Understanding the Landscape of Ampere GPU Memory Errors

Zhu Zhu George Mason University USA zzhu22@gmu.edu Yu Sun George Mason University USA ysun23@gmu.edu Dhatri Parakal University of Illinois Urbana-Champaign USA dhatrip2@illinois.edu

Bo Fang
Pacific Northwest National
Laboratory / University of Texas,
Arlington
USA
bo.fang@pnnl.gov

Steven Farrell
National Energy Research Scientific
Computing Center
USA
sfarrell@lbl.gov

Gregory H. Bauer
National Center for Supercomputing
Applications
USA
gbauer@illinois.edu

Brett Bode
National Center for Supercomputing
Applications
USA
brett@illinois.edu

brett@illinois.edu
William Gropp
Jniversity of Illinois

University of Illinois
Urbana-Champaign
USA
wgropp@illinois.edu

Ian T. Foster
University of Chicago / Argonne
National Laboratory
USA
foster@cs.uchicago.edu

Zhao Zhang Rutgers University USA zhao.zhang@rutgers.edu Michael E. Papka Argonne National Laboratory / University of Illinois Chicago USA papka@anl.gov

Lishan Yang George Mason University USA lyang28@gmu.edu

Abstract

Graphics Processing Units (GPUs) have become a de facto solution for accelerating high-performance computing (HPC) applications. Understanding their memory error behavior is an essential step toward achieving efficient and reliable HPC systems. In this work, we present a large-scale cross-supercomputer study to characterize GPU memory reliability, covering three supercomputers – Delta, Polaris, and Perlmutter - all equipped with NVIDIA A100 GPUs. We examine error logs spanning 67.77 million GPU device-hours across 10,693 GPUs. We compare error rates and mean-time-betweenerrors (MTBE) and highlight both shared and distinct error characteristics among these three systems. Based on these observations and analyses, we discuss the implications and lessons learned, focusing on the reliable operation of supercomputers, the choice of checkpointing interval, and the comparison of reliability characteristics with those of previous-generation GPUs. Our characterization study provides valuable insights into fault-tolerant HPC system design and operation, enabling more efficient execution of HPC applications.

1 Introduction

Graphics Processing Units (GPUs) are widely deployed in modern supercomputers to accelerate high-performance computing (HPC) applications. These HPC systems operate at massive scales, with tens of thousands of GPUs in a single supercomputer. A range of workloads is supported, from conventional HPC workloads such as

fluid simulation [1, 2] and molecular dynamics [3, 4], to emerging tasks including transformer-based large language models (LLMs) [5, 6] and scientific foundation models [7]. However, the high-density transistor arrays in GPUs increase their susceptibility to hardware faults and memory errors due to cosmic radiation [8] and shrinking transistors [9], leading to silent data corruptions, exceptions, and crashes [10–15]. Even just one single memory error on one GPU can jeopardize the entire application execution, which involve up to $O(10^5)$ of GPUs over weeks to months [11, 16]. For example, during Meta's OPT-175B training, hardware errors are reported on 18 out of 55 days [11]. These errors can delay the training process and, in the worst case, compromise model convergence [13, 14]. Therefore, understanding error behavior in large-scale systems is the key first step towards designing error mitigation techniques and avoiding the aforementioned severe consequences of GPU errors.

Limitation of state-of-art approaches. Existing supercomputer reliability analyses do not meet the practical needs for recent large foundation model (LFM) training and scientific applications. Many studies focus on CPUs [17–20] and DRAMs (Dynamic Random Access Memories) [21]. GPU error studies primarily reveal the error characteristics of earlier GPU generations including NVIDIA K20X [22–27] and V100 [28]. Moreover, past GPU error characterization studies typically focus on a single supercomputer [24, 26, 28, 29]. This lack of a cross-supercomputer perspective may result in observations biased by the studied system and thus not generalizable to other HPC systems.

Motivation and Challenges. To address the growing computing demands for the new LFM paradigm, academia and industrial stakeholders have built new supercomputers with massive numbers of GPUs. This widespread deployment and growing popularity highlights the necessity of revisiting GPU error characteristics in production supercomputers, with the goal of deriving effective operation suggestions for HPC users and system administrators to preserve high execution efficiency and machine utilization.

As we demonstrate in the paper, a *cross-supercomputer* perspective is essential in such error behavior study. While clusters with the same GPU generation have common reliability traits, their unique hardware settings and workloads may introduce unknown biases. Performing a cross-supercomputer study helps remove these biases to uncover the natural behavior of GPU errors. However, such study is challenging: 1) the differences in supercomputer configurations complicate the analysis; 2) the heterogeneity of error data across systems renders data discovery and fair comparisons difficult.

Key insights and contributions. We perform a cross-institute collaboration on a comprehensive error characterization study across 10,693 NVIDIA Ampere GPUs in three heavily utilized, inproduction supercomputers: Delta [30], Polaris [31], and Perlmutter [32]. We monitor and record ECC-reported errors of GPUs to form the error log, i.e., a GPU memory error dataset, resulting in a total of 67.77 million GPU device-hours. Our key observations are summarized as follows:

- The observed error rates on different clusters can vary by over three orders of magnitude. Trusting the results from one single cluster may introduce significant biases in error behavior characterization.
- Errors are not uniformly distributed over time. Burstiness is generally observed in all three clusters, while their severeness varies.
- Both bursty error patterns and supercomputer scale can affect the characteristics of error interarrival times. Using a single number, the mean-time-between-error (MTBE), may introduce bias. Instead, the distribution of interarrival times should be considered.
- There is no observable periodicity of errors.
- For environmental factors such as temperature, power, and GPU utilization, we do not observe any strong correlation with errors
- Erroneous nodes vary over time.

Based on these observations and analyses, we further discuss the implications and lessons learned from this study, focusing on the reliable operation of supercomputers, the choice of checkpointing interval, and the comparison of reliability characteristics with those of previous-generation GPUs. Our key take-away messages are:

- NVIDIA V100 and A100 GPUs both use HBM2 and have similar memory error characteristics.
- Coarser-level error monitoring does not necessarily lead to much information loss, yet monitoring errors at a finer level enables faster responses to errors.
- Dynamic node monitoring and erroneous GPU prediction strategies are needed for the efficient and reliable operation of supercomputers.

 Dynamic checkpointing is suggested to accommodate the bursty error patterns observed in all three supercomputers.
 GPU error monitoring could be used to guide the frequency of dynamic checkpointing.

2 Background and Data Collection Method

We first describe the architecture and organization of the studied supercomputers and their constituent GPUs. Then, we discuss our data collection method and the basic information contained in our datasets. Last but not least, we discuss the scope and limitations of this work.

2.1 Supercomputer Organizations

We focus on three highly utilized, in-production supercomputers in this error characterization study. Below we briefly describe the architecture of each supercomputer.

Delta [30] is a supercomputer designed by Hewlett Packard Enterprise (HPE) and National Center for Supercomputing Applications (NCSA). There are three types of NVIDIA GPU nodes, with 849 GPUs in total: Delta has 100 four-way NVIDIA A40 GPU compute nodes, each equipped with a single 2.45 GHz AMD Milan 64-core CPU. The four A40 GPUs are connected via PCIe. Another 100 four-way GPU compute nodes have NVIDIA A100s connected via NVLink, with all other configurations identical to the A40 nodes. There also exist six eight-way NVIDIA A100 GPU compute nodes, each with two 2.45 GHz AMD Milan 64-core CPU sockets. The eight A100 GPUs are connected via NVLink.

Polaris [31] is a 560-node HPE Apollo 6500 Gen 10+ based system operated by Argonne Leadership Computing Facility (ALCF). There are 40 racks organized in three rows with 16, 12, and 12 racks, respectively. In each rack, there are seven chassis, each containing two nodes. Each node is equipped with a single 2.8 GHz AMD EPYC Milan 7543P 32-core CPU and four NVIDIA A100 GPUs in HGX platform connected via NVLink, two local 1.6TB of SSDs in RAID0 and two Slingshot 11 network adapters. Polaris has 2240 GPUs in total.

Perlmutter [32] is a HPE Cray EX supercomputer with AMD EPYC CPUs and NVIDIA A100 GPUs operated by the National Energy Research Scientific Computing Center (NERSC). Initially, Perlmutter has 14 GPU compute cabinets in total, each segmented into eight chassis. Each chassis contains eight compute blades, each with two nodes. There are 1536 NVIDIA A100 GPUs (40G) nodes and 256 NVIDIA A100 GPUs (80G) nodes. Each node contains a single AMD EPYC Milan 7763 64-core CPU and four NVIDIA A100 GPUs connected via PCIe, along with four HPE Slingshot 11 NICs. More GPUs are added to the cluster and at the time of error log collection, Perlmutter has 7604 GPUs in total.

2.2 GPU Architecture

All three supercomputers are equipped with NVIDIA Ampere GPUs which are released in 2020. All of the GPUs in Perlmutter and Polaris are NVIDIA A100, while Delta has 400 A40 GPUs and 449 A100 GPUs.

NVIDIA A100 [33]. The NVIDIA Ampere architecture A100 GPU contains seven Graphics Processing Clusters (GPCs), each with seven or eight Texture Processing Clusters (TPCs), and each TPC has two Streaming Multiprocessors (SMs), which is a total of 108 SMs sharing the 40 MB L2 cache. Each SM contains 64 FP32 CUDA Cores and four Tensor Cores, resulting in a total of 6912 FP32 CUDA Cores and 432 Tensor Cores per GPU. The A100 GPU includes 40 GB of fast HBM2 DRAM memory on its SXM4-style circuit board, which is organized as five active HBM2 stacks with eight memory dies per stack. The A100 HBM2 memory subsystem supports single-error correction double-error detection (SECDED) error-correcting code (ECC) to protect data. Other key memory structures in A100 are also protected by SECDED ECC, including aches and register files.

NVIDIA A40 [33]. The NVIDIA Ampere architecture A40 GPU contains 84 SMs, each containing 128 CUDA Cores, four Tensor Cores, and one RT Core, which is 10,752 FP32 CUDA Cores, 432 Tensor Cores, and 84 RT Cores per GPU. The A40 GPU includes 48 GB of GDDR6 DRAM memory with ECC protection.

2.3 Data Collection Method and Datasets

There are various types of GPU-related errors, to name a few, Dynamic Random Access Memory (DRAM) Single-Bit Errors (SBEs) and Double-Bit Errors (DBEs) reported by Error Correction Codes (ECC), NVLink errors, and Dynamic Page Retirement errors. Our focus is primarily on memory errors in the DRAM identified and reported by ECC. We use Data Center GPU Manager (DCGM) [34], a suite of tools to manage and monitor GPUs developed by NVIDIA in data centers to monitor SBEs and DBEs in GPU memory. DCGM regularly updates the aggregated counts of ECC-reported SBEs and DBEs for each GPU, i.e., tracks the historical records of error count. New errors are identified by observing increases in this error count. The collected dates and frequency of the error logs studied in this work are summarized in Table 1.

It is essential to distinguish between two terms: error occurrence event and error count. In one error occurrence event where there is an increased error count, multiple (single- or double-bit) errors may be observed in this single error occurrence event. Distinguishing these two terms is necessary, especially when dealing with DBEs that are detectable but uncorrectable. When a DBE event happens, upon the detection of DBE(s), ECC would throw an exception and trigger the termination of the GPU application, irrespective of the quantity of DBEs identified. Therefore, while analyzing the number of detected DBEs can shed light on the severity of such errors, the number of error occurrence events affects the overall cost of addressing DBEs. The interarrival time between error occurrence events can be averaged to calculate the Mean-Time-Between-Errors (MTBE), an important metric when accessing system resilience.

Delta Cluster Error Dataset. This dataset has rich information covering a period of more than one year with 849 GPUs, which equates to over 7.32 million GPU device-hours of data. Each recorded error occurrence event includes event type (i.e., SBE or DBE), timestamp, associated node, and GPU involved in the event. Note that within Delta, there are 400 NVIDIA A40 GPUs and 449 NVIDIA A100 GPUs, but we do not have the exact mapping of the GPU IDs to their GPU type. We acknowledge this as a limitation of our current study.

Polaris Cluster Error Dataset. We record and analyze Polaris errors across 259 days with 2240 GPUs in the cluster. The missing data from 12/15/2023 to 06/30/2024 is due to lack of access to the database. This results in a total of more than 14.66 million GPU device-hours of data. On Polaris, only DBEs are recorded. There is no information about SBEs.

Perlmutter Cluster Error Dataset. The Perlmutter dataset contains the GMU memory error information across 326 days with 7604 GPUs, resulting in a total of more than 45.79 million GPU device-hours of data. Several months of data are permanently lost due to database failures. Of the three studied clusters, Perlmutter is the largest.

2.4 Limitations and Scope

Despite our thorough data collection and analysis efforts, our study is subject to certain limitations. In this section, we discuss these constraints and clarify the scope of this study.

Firstly, we only have error information on the recorded SBE and DBE errors, thus this study exclusively focuses on the error occurrences in supercomputers. Due to privacy concerns, the workload and job scheduling information is not available. Hence, the correlation between hardware failures and software applications cannot be studied in this work. We partially compensate this limitation by analyzing the correlation of errors with GPU power consumption, temperature, and utilization, which can reflect the workload patterns. Limited data availability restricts our capacity to explore further insights into the systems. SBEs are not recorded on Polaris, which limits the SBE behavior study to Delta and Perlmutter.

Moreover, due to the limited time of data collection, it is possible that our conclusions might not fully capture the behavior of GPU errors outside this period. There are 4 days of missing data in Delta, due to possible system failure or maintenance. For Polaris, we only have access to the 259 days listed in Table 1. For Pulmutter, there are several months of data missing due to database storage error. Despite the missing data, each cluster still contains several hundred days of logged data, totaling 67.77 million GPU device-hours. Therefore, this study holds statistical significance.

We focus on NVIDIA Ampere GPUs, because they are still one of the main accelerators used in top-100 supercomputers [35] and the three supercomputers still have over 50% utilization. Although NVIDIA Hopper GPUs are becoming popular in supercomputers, we do not consider them, as they are still in its early stage of deployment where the utilization is low (around 20%) [29]. We do not have access to obtain the error logs of Hopper GPUs.

The primary focus of this work is error behavior characterization. We aim to derive insightful observations and take-away messages to share with the community. The prediction and mitigation of errors fall outside the scope of this study and are subject to future work.

3 Cross-Supercomputer Error Characterization

In this section, we present a detailed characterization study of GPU memory errors across three supercomputers, all equipped with NVIDIA Ampere GPUs. We focus on the following aspects:

(1) **Overall error behavior** (Section 3.1). We start with presenting an overview of memory error severity through an

Cluster	Delta	Polaris	Perlmutter
Node Count	207	560	1901
GPU Count	849	2240	7604
Log Collection Dates	12/16/2022 - 01/07/2024	10/01/2023 - 12/14/2023 07/01/2024- 12/31/2024	07/01/2023 - 09/30/2023 11/01/2023 - 12/20/2023 08/01/2024 - 01/31/2025
Log Length	388 days	259 days	326 days
Log Frequency	Every minute	Every 4 seconds	Every hour
GPU Hours	7.32 million	14.66 million	45.79 million

Table 1: Basic characteristics of the three supercomputers studied in this work.

analysis of overall error rates and daily error occurrences. The observed error rates on different clusters can vary by over three orders of magnitude and we observe burty errors in all three clusters.

- (2) Interarrival Time of Errors (Section 3.2). We quantify the mean-time-between-errors (MTBE) of clusters to assess the frequency of users encountering such errors. We analyze the distribution of error interarrival times to provide further insights into the burstiness of errors. Our in-depth study reveals that both bursty error patterns and supercomputer scale can affect the characteristics of error interarrival times.
- (3) Correlation with environmental factors (Section 3.3). We analyze the correlation between potential contributing factors including temperature, power, GPU utilization, and the cooling mechanisms. For most environmental factors, we observe weak correlations with errors. No strong correlation is observed.
- (4) Spatial and temporal analysis (Section 3.4). We analyze the spatial and temporal characteristics of errors from a supercomputer management and maintenance perspective to identify GPUs that may pose reliability concerns. Our analysis reveals that GPU memory errors exhibit spatial and temporal correlation, and the set of erroneous GPUs changes over time.

3.1 Overview of Errors

The overall numbers of SBEs and DBEs observed in the three clusters during the study period are shown in Table 2. The three clusters are presented in ascending order of scale: Delta, Polaris, and Perlmutter, with 849, 2240, and 7604 GPUs, respectively. Comparing the error rates, Perlmutter exhibits a higher SBE rate (5.34× higher than that of Delta). The DBE rate of Polaris is notably higher than on the other two clusters: 2555.56× higher than that of Delta, and 8.41× higher than that of Perlmutter. In general, Delta is the most reliable one. The difference highlights the necessity of a cross-supercomputer study: analyzing one cluster can introduce overestimation or under-estimation of the error severity.

Observation 1: Observed error rates on different clusters can vary by over three orders of magnitude. Delta is more reliable than Polaris and Perlmutter supercomputers.

In addition to the error rate numbers, we present the daily error counts for the three clusters in Figure 1. Instances of bursty errors are observed in all three clusters, for example, 112,656 SBEs on 08/14/2023 in Delta, 39,144 DBEs on 10/26/2023 in Polaris, 1,405,332 SBEs and 08/06/2023 in Perlmutter. The existence of bursty errors is also observed in another GPU study of the Titan supercomputer [25].

For a comprehensive understanding of these bursty error patterns, we calculate and plot the empirical cumulative distribution functions (CDFs) of SBEs in the Delta and Perlmutter clusters, see Figure 2. In Delta, 70.88% of days experience no SBEs, indicating that the majority of days are SBE-free. 17.78% of the monitored days in Delta have less than 10 daily errors. Additionally, there are outliers: 1.55% of the days experience more than 1000 SBEs, highlighting the bursty nature of error occurrences. The highest SBE count is recorded on August 14, 2023, with 112,656 SBEs. In Perlmutter, 27.61% of days are SBE-free, which is much less tha the percentage of SBE-free days in Delta 70.88%. The majority, 41.10% of days, have less than 100 SBEs. The number of days exceeding 1000 SBEs in Perlmutter is 19.02%, which is much higher than Delta. The largest daily SBE count in Perlmutter is 1,405,332. The high SBE days observed in these clusters confirm the existence of bursty errors. Moreover, these two distinct daily SBE count distributions emphasize the importance of our cross-supercomputer perspective to quantitatively understand potential biases in singlesupercomputer studies and identify common error characteristics

Similar bursty error patterns of DBEs are observed in Polaris and Perlmutter, as shown in Figure 3. The DBE-free days are 66.02% and 84.97% for Polaris and Perlmutter, respectively. 4.55% of days in Polaris and 9.32% in Perlmutter experience bursty DBEs (more than 1000 errors per day).

Observation 2: Errors are not uniformly distributed over time. Burstiness is generally observed in all three clusters, while their severeness varies.

Note that while the number of errors and error events are a lot, the actual number of GPUs suffering from errors is not huge, as we show in Table 2. Around 4.70% of the GPUs in both Delta and Perlmutter suffer from SBEs. In all three clusters, DBE-occuring GPUs are less than 0.53%. Recall that DBEs are detected but cannot

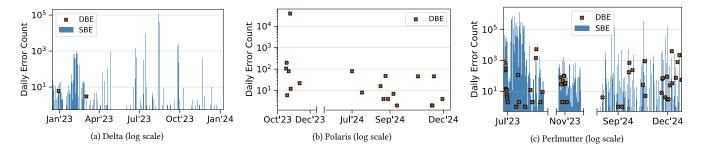


Figure 1: Daily SBEs and DBEs in the three studied clusters. Instances of bursty errors are observed in all three clusters.

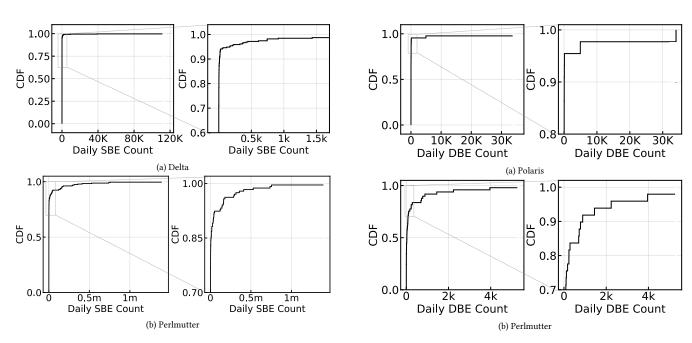


Figure 2: CDFs (cumulative distribution functions) of daily SBE count. Both clusters suffer from bursty errors, while Delta exhibits much more severe burstiness.

Figure 3: CDF of daily DBE count. Burstiness is observed when considering DBEs.

be corrected by ECC, it is useful to dig out whether SBE occurrence can indicate DBE occurrence.

In Delta cluster, there are only two DBE events observed throughout the year, each on distinct GPUs belonging to different nodes. Following its DBE event, one GPU encounters 2 SBEs concurrently. The other GPU, however, has no SBEs across the whole year. Compared to the days with over 1000 SBEs, these two DBE events are not happening on the days that experiencing bursty errors.

For Perlmutter supercomputer, there are 29 GPUs encountering both SBEs and DBEs (see the last row in Table 2). Their time-wise distribution of SBEs and DBEs are shown in Figure 4. We observe that for most of the DBE events, there are SBEs occurring on the same GPU on the same day. Although the data points are too few to calculate meaningful correlation for each GPU, it is clear that SBEs and DBEs are correlated. SBEs can be used as an indicator of possible future DBE occurrence. This also highlihgts the necessity of studying SBEs despite that it can be protected fully by ECC.

Observation 3: We observe correlation of SBEs and DBEs on the Perlmutter supercomputer.

3.2 Interarrival Time of Errors

We next analyze the error frequency by measuring the interarrival time of errors, i.e., Time Between Errors. We measure the error interarrival time in the system by calculating the elapsed time between two error-occurring events. Mean-Time-Between-Errors (MTBE) is then calculated by averaging the interarrival time values and reported in Table 3. To ensure a fair comparison of the interarrival time and MTBE across clusters, we maintain a consistent granularity across all datasets by aggregating error-occurring events into one-hour intervals, matching the log collection frequency of Perlmutter supercomputer (the longest one). We separately report the MTBE of SBEs and DBEs due to their distinct error rates and consequences, considering that SBEs are correctable, whereas DBEs

would cause program crashes. We do not calculate the double-bit MTBE for Delta, because there are only two such events. We also report the standard deviation of MTBE to indicate the distribution of interarrival times. As shown in Table 3, Perlmutter exhibits a shorter per-cluster MTBE than Delta (for SBEs) and Polaris (for DBEs), which is consistent with their respective system scales.

Observation 4: MTBE is affected by the cluster scale. As the largest cluster, Perlmutter has the lowest MTBE.

Similar to the overall error rates, one single number of MTBE is not sufficient to represent the overall resilience of a cluster. The bursty error patterns are reflected in the large standard deviation

Table 2: Overview of errors in the studied clusters.

Cluster Name	Delta	Polaris*	Perlmutter
# SBEs	173936	N/A	7010888
# SBE Events	3324	N/A	2016
SBE Rate (Per GPU Per Day)	0.53	N/A	2.83
SBE-occuring GPUs	43 (5.06%)	N/A	344 (4.52%)
# DBEs	9	39837	17926
# DBE Events	2	44	77
DBE Rate (Per GPU Per Day)	0.000027	0.069	0.0082
DBE-occuring GPUs	2 (0.24%)	68 (0.030%)	35 (0.46%)
GPUs w/ SBE+DBE	1	N/A	29

^{*}As SBEs are not recorded in Polaris, those SBE entries are "N/A".

Table 3: MTBE of the studied clusters.

Cluster Name	Delta	Polaris	Perlmutter
MTBE (SBE)	24.84 ±67.85	N/A*	7.08 ± 27.31
MTBE (DBE)	N/A [†]	228.23 ± 298.27	109.97±199.47

^{*}Polaris logs do not record SBEs.

values in Table 3. We investigate the distributions of interarrival times in clusters, as we show in Figure 5.

Figure 5(a) shows the CDF of SBE interarrival times in Delta cluster. 94.31% of the interarrival times are less than one hour, indicating the bursty error patterns. Meanwhile, we observe a long tail in the distribution: 0.81% of the SBE interarrival times exceed 120 hours, extending up to 851 hours (35 days). This long tail further underscores the bursty error patterns in Delta.

The SBE interarrival times of Perlmutter are presented in Figure 5(b). Among the SBE interarrival times in Perlmutter, 51.42% fall below one hour, which is significantly less than the Delta case where the majority (94.31%) are below one hour, indicating that Delta exhibits more severe bursty patterns than Perlmutter. The maximum SBE interarrival time in Perlmutter is around 5 days, which is much shorter compared to the one observed in Delta (35 days). This disparity is related to the scales of the two supercomputers. For a simplified illustration example, Perlmutter is about 8.96× larger than Delta; even assuming the same GPU error rate, this scale difference implies that encountering an SBE in Perlmutter is approximately 8.96× more likely than in Delta. Therefore, the observed difference of the longest interarrival time is reasonable and aligns with the expectations set by their scales. Moreover, these numbers do not reveal a linear relationship between the scale of the

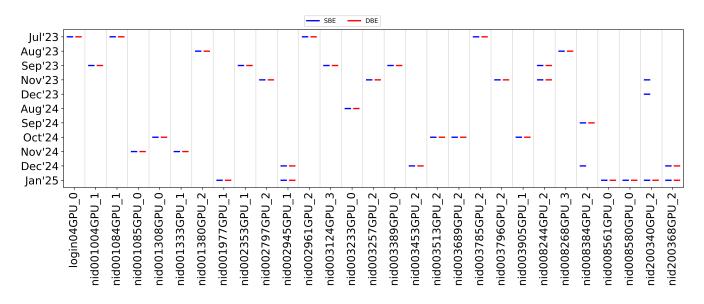


Figure 4: Timeline of GPUs that encounter both SBEs and DBEs in Perlmutter supercomputer. SBEs and DBEs are often correlated.

[†]Double-bit MTBE is not computed as Delta logs record only two DBEs.

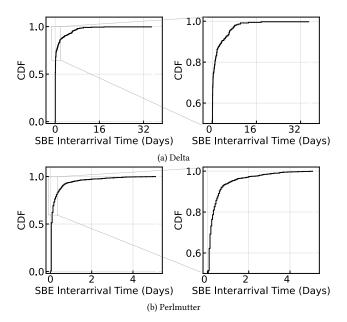


Figure 5: CDF of SBE interarrival times. Bursty errors are observed and Delta experience more severe burstiness than Perlmutter.

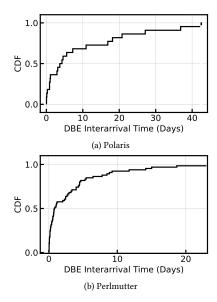


Figure 6: CDF of interarrival times of DBEs. Cluster scale affects error characteristics.

cluster and the error interarrival times, likely due to the complex nature of the cluster environment.

Figure 6 shows the CDF of interarrival times of DBEs. Given that the DBE rate is significantly lower than the SBE rate, the CDFs of DBE interarrival times are more sparse than the SBE ones and their characteristics are distinct. In Polaris, 27.27% of DBE interarrival times are less than one day, while in Perlmutter, the proportion is

Table 4: Correlation of errors and environmental factors. No strong correlation is observed.

		Temperature	Power	GPU Utilization
Delta	SBE	0.14	0.14	0.20
Polaris	DBE	0.089	0.17	0.18
Perlmutter	SBE	0.011	0.23	N/A*
	DBE	0.19	0.32	N/A*

^{*} Perlmutter logs do not record GPU utilization data.

51.56%. For the distribution tail which indicates long interarrival times, the longest DBE interarrival time in Polaris is 42 days, while the longest one in Perlmutter is 23 days. All these characteristics are intrinsically linked to the scale and the level of error burstiness of the two clusters.

Observation 5: Both bursty error patterns and supercomputer scale can affect the characteristics of error interarrival times. Relying on a single number, MTBE, may introduce bias into resilience estimation. It is necessary to consider the distribution of error interarrival times to obtain an accurate and holistic resilience assessment.

We explore the potential temporal correlation and periodicity of errors leveraging the autocorrelation function of error interarrival times. Autocorrelation evaluates the similarity between a time series and its lagged version and is always in the range of [-1, 1] [36]. A higher positive number refers to a stronger correlation, suggesting a higher level of periodicity with a certain period length, i.e., the lag. Zero values indicate the absence of correlation at the given lag and negative autocorrelation numbers point to the opposite relationship with its lagged series. We vary lags by hour to compute the autocorrelation values and explore possible periodicity. Figure 7 shows the autocorrelation of error interarrival times. High autocorrelations are observed with lags of less than 24 hours, which confirms the large body of bursty errors in all clusters. With lags increasing, we do not observe any notable correlation. We conclude that burstiness is severe in all clusters but no periodicity is observed.

Observation 6: No periodicity of errors is observed in clusters.

3.3 Correlation with Environmental Factors

In this section, we examine the correlation of SBE/DBE occurrences with environmental factors that may be related to error behavior. Specifically, we consider temperature, power consumption, and GPU utilization. Additionally, we also discuss the liquid cooling systems used in the three supercomputers and their impact on GPU error behavior.

Table 4 summarizes the correlation of SBEs and DBEs with these factors across the three clusters. While we do not have access to the detailed workload information (due to privacy issues), these

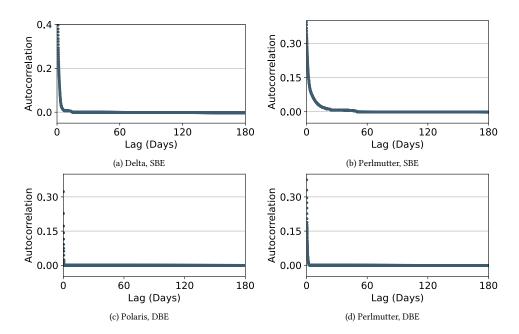


Figure 7: Autocorrelation of the DBE interarrival times. The high autocorrelation within a short lag indicates the bursty error patterns. There is no periodicity of errors observed.

environmental factors can serve as indirect indicators of workload patterns. Overall, we do not observe strong correlations. While weak – but not entirely negligible – correlations are present in most of the cases, except that the DBEs in Polaris and the SBEs in Perlmutter show no observable correlation with temperature. It is inherently complex to design and build a supercomputer, thus the sources and contributing factors of memory errors are also complicated, including, chip manufacturing variability, physical node placement, and other hardware-level influences. As such, we are not able to draw definitive conclusions regarding the influence of environmental factors. Nonetheless, we suggest prioritizing GPUs that consistently exhibit high temperature, power consumption, and GPU utilization for proactive reliability management.

All three supercomputers leverage direct liquid cooling systems with similar configurations. For brevity, here we present the liquid cooling flow of a single Perlmutter blade as an example, see Figure 8. Each blade hosts two nodes, each with one CPU and four GPUs. Within each node, two GPUs share one cooling loop. Additionally, we calculate the average temperature of GPUs, shown

Table 5: Average temperature in 3 clusters.

GPU ID	Delta	Polaris	Perlmutter
0		36.43	29.56
1	N/A [†]	39.88	28.48
2		36.38	29.57
3		41.18	28.49
Average	35.66	38.47	29.03

[†] We do not have information of the GPU ID mapping in Delta.

in Table 5. On average, Perlmutter shows the lowest temperatures, followed by Delta, while Polaris exhibits the highest. On both Polaris and Perlmutter, GPUs 0 and 1 share one cooling loop and GPUs 2 and 3 share the other. For Polaris, the loop starts with the lower-numbered GPU, whereas in Perlmutter, the loop begins with the higher-numbered GPU. This is confirmed by the temperature values in Table 5. Recall the error rate statistics in Table 2, Delta is the most reliable system, although here the average temperature of Delta is much higher than Perlmutter. Still, we cannot draw any conclusive relationship between temperature, cooling system design, and error characteristics.

Observation 7: For most environmental factors, we observe weak correlations with errors. No strong correlation is observed.

3.4 Spatial and Temporal Behaviour

From a supercomputer management and maintenance perspective, the spatial and temporal reliability characteristics of different GPUs are of interest to cluster administrators. We start with characterizing the transition of spatial error behavior across time in clusters, then we further investigate the severity of GPU errors over time.

We first investigate the spatial correlation of errors: in particular, whether GPUs within the same cabinet exhibit similar error characteristics. At the time of writing this paper, we only have the full machine topology for Delta, shown in Figure 9. In Delta, 207 GPU nodes with a total of 849 GPUs are organized in eight cabinets. We leverage this architectural information to determine the spatial distribution of SBEs in Delta. Given that bursty errors are commonly observed in Delta, instead of presenting the number of SBEs in each

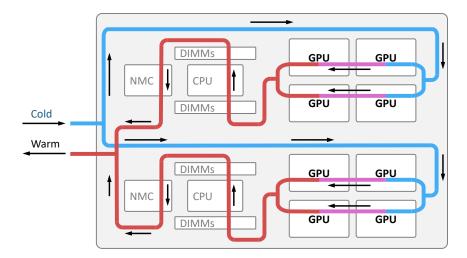


Figure 8: The liquid cooling flow of one Perlmutter blade. Each blade contains two nodes. Two GPUs share one cooling loop. The other two systems, Delta and Polaris, have a similar direct liquid cooling system.

cabinet, we calculate the number of days that a cabinet encounters errors (Figure 10) and the number of SBE-occurring GPUs within a cabinet (Figure 11). For brevity, we select three time windows (in February, June, and August) to present the dynamics of the physical distribution of errors. We observe that hotspots of SBEs vary across different months. This trend is consistent considering the two metrics, error-occurring days and error-occurring GPUs. This variation indicates physical correlations among errors, with the SBE-occurring cabinets changing over time.

Observation 8: GPU memory errors are spatially and temporally correlated in the Delta supercomputer.

We further investigate the severity of GPU errors over time. Figure 12 shows the number of SBEs over time observed in SBE-occurring GPUs in Delta. For ease of reading, we present the weekly SBE count, with darker colors indicating more errors observed in a GPU (x-axis) during a certain week (y-axis). Most of the GPUs suffer from SBEs for only 1–3 weeks. We do not observe any GPU that continually encounters SBEs throughout the whole year. Similar observations are drawn from the other SBE and DBE datasets, as depicted in Figure 13 (SBEs in Perlmutter) and Figure 14 (DBEs in Polaris and Perlmutter clusters). In general, GPUs with high error counts tend to vary over time, suggesting that for system monitoring and maintenance, focusing on GPUs with historically high error counts may not be a good strategy.

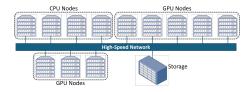


Figure 9: Physical location of nodes in Delta.

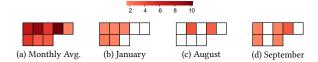


Figure 10: Number of error-occurring days within each cabinet over time. Hotspots vary over time.

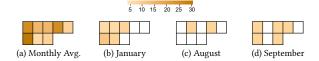


Figure 11: Number of SBE-occurring GPUs within each cabinet over time. Although the heatmap pattern changes slightly comparing to the characteristics when considering the number of error-occurring days shown in Figure 10, the same observation preserves: hotspots vary over time.

Observation 9: Erroneous GPUs vary over time. For the management of supercomputers, focusing on GPUs with historically high SBE counts may not be a good strategy.

4 Discussions and Lessons Learned

In this section, we discuss the opportunities for efficient machine health management and large-scale application checkpointing in HPC systems. These opportunities are enabled by our GPU memory error characterization study presented in Section 3. We also compare the Ampere GPU error statistics with previous generations to

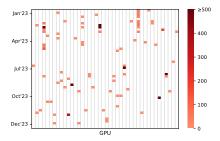


Figure 12: Number of SBEs observed on SBE-occuring GPUs over time in Delta. Most of the GPUs encounter SBEs for only 1–3 weeks. There is no GPU that continually encounters SBEs throughout the whole year.

observe the trend of GPU reliability. Specifically, we present quantitative comparisons to the Summit supercomputer with NVIDIA V100 GPUs.

4.1 Compared to Previous GPU Generations

Large-scale GPU memory error log analyses for supercomputers mainly focus on three systems: Summit, with V100 GPUs (the predecessor of A100) [28], Titan, with K20X GPUs [24–26], and Blue Waters, with K20X GPUs [22]. We briefly compare observations derived from our A100 GPU datasets with those of previous GPU generations, to seek similarities and differences. First, the bursty patterns are confirmed in the Titan supercomputer [25]. Second, notable periodicity is observed in Titan [25], however, we do not observe any periodicity in Delta or Perlmutter.

Furthermore, we quantitatively compare the characteristics of DBEs¹ between V100 (in the Summit supercomputer [28]) and A100 GPUs. Both generations of GPUs are equipped with HBM2 (High Bandwidth Memory 2). Note that consecutive DBEs are grouped and considered as one single DBE event in [28]. Here we do not cluster consecutive errors because grouping consecutive errors together filters out the lasting time of these errors which are meaningful to show the level of severeness of the error consequences. By processing these datasets through the same way and the same granularity (per hour), we still ensure a fair and meaningful comparison.

Figure 15 shows the daily DBE count in Summit, confirming a moderate level of error burstiness. Comparing to the daily DBE count of the three A100 supercomputers (Figure 3) where we sometimes observe more than 1000 errors per day, the maximum daily error count in Summit is 138. This indicates that the Summit supercomputer experiences bursty errors, but not as much as the supercomputers accelerated by A100 GPUs.

Furthermore, we calculate the DBE rate and MTBE values and present the statistics in Table 6. The DBE rate of Summit is higher than Delta but lower than that of Polaris and Perlmutter. There are more DBE-occurring GPUs in Summit, both in terms of raw numbers and the normalized percentage. The MTBE of Summit is slightly longer than Perlmutter but lower than Polaris. The standard deviation of Summit MTBE is much smaller than the A100 clusters,

Table 6: Error statistics in the Summit Supercomputer.

Cluster Name	Summit
# GPUs	28471
# Logged Days	1026
# DBEs	1088
# DBE Events	1008
DBE Rate (Per GPU Per Day)	0.00022
DBE-occuring GPUs	124 (2.6%)
MTBE	122.62 ±81.64

confirming that Summit suffers less from bursty errors. In general, we cannot clearly conclude the reliability ranking of these clusters. Given that all four clusters use HBM2 memory, it is reasonable that they share similar memory error characteristics.

Take-away Message 1: NVIDIA V100 and A100 GPUs both use HBM2 and share similar memory error characteristics.

4.2 Reliable Operation of Supercomputers

We discuss the insights derived from our characterization study from two major observations. Firstly, the observation of bursty error patterns suggests GPU ECC monitoring frequency. Secondly, our spatial and temporal analysis provides insights for cluster management and maintenance.

4.2.1 GPU ECC Monitoring Frequency. Among the three clusters we studied, GPU ECC errors are monitored at different frequencies. Polaris error logs are recorded every four seconds, while Delta errors are recorded every minute, and we retrieve Perlmutter errors using a frequency of every hour. This variation motivates a discussion of the trade-off between the overhead of error monitoring and the ability to promptly identify and react to errors. Ideally, monitoring should be performed as infrequently as possible while still capturing sufficient information to detect abnormal GPU behaviors, especially bursty errors.

Taking Delta as an example, we observe a series of consecutive error occurrence events: there are around 300 SBEs per minute over an eight-hour duration, as shown in Figure 16(a). We plot the error count in Figure 16(a) using the original error logging frequency of one minute, then present an aggregated hourly error count in Figure 16(b). With the hour-by-hour error logging, the bursty error pattern is still captured with hourly error counts of around 18,000. The timespan of the bursty error pattern (up to 8 hours in this case) is sufficiently long, thus the erroneous GPUs are still captured even with the logging frequency of an hour. Another example is shown in Figure 17. Although the hourly error counts capture the errors and indicate some level of burstiness, the bursty error patterns are not as clear as minute-by-minute error logging. Additionally, error log with higher frequency enables sooner erroneous GPU identification, so that system administrators can take action immediately.

Take-away Message 2: Coarser-level error monitoring does not suffer much information loss, yet monitoring errors at a finer level enables faster responses to errors.

 $^{^1\}mathrm{The}$ Summit public dataset [28] of V100 GPU errors does not contain SBE information.

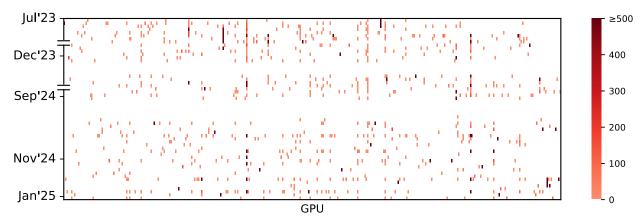


Figure 13: Per-GPU SBE counts over time in Perlmutter. Here we show all the SBE-occurring GPUs. Erroneous GPUs vary over time

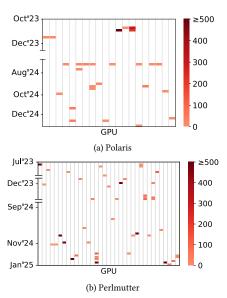


Figure 14: Per-GPU DBE counts observed over time on Polaris and Perlmutter. There are less DBE-occurring GPUs than the case of SBEs. The same observation can be derived: DBE-encountering GPUs vary over time.

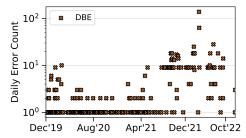


Figure 15: Daily DBE count (log scale) in the Summit supercomputer, equipped with NVIDIA V100 GPUs. Bursty patterns are observed, but not as severe as the A100 GPUs.

4.2.2 Lessons for GPU Management. The characterization of spatial and temporal reliability behavior motivates two general key

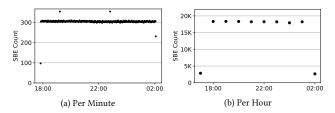


Figure 16: Example of Delta bursty patterns under different monitoring frequencies from 08/14/2023 to 08/15/2023. Changing monitoring frequency can lead to information change, but no significant information loss is noted.

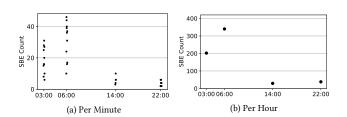


Figure 17: Another example of Delta bursty patterns shown in different monitoring frequencies on 02/01/2023.

observations that summarize the dynamics of erroneous GPUs: 1) SBEs can be used as an indicator of future DBE occurrence and 2) erroneous GPUs vary over time. These observations motivate the need for a smart and flexible approach to efficient and low-cost management of supercomputers. Developing proactive and dynamic GPU monitoring strategies based on erroneous GPU prediction is a potential solution and is subject to our future work.

Take-away Message 3: Dynamic GPU monitoring and erroneous GPU prediction strategies are needed for efficient and reliable operation of supercomputers.

4.3 Determining Checkpoint Intervals

A standard and common practice for handling fail-stop failures is checkpointing, where copies of application state are stored persistently during execution and these copies enable applications to be re-executed from the checkpointed state. Checkpointing can require application and system knowledge to accurately capture the application state. Prior studies [37, 38] have shown that an optimal checkpoint interval can be determined based on two factors: the time to save a checkpoint into persistent storage, and the mean-time-between-failure of a system. Taking LFM training as an example, pre-training a model with hundreds of billions of parameters takes thousands of GPUs for months. One DBE could result in a hardware exception and cause the whole program to stop without terminating the batch job (e.g., a SLURM job). The allocated computing resources can be wasted if monitoring and restart are not performed in a timely manner. This problem is exacerbated as state-of-the-art LFM training exceeds $O(10^4)$ GPUs [39]. The insights from our study can be used as a reference for determining the checkpointing interval of LFM training and general scientific application execution. However, considering the bursty patterns observed in this study, simply computing an optimal but static checkpoint interval might lead to sub-optimal performance and utilization.

An alternative approach is to use dynamic checkpointing based on frequent memory error monitoring. This dynamic checkpointing strategy should be intelligent enough to tell the period within bursty errors and between bursty errors, so that the strategy can adjust the checkpoint interval based on the error-free time length as well as upon the observation of SBEs and DBEs. In this way, we will be able to minimize the checkpoint overhead while minimizing the failure loss upon uncorrectable memory errors.

Take-away Message 4: The current practice of popular HPC applications for checkpointing is based on a fixed frequency that either fails to meet the reliability requirement or causes high overhead. Dynamic checkpointing is suggested to accommodate bursty error patterns observed in supercomputers.

5 Related Work

There exist numerous studies of failures for different components in clusters, data centers, and supercomputers [40–46]. Many works focus on the characterization and analysis of failures in supercomputers [43–46]. Roy et al. perform reliability analysis on the Mira Supercomputer from the perspective of the cooling system [42]. Several works delve into silent data corruption (SDC) in data centers, raising awareness within the community of the need for SDC mitigations [12, 41]. There have been many studies of the impact of memory errors on large machines [17–21, 24–26, 47, 48]. Researchers have examined how correctable errors affect HPC application performance via simulation [19]. Bautista et al. study the raw DRAM error rate without ECC protection on CPU memory [18]. Feng Shui [20] studied the spatial distribution of DRAM and SRAM faults on legacy Cielo and Jaguar supercomputers, confirming that the DRAM and SRAM of Cielo exhibit no aging effects or noticeable

increase in the five-year lifetime [17]. Recently, Beigi et al. performed a detailed study on DDR4 DRAM faults [21], highlighting concerns regarding multi-bit errors.

On the GPU side, the majority of the studies focus on GPU failures [29, 49, 50] and memory errors [22-28, 51]. Taherin et al. characterize GPU failures and repairs on Tsubame-2 and Tsubame-3 based on system maintenance log, focusing on fatal errors such as GPU driver-related problems, kernel panic, and software bugs, but no GPU memory errors were reported by ECC [49]. GPU failure logs are used to perform error prediction using machine learning models [50]. Most prior GPU memory error studies have been performed on the Titan supercomputer equipped with K20X GPUs [24– 27, 51, 52]. Periodicity of memory errors is observed in the Titan supercomputer [25]. Di et al. study error behavior of the Blue Waters supercomputer with K20X GPUs [22] but with a focus on CPU-GPU comparison. The temperature effect is found to be related to GPU memory errors [25, 26]. Debardeleben et al. perform experiments on NVIDIA Tesla M2090 GPUs (using Fermi 2.0 architecture) in Moonlight [23] and report the rate of DBEs. These studies mostly focus on earlier generations of GPU architecture. Most of the machines in these works have been decommissioned at the time of study.

There are two recent studies on the resilience characterization of GPU-accelerated supercomputers. Oles et al. [28] perform a case study of GPU memory errors on the Summit supercomputer, equipped with NVIDIA V100 GPUs, focusing on double-bit errors (DBEs) but not single-bit errors (SBEs). Their analysis explored potential causes of DBEs, such as GPU placement and workload characteristics. The latest study of GPU reliability is performed on the Delta supercomputer [29] where both NVIDIA A100 and H100 GPUs are considered, focusing on GPU hardware failures and interconnect errors but not memory errors. While we acknowledge that studies based on a single supercomputer can still produce valuable insights, we argue that conclusions drawn from a single system may not generalize to broader GPU deployments, given the varying reliability characteristics across the three supercomputers examined in this work.

In short, our study provides a large-scale, cross-supercomputer analysis of GPU memory errors on machines currently in production. As detailed in Section 3 and Section 4, our findings from a cross-supercomputer perspective offer timely and practical insights into the reliability of modern GPU-based systems, filling an important gap in the existing literature.

6 Conclusions and Future Work

We have presented an in-depth quantitative error analysis of memory errors in Ampere GPUs across three heavily utilized, in-production supercomputers of varying scales: Delta [30], Polaris [31], and Perlmutter [32]. We collect and analyze GPU memory ECC (Error Correction Codes) error logs, resulting in a total of 67.77 million GPU device-hours of data covering 10,693 NVIDIA Ampere GPUs.

We quantitatively measure the error rate and Mean-Time-Between-Errors (MTBE) in these clusters. The burstiness, periodicity, spatial and temporal relation of errors, and environmental factors in these clusters are also explored in detail. The key observations include 1)

Bursty error patterns have a significant impact on the characteristics of error rate and MTBE; 2) Cluster scale also affects MTBE but the relationship is not linear; 3) Errors have spatial correlation but exhibit no periodical patterns; 4) No strong correlation of errors is observed with environmental factors such as temperature, power, and GPU utilization; and 5) GPU errors are spatially and temporally correlated.

Furthermore, we summarize the lessons learned from this study. We also compare the Ampere GPU error statistics with previous generations to observe the trend of GPU reliability and understand the nature of GPU errors. We discuss the opportunities for efficient machine health management and large-scale application checkpointing. We bring up the possibility of dynamic checkpointing strategies informed by memory error monitoring. Our observations and analyses motivate the future studies to explore the possibility of designing error prediction models and dynamic checkpointing algorithms based.

Acknowledgments

This material is based upon work supported by the National Science Foundation (NSF) grants (#2402940 and #2410856) and the Commonwealth Cyber Initiative (CCI) grant (#HC-3Q24-047). The platforms used for evaluation in this work are supported by the U.S. DOE Office of Science, Office of Advanced Scientific Computing Research, under award 66150: "CENATE - Center for Advanced Architecture Evaluation". The Pacific Northwest National Laboratory is operated by Battelle for the U.S. Department of Energy under Contract DE-AC05-76RL01830. This project is supported by resources provided by the Office of Research Computing at George Mason University (URL: https://orc.gmu.edu) and funded in part by grants from the National Science Foundation (Award Number 2018631). This research used both the DeltaAI advanced computing and data resource, which is supported by the National Science Foundation (award OAC 2320345) and the State of Illinois, and the Delta advanced computing and data resource which is supported by the National Science Foundation (award OAC 2005572) and the State of Illinois.. Delta and DeltaAI are joint efforts of the University of Illinois Urbana-Champaign and its National Center for Supercomputing Applications. This research used resources of the Argonne Leadership Computing Facility, a U.S. Department of Energy (DOE) Office of Science user facility at Argonne National Laboratory and is based on research supported by the U.S. DOE Office of Science-Advanced Scientific Computing Research Program, under Contract No. DE-AC02-06CH11357. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a Department of Energy Office of Science User Facility using NERSC award DDR-ERCAP0032037.

References

- Y. Chen, W. Li, R. Fan, and X. Liu, "Gpu optimization for high-quality kinetic fluid simulation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 28, no. 9, pp. 3235–3251, 2021.
- [2] Y. Liu, X. Liu, and E. Wu, "Real-time 3d fluid simulation on gpu with complex obstacles," in 12th Pacific Conference on Computer Graphics and Applications, 2004. PG 2004. Proceedings., pp. 247–256, IEEE, 2004.
- [3] J. Glaser, T. D. Nguyen, J. A. Anderson, P. Lui, F. Spiga, J. A. Millan, D. C. Morse, and S. C. Glotzer, "Strong scaling of general-purpose molecular dynamics simulations on gpus," Computer Physics Communications, vol. 192, pp. 97–107, 2015.
- lations on gpus," Computer Physics Communications, vol. 192, pp. 97–107, 2015.
 [4] T. Boku, M. Sugita, R. Kobayashi, S. Furuya, T. Fujie, M. Ohue, and Y. Akiyama, "Improving performance on replica-exchange molecular dynamics simulations by

- optimizing gpu core utilization," in Proceedings of the 53rd International Conference on Parallel Processing, pp. 1082–1091, 2024.
- [5] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding with unsupervised learning," 2018. Technical report, OpenAI. https://openai.com/index/language-unsupervised/.
- [6] S. Zhang, S. Roller, N. Goyal, M. Artetxe, M. Chen, S. Chen, C. Dewan, M. Diab, X. Li, X. V. Lin, T. Mihaylov, M. Ott, S. Shleifer, K. Shuster, D. Simig, P. S. Koura, A. Sridhar, T. Wang, and L. Zettlemoyer, "OPT: Open pre-trained transformer language models," *Preprint arXiv:2205.01068*, 2022.
- M. Zvyagin, A. Brace, K. Hippe, Y. Deng, B. Zhang, C. Orozco Bohorquez, A. Clyde, B. Kale, D. Perez-Rivera, H. Ma, C. M. Mann, M. Irvin, J. G. Pauloski, L. Ward, V. Hayot, M. Emani, S. Foreman, Z. Xie, D. Lin, M. Shukla, W. Nie, J. Romero, C. Dallago, A. Vahdat, C. Xiao, T. Gibbs, I. Foster, J. J. Davis, M. E. Papka, T. Brettin, R. Stevens, A. Anandkumar, V. Vishwanath, and A. Ramanathan, "GenSLMs: Genome-scale language models reveal SARS-CoV-2 evolutionary dynamics," The International Journal of High Performance Computing Applications, vol. 37, no. 6, pp. 683–705, 2023.
 V. Fratin, D. A. G. de Oliveira, C. B. Lunardi, F. Santos, G. Rodrigues, and
- [8] V. Fratin, D. A. G. de Oliveira, C. B. Lunardi, F. Santos, G. Rodrigues, and P. Rech, "Code-dependent and architecture-dependent reliability behaviors," in DSN, pp. 13–26, 2018.
- [9] S. Ganapathy, J. Kalamatianos, B. M. Beckmann, S. Raasch, and L. G. Szafaryn, "Killi: Runtime fault classification to deploy low voltage caches without MBIST," in 25th IEEE International Symposium on High Performance Computer Architecture, pp. 304–316, IEEE, 2019.
- [10] M. B. Sullivan, N. Saxena, M. O'Connor, D. Lee, P. Racunas, S. Hukerikar, T. Tsai, S. K. S. Hari, and S. W. Keckler, "Characterizing and mitigating soft errors in GPU dram," in MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture, pp. 641–653, 2021.
- [11] MetaSeq, "OPT-175B Baselines logbook," 2022. https://github.com/ facebookresearch/metaseq/blob/main/projects/OPT/chronicles/OPT175B_ Logbook.pdf.
- [12] H. D. Dixit, S. Pendharkar, M. Beadon, C. Mason, T. Chakravarthy, B. Muthiah, and S. Sankar, "Silent data corruptions at scale," *Preprint arXiv:2102.11245*, 2021.
- [13] Y. He, M. Hutton, S. Chan, R. De Gruijl, R. Govindaraju, N. Patil, and Y. Li, "Understanding and mitigating hardware failures in deep learning training systems," in 50th Annual International Symposium on Computer Architecture, pp. 1–16, 2023.
- [14] Z. Zhang, L. Huang, R. Huang, W. Xu, and D. S. Katz, "Quantifying the impact of memory errors in deep learning," in *IEEE International Conference on Cluster Computing*, pp. 1–12, IEEE, 2019.
- [15] G. Li, S. K. S. Hari, M. Sullivan, T. Tsai, K. Pattabiraman, J. Emer, and S. W. Keckler, "Understanding error propagation in deep learning neural network (DNN) accelerators and applications," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12, 2017.
- [16] BigScience, "TR11-176B Training Logbook," 2022. https://github.com/bigscience-workshop/bigscience/blob/master/train/tr11-176B-ml/chronicles.md.
- [17] S. Levy, K. B. Ferreira, N. DeBardeleben, T. Siddiqua, V. Sridharan, and E. Baseman, "Lessons learned from memory errors observed over the lifetime of Cielo," in SC18: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 554–565, IEEE, 2018.
- [18] L. Bautista-Gomez, F. Zyulkyarov, O. Unsal, and S. McIntosh-Smith, "Unprotected computing: A large-scale study of DRAM raw error rate on a supercomputer," in SC'16: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 645–655, IEEE, 2016.
- [19] K. B. Ferreira, S. Levy, V. Kuhns, N. DeBardeleben, and S. Blanchard, "Understanding the effects of DRAM correctable error logging at scale," in *IEEE International Conference on Cluster Computing*, pp. 421–432, IEEE, 2021.
- [20] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi, "Feng Shui of supercomputer memory: Positional effects in DRAM and SRAM faults," in *International Conference on High Performance Computing, Networking, Storage and Analysis*, pp. 1–11, 2013.
- [21] M. V. Beigi, Y. Cao, S. Gurumurthi, C. Recchia, A. Walton, and V. Sridharan, "A systematic study of DDR4 DRAM faults in the field," in *IEEE International Symposium on High-Performance Computer Architecture*, pp. 991–1002, IEEE, 2023.
- [22] C. Di Martino, Z. Kalbarczyk, R. K. Iyer, F. Baccanico, J. Fullop, and W. Kramer, "Lessons learned from the analysis of system failures at petascale: The case of Blue Waters," in 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 610–621, IEEE, 2014.
- [23] N. DeBardeleben, S. Blanchard, L. Monroe, P. Romero, D. Grunau, C. Idler, and C. Wright, "GPU behavior on a large HPC cluster," in *Euro-Par Workshops* 2013, pp. 680–689, Springer, 2014.
- [24] D. Tiwari, S. Gupta, J. Rogers, D. Maxwell, P. Rech, S. Vazhkudai, D. Oliveira, D. Londo, N. DeBardeleben, P. Navaux, L. Carrot, and A. Bland, "Understanding GPU errors on large-scale HPC systems and the implications for system design and operation," in *IEEE 21st International Symposium on High Performance Computer Architecture*, pp. 331–342, IEEE, 2015.
- [25] B. Nie, D. Tiwari, S. Gupta, E. Smirni, and J. H. Rogers, "A large-scale study of soft-errors on GPUs in the field," in *IEEE International Symposium on High*

- Performance Computer Architecture, pp. 519-530, IEEE, 2016.
- [26] B. Nie, J. Xue, S. Gupta, C. Engelmann, E. Smirni, and D. Tiwari, "Characterizing temperature, power, and soft-error behaviors in data center systems: Insights, challenges, and opportunities," in IEEE 25th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, pp. 22–31, IEEE, 2017.
- [27] G. Ostrouchov, D. Maxwell, R. A. Ashraf, C. Engelmann, M. Shankar, and J. H. Rogers, "GPU lifetimes on Titan supercomputer: Survival analysis and reliability," in SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pp. 1–14, IEEE, 2020.
- [28] V. Oles, A. Schmedding, G. Ostrouchov, W. Shin, E. Smirni, and C. Engelmann, "Understanding gpu memory corruption at extreme scale: The summit case study," in *Proceedings of the 38th ACM International Conference on Supercomputing*, pp. 188–200, 2024.
- [29] S. Cui, A. Patke, Z. Chen, A. Ranjan, H. Nguyen, P. Cao, S. Jha, B. Bode, G. Bauer, C. Narayanaswami, et al., "Characterizing gpu resilience and impact on ai/hpc systems," arXiv preprint arXiv:2503.11901, 2025.
- [30] National Center for Supercomputing Applications, "Delta." https://www.ncsa. illinois.edu/research/project-highlights/delta/.
- [31] Argonne National Laboratory, "Polaris." https://www.alcf.anl.gov/polaris.
- [32] National Energy Research Scientific Computing Center, "Perlmutter." https://docs.nersc.gov/systems/perlmutter/architecture/.
- [33] NVIDIA Corporation, "NVIDIA Ampere Architecture Whitepaper." https://www.nvidia.com/content/PDF/nvidia-ampere-ga-102-gpu-architecture-whitepaper-v2.pdf.
- [34] NVIDIA Corporation, "NVIDIA Data Center GPU Manager (DCGM)." https://docs.nvidia.com/data-center-gpu-manager-dcgm/index.html.
- [35] Top500, "NOVEMBER 2023 Top500 Supercomputers." https://www.top500.org/lists/top500/2023/11/.
- [36] L. M. Leemis and S. K. Park, Discrete-event simulation: A first course. Pearson Prentice Hall Upper Saddle River, 2006.
- [37] J. W. Young, "A first order approximation to the optimum checkpoint interval," Communications of the ACM, vol. 17, no. 9, pp. 530–531, 1974.
- [38] J. T. Daly, "A higher order estimate of the optimum checkpoint interval for restart dumps," Future Generation Computer Systems, vol. 22, no. 3, pp. 303–312, 2006.
- [39] Z. Jiang, H. Lin, Y. Zhong, Q. Huang, Y. Chen, Z. Zhang, Y. Peng, X. Li, C. Xie, S. Nong, Y. Jia, S. He, H. Chen, Z. Bai, Q. Hou, S. Yan, D. Zhou, Y. Sheng, Z. Jiang, H. Xu, H. Wei, Z. Zhang, P. Nie, L. Zou, S. Zhao, L. Xiang, Z. Liu, Z. Li, X. Jia, J. Ye, X. Jin, and X. Liu, "MegaScale: Scaling large language model training to more than 10,000 GPUs," Preprint arXiv:2402.15627, 2024.
- [40] S. Han, P. P. Lee, F. Xu, Y. Liu, C. He, and J. Liu, "An in-depth study of correlated failures in production SSD-Based data centers," in 19th USENIX Conference on File and Storage Technologies, pp. 417–429, 2021.

- [41] S. Wang, G. Zhang, J. Wei, Y. Wang, J. Wu, and Q. Luo, "Understanding silent data corruptions in a large production CPU population," in 29th Symposium on Operating Systems Principles, pp. 216–230, 2023.
- [42] R. B. Roy, T. Patel, R. Kettimuthu, W. Allcock, P. Rich, A. Scovel, and D. Tiwari, "Operating liquid-cooled large-scale systems: Long-term monitoring, reliability analysis, and efficiency measures," in *IEEE International Symposium on High-*Performance Computer Architecture, pp. 881–893, IEEE, 2021.
- [43] S. Di, H. Guo, E. Pershey, M. Snir, and F. Cappello, "Characterizing and understanding HPC job failures over the 2K-day life of IBM BlueGene/A system," in 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 473–484, IEEE, 2019.
- [44] Y. Gao, X. Shi, H. Lin, H. Zhang, H. Wu, R. Li, and M. Yang, "An empirical study on quality issues of deep learning platform," in IEEE/ACM 45th International Conference on Software Engineering: Software Engineering in Practice, pp. 455–466, IEEE, 2023.
- [45] A. Das, F. Mueller, and B. Rountree, "Systemic assessment of node failures in HPC production platforms," in *IEEE International Parallel and Distributed Processing Symposium*, pp. 267–276, IEEE, 2021.
- [46] E. Rojas, E. Meneses, T. Jones, and D. Maxwell, "Analyzing a five-year failure record of a leadership-class supercomputer," in 31st International Symposium on Computer Architecture and High Performance Computing, pp. 196–203, IEEE, 2019
- [47] D. Zivanovic, P. E. Dokht, S. Moré, J. Bartolome, P. M. Carpenter, P. Radojković, and E. Ayguadé, "DRAM errors in the field: A statistical approach," in *International Symposium on Memory Systems*, pp. 69–84, 2019.
- [48] R.-T. Liu and Z.-N. Chen, "A large-scale study of failures on petascale supercomputers," Journal of Computer Science and Technology, vol. 33, no. 1, pp. 24–41, 2018.
- [49] A. Taherin, T. Patel, G. Georgakoudis, I. Laguna, and D. Tiwari, "Examining failures and repairs on supercomputers with multi-GPU compute nodes," in 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 305–313. IEEE, 2021.
- [50] H. Liu, Z. Li, C. Tan, R. Yang, G. Cao, Z. Liu, and C. Guo, "Predicting GPU failures with high precision under deep learning workloads," in 16th ACM International Conference on Systems and Storage, pp. 124–135, 2023.
 [51] S. S. Vazhkudai, R. Miller, D. Tiwari, C. Zimmer, F. Wang, S. Oral, R. Gunasekaran,
- [51] S. S. Vazhkudai, R. Miller, D. Tiwari, C. Zimmer, F. Wang, S. Oral, R. Gunasekaran, and D. Steinert, "GUIDE: a scalable information directory service to collect, federate, and analyze logs for operational insights into a leadership HPC facility," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12, 2017.
- [52] D. Tiwari, S. Gupta, G. Gallarno, J. Rogers, and D. Maxwell, "Reliability lessons learned from GPU experience with the Titan supercomputer at Oak ridge Leadership Computing Facility," in *International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–12, 2015.