Wintermute [27], focus on different analyses and are mutually complementary with our work.

Characterizations of workloads: There are various articles on the topic of characterizing workloads from Google [77,78], FinTech [3,20], scientific computing environments [3,20,79], etc. Adding to this topic, we demonstrate our workload is unique in terms of properties. Additionally, many of the jobs are from the ML domain, which, combined with the machine metric characterization, provides interesting (and sometimes contrasting) insights.

Characterizations of machine metrics: Different in this work from the body of related work are the many additional and novel analyses (see Section 3.3) and the exemplary implications (see Section 7). Uta et al. [15] provide but do not analyze the dataset used in this work. Much related work focuses on a few, high-level metrics [34,35,80,81]. Like Patel et al. [33], we cover power consumption for CPU-only nodes; differently, our analysis also includes *machines with accelerators*. Whereas Gupta et al. [52] focus on machine failures, we analyze *job* failures.

Metric correlations: Some related work makes use of metric correlations, e.g., finding (virtual) machines executing the same application [82], finding (virtual) machines that correlate in resource utilization [80] to minimize contention, checking for correlations between resources requested in datacenters [35]. Closest is finding metric correlations that hold over longer periods of time [12]; differently, we also give strong evidence that correlations are workload dependent. Unlike prior work, which typically uses a single method to determine correlations, our study uses three correlation methods, which reduce the impact of (incorrect) implicit assumptions. We also make a case for novelty based on scale: we use two orders-of-magnitude more metric-pairs than previous work.

9. Conclusion and future directions

To conquer the ever-increasing complexity of our datacenters, we posit the need for more holistic and diverse analysis of such systems—in contrast to optimizing only for the few metrics we measure now. In this work we propose a method of analysis for datacenter operations using rich datasets.

We applied our method on a public, long-term datacenter trace of unprecedented temporal and spatial granularity. We made over 30 observations, which give detailed, deep insights into the operation of a public scientific infrastructure. Finally, we discussed the implications of our findings on daily operations and on long-term datacenter design.

We envision our work, and similar pioneering efforts, as motivators for a community-driven approach embracing deep, holistic analysis.

CRediT authorship contribution statement

Laurens Versluis: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Project administration. **Mehmet Cetin:** Software, Validation, Data curation, Writing – original draft, Visualization. **Caspar Greeven:** Software, Visualization, Resources. **Kristian Laursen:** Software, Investigation. **Damian Podareanu:** Investigation, Resources, Validation. **Valeriu Codreanu:** Methodology, Investigation, Resources, Validation. **Alexandru Uta:** Conceptualization, Methodology, Software, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision. **Alexandru Iosup:** Conceptualization, Methodology, Validation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Alexandru Iosup reports financial support was provided by Nederlandse Organisatie voor Wetenschappelijk Onderzoek Utrecht. Damian Podareanu reports a relationship with Cooperatie SURF UA that includes: employment. Caspar Greeven reports a relationship with Cooperatie SURF UA that includes: employment. Valeriu Codreanu reports a relationship with Cooperatie SURF UA that includes: employment. Co-author (Kristian Laursen) completed an internship at Surf, Amsterdam. The outcome of his work (the dataset) is the foundation of the analysis in our work.

Data availability

The article contains links to software and data hosted on GitHub and Zenodo, respectively.

Acknowledgments

This work was partially funded by the Dutch National Science Foundation NWO, The Netherlands through Veni grant VI.202.195 supporting Alexandru Uta.

References

[1] Dutch Data Center Association, State of the Dutch data centers, 2020, https://www.dutchdatacenters.nl/en/publications/state-of-the-dutch-data-centers-2020/.
[2] Feitelson, et al., Experience with using the parallel workloads archive, JDPC 74 (2014).
[3] Amvrosiadis, et al., On the diversity of cluster workloads and its impact on research results, in: ATC, 2018.
[4] Li, et al., Ease.ML: Towards multi-tenant resource sharing for machine learning workloads, Proc. VLDB Endow. 11 (2018).
[5] Jeon, et al., Analysis of large-scale multi-tenant GPU clusters for DNN training workloads, in: ATC, 2019.
[6] Lockwood, et al., A year in the life of a parallel file system, in: SC, 2018.
[7] Maricq, et al., Taming performance variability, in: OSDI, 2018.
[8] Uta, et al., Is big data performance reproducible in modern cloud networks? in: NSDI, 2020.
[9] Gunawi, et al., Fail-slow at scale: Evidence of hardware performance faults in large production systems, TOS 14 (2018).
[10] Verma, et al., Large-scale cluster management at Google with Borg, in: EuroSys, 2015.
[11] Tirmazi, et al., Borg: the next generation, in: EuroSys, 2020.
[12] Cortez, et al., Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms, in: SOSP, 2017.
[13] Shahrad, et al., Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider, in: ATC, 2020.
[14] Laursen, et al., Beneath the SURFace: An MRI-like view into the life of a 21st century datacenter, 2020, Zenodo dataset, https://zenodo.org/record/3878143.
[15] Uta, et al., Beneath the SURFace: An MRI-like view into the life of a 21st-century datacenter, USENIX Login 45 (2020).
[16] Thain, et al., Distributed computing in practice: the condor experience, CCPE 17 (2005).
[17] Verma, et al., Two sides of a coin: Optimizing the schedule of mapreduce jobs to minimize their makespan and improve cluster performance, in: MASCOTS, 2012.
[18] Dean, Barroso, The tail at scale, CACM 56 (2013).
[19] Ghodsi, et al., Dominant resource fairness: Fair allocation of multiple resource types, in: NSDI, Vol. 11, 2011.
[20] Versluis, et al., The workflow trace archive: Open-access data from public and private computing infrastructures, TPDS 31 (2020).
[21] Bouwers, et al., Getting what you measure, CACM 55 (2012).
[22] Ousterhout, Always measure one level deeper, CACM 61 (2018).
[23] Dayarathna, et al., Data center energy consumption modeling: A survey, COMST 18 (2015).
[24] Goiri, et al., Greenslot: scheduling energy consumption in green datacenters, in: SC, 2011.
[25] Bourassa, et al., Operational data analytics: Optimizing the national energy research scientific computing center cooling systems, in: ICPP Workshop, 2019.
[26] Vazhkudai, et al., GUIDE: a scalable information directory service to collect, federate, and analyze logs for operational insights into a leadership HPC facility, in: SC, 2017.
[27] Netti, et al., DCDB wintermute: Enabling online and holistic operational data analytics on HPC systems, in: HPDC, 2020.
[28] Silva, et al., Community resources for enabling research in distributed scientific workflows, in: E-Science, 2014.
[29] Legrand, et al., Monitoring and control of large systems with monalisa, Commun. ACM 52 (9) (2009) 49–55.
[30] Sigelman, et al., Dapper, a large-scale distributed systems tracing infrastructure, 2010.
[31] Zhao, et al., Non-intrusive performance profiling for entire software stacks based on the flow reconstruction principle, in: OSDI, 2016.
[32] Iosup, et al., The grid workloads archive, FGCS 24 (2008).
[33] Patel, et al., What does power consumption behavior of HPC jobs reveal? : Demystifying, quantifying, and predicting power consumption characteristics, in: IPDPS, 2020.
[34] Patel, et al., Job characteristics on large-scale systems: long-term analysis, quantification, and implications, in: SC, 2020.
[35] Shen, et al., Statistical characterization of business-critical workloads hosted in cloud datacenters, in: CCGrid, 2015.
[36] Wilkinson, et al., The FAIR guiding principles for scientific data management and stewardship, Nat. SciData 3 (2016).
[37] Dutch Government, Ontwikkeling COVID-19 in grafieken, 2020, https://www.rivm.nl/coronavirus-covid-19/grafieken.
[38] Xiong, et al., VPerfGuard: an automated model-driven framework for application performance diagnosis in consolidated cloud environments, in: ICPE, 2013, pp. 271–282.
[39] Ibidunmoye, Hernández-Rodriguez, Elmroth, Performance anomaly detection and bottleneck identification, ACM Comput. Surv. 48 (1) (2015) 4:1–35.
[40] Ghiasvand, Ciorba, Anomaly detection in high performance computers: A vicinity perspective, in: ISPDC, 2019, pp. 112–120.
[41] Chapin, Cirne, Feitelson, et al., Benchmarks and standards for the evaluation of parallel job schedulers, in: JSSPP, 1999, pp. 67–90.
[42] Gregg, Linux load averages: Solving the mystery, 2017, http://www.brendangregg.com/blog/2017-08-08/linux-load-averages.html.
[43] Intel, Inc, The intel® xeon® silver 4110 CPU, 2017, https://ark.intel.com/content/www/us/en/ark/products/123547/intel-xeon-silver-4110-processor-11m-cache-2-10-ghz.html.
[44] NVIDIA, Geforce GTX 1080 Ti graphics cards | NVIDIA geforce, 2020, https://www.nvidia.com/en-sg/geforce/products/10series/geforce-gtx-1080-ti/ Accessed: 2021-04-08.
[45] NVIDIA, The world's most powerful graphics card | NVIDIA TITAN V, 2020, https://www.nvidia.com/en-us/titan/titan-v/, Accessed: 2021-04-08.
[46] NVIDIA, TITAN RTX ultimate PC graphics card with turing | NVIDIA, 2020, https://www.nvidia.com/en-gb/deep-learning-ai/products/titan-rtx/, Accessed: 2021-04-08.
[47] Versluis, et al., A Holistic Approach for Datacenter Analysis – Extended, Technical Report, 2021, CoRR.
[48] Peng, et al., On the memory underutilization: Exploring disaggregated memory on HPC systems, in: SBAC-PAD, 2020.
[49] Duy, et al., Performance evaluation of a green scheduling algorithm for energy savings in cloud computing, in: IPDPSW, 2010.
[50] Toms Hardware, Nvidia's RTX 3000 power supply requirements amp up PSU shortage concerns, 2020, https://www.tomshardware.com/news/nvidias-rtx-3000-power-supply-requirements-PSU-shortage-2020.
[51] Zhang, et al., Comparison and evaluation of air cooling and water cooling in resource consumption and economic performance, Energy (2018).
[52] Gupta, et al., Failures in large scale systems: long-term measurement, analysis, and implications, in: SC, 2017.
[53] Javadi, et al., The failure trace archive: Enabling the comparison of failure measurements and models of distributed systems, JPDC 73 (8) (2013) 1208–1223.
[54] Xiao, et al., Using Spearman's correlation coefficients for exploratory data analysis on big dataset, CCPE 28 (2016).
[55] Schober, et al., Correlation coefficients: appropriate use and interpretation, Anesth. Analg. 126 (2018).
[56] Pluviophile, Pearson vs Spearman vs Kendall, 2019, https://datascience.stackexchange.com/a/64261, Accessed: 2022-03-22.
[57] Jansen, et al., DDLBench: Towards a scalable benchmarking infrastructure for distributed deep learning, in: DLS At ICS, 2020, pp. 31–39.
[58] P.E. Ross, Why cpu frequency stalled, IEEE Spectr. (2008).
[59] R. Evans, J. Lowe, Deep dive into GPU support in apache spark 3.x, 2020, https://www.databricks.com/session_na20/deep-dive-into-gpu-support-in-apache-spark-3-x.
[60] M.S. Brunella, et al., Hyperion: A case for unified, self-hosting, zero-CPU data-processing units (DPUs), 2022, arXiv preprint arXiv:2205.08882.
[61] NVIDIA, Accelerating apache spark 3.0 with GPUs and RAPIDS, 2020, https://developer.nvidia.com/blog/accelerating-apache-spark-3-0-with-gpus-and-rapids/.
[62] C. Li, et al., Quantifying the cost of context switch, in: Experimental Computer Science Workshop, 2007.
[63] Z. Wang, et al., Zero overhead monitoring for cloud-native infrastructure using {*RDMA*}, in: USENIX ATC, 2022.
[64] A. Kalia, et al., Design guidelines for high performance RDMA systems, in: USENIX ATC, 2016.
[65] Koomey, et al., Recalibrating global data center energy-use estimates, Science 367 (2020).
[66] El-Sayed, et al., Temperature management in data centers: why some (might) like it hot, in: SIGMETRICS, 2012.
[67] Gu, et al., Efficient memory disaggregation with infiniswap, in: NSDI, 2017.
[68] Uta, et al., Towards resource disaggregation—Memory scavenging for scientific workloads, in: CLUSTER, 2016.
[69] Jouppi, et al., Motivation for and evaluation of the first tensor processing unit, IEEE Micro 38 (3) (2018) 10–19.

[70] Amazon, Inc., AWS inferentia: High performance machine learning inference chip, custom designed by AWS, 2018-2021, https://aws.amazon.com/machine-learning/inferentia/.
[71] NVIDIA, NVIDIA deep learning accelerator (NVDLA), 2017-2021, http://nvdla.org/.
[72] XILINX, The xilinx® deep learning processor unit (DPU), 2020-2021, https://www.xilinx.com/products/intellectual-property/dpu.html.
[73] Stephens, et al., Your programmable nic should be a programmable switch, in: HotNets, 2018.
[74] Andreadis, et al., A reference architecture for datacenter scheduling: design, validation, and experiments, in: SC, 2018.
[75] Liu, et al., NIST Cloud Computing Reference Architecture, Vol. 500, NIST Special Publication, 2011.
[76] Lam, et al., Fiber optic communication technologies: What's needed for datacenter network operations, IEEE Commun. Mag. 48 (2010).
[77] Rosà, et al., Predicting and mitigating jobs failures in big data clusters, in: CCGrid, 2015.
[78] Rosà, et al., Failure analysis and prediction for big-data systems, TSC 10 (2017).
[79] Silva, et al., Workflowhub: Community framework for enabling scientific workflow research and development, in: WORKS, 2020.
[80] Kim, et al., Correlation-aware virtual machine allocation for energy-efficient datacenters, in: DATE, 2013.
[81] Birke, et al., Data centers in the cloud: A large scale performance study, in: CLOUD, 2012.
[82] Canali, Lancellotti, Identifying communication patterns between virtual machines in software-defined data centers, SIGMETRICS 44 (2017).

Laurens Versluis received his B.Sc. and M.Sc. degrees in computer science (Distributed Systems, Software Technology) from the Technical University of Delft, The Netherlands.

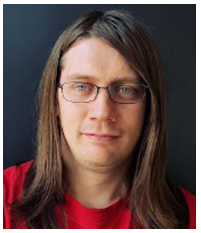
Currently, he is a Ph.D. student with the Massiving Computer Systems Group of the Department of Computer Science, Faculty of Sciences, VU Amsterdam and a data engineer at ASML, Veldhoven.

His research interests include cloud computing, distributed systems, scheduling, complex workflows, and high-performance computing.

Mehmet Cetin obtained a B.Sc. in computer science at Vrije Universiteit Amsterdam.

He has completed an honors research project on characterizing datacenter operations. His professional interests span the field of computer science with a focus on datacenter technologies and distributed systems.

Caspar Greeven earned his Master of Science degree at the Universiteit van Amsterdam. He currently works as Young Talent at Surf where he investigates how machine learning can be used to extract useful and actionable information from network-related information.



Kristian Laursen earned his Bachelor of Science degree at the Vrije Universiteit Amsterdam. During his bachelor thesis, he interned at Surf where he investigated how to collect and investigate machine performance data. He currently works as software developer at Adyen, Amsterdam.

Damian Podareanu has a bachelor of science in mathematics and computer science from the University of Bucharest, a master of science in Advanced Computer Architectures at the Polytechnic University of Bucharest, and a master of science in Artificial Intelligence at the University of Groningen.

He joined Surf in 2016 as a HPC consultant and currently is a senior machine learning consultant and team lead.

Valeriu Codreanu has a Ph.D. in Electrical Engineering with a thesis on developing a novel multi-threaded computer architecture.

Afterwards he did a postdoc on GPU computing at the University of Groningen, where he helped develop a CPU-to-GPU compiler called GPSME and another postdoc on Embedded Computing at Technical University of Eindhoven focusing on real-time embedded systems.

Valeriu joined Surf in 2014 as an HPC consultant. He was the PI of the GPU Research Center and also of the current Intel Parallel Computing Center.

His interests lie in efficient computing, scaling, and in the application area of deep learning.

Alexandru Uta is an assistant professor in the computer systems group at LIACS, Leiden University.

He received his Ph.D. in 2017 from VU Amsterdam on topics related to distributed storage systems for scientific workloads.

His current research interests are in taming large-scale infrastructure: from designing reproducible experiments, to understanding and evaluating performance, as well as designing efficient large-scale computer systems.

His research is funded through industry and academic grants and was recently awarded the NWO Veni (early career) award.

Alexandru Iosup is tenured full Professor and University Research Chair with the Vrije Universiteit Amsterdam, the Netherlands. He is also Chair of the SPEC Research Cloud Group. He received a Ph.D. from TU Delft, the Netherlands (2009) and an M.Sc. from Politehnica University of Bucharest, Romania (2004), both in computer science. He has received numerous awards and nominations. Topics include cloud computing and big data, with applications in big science, big business, online gaming, and (upcoming) massivized education. His work is funded by a combination of prestigious personal grants, generous industry gifts and collaborations, and EU and national projects.