GPU Lifetimes on Titan Supercomputer: Survival Analysis and Reliability

George Ostrouchov
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
ostrouchovg@ornl.gov

Don Maxwell
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN, USA
maxwellde@ornl.gov

Rizwan A. Ashraf
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
ashrafra@ornl.gov

Christian Engelmann
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Oak Ridge, TN, USA
engelmannc@ornl.gov

Mallikarjun Shankar
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN, USA
shankarm@ornl.gov

James H. Rogers
National Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN, USA
jrogers@ornl.gov

***Abstract*—**
The Cray XK7 Titan was the top supercomputer system in the world for a long time and remained critically important throughout its nearly seven year life. It was an interesting machine from a reliability viewpoint as most of its power came from 18,688 GPUs whose operation was forced to execute three rework cycles, two on the GPU mechanical assembly and one on the GPU circuitboards. We write about the last rework cycle and a reliability analysis of over 100,000 years of GPU lifetimes during Titan's 6-year-long productive period. Using time between failures analysis and statistical survival analysis techniques, we find that GPU reliability is dependent on heat dissipation to an extent that strongly correlates with detailed nuances of the cooling architecture and job scheduling. We describe the history, data collection, cleaning, and analysis and give recommendations for future supercomputing systems. We make the data and our analysis codes publicly available.

***Index Terms*—GPU, reliability, supercomputer, NVIDIA, Cray, large-scale systems, log analysis, MTBF, Kaplan-Meier survival, Cox regression, GPU failure data set**

I. Introduction

The Cray XK7 Titan supercomputer [23] was the #1 system in the world for a long time [18], and has remained a critically important computer system through the end of its life in the Summer of 2019. It defied scale with 18,688 individual NVIDIA GPU accelerated compute nodes and delivered tens of billions of computing hours to the U.S. Department of Energy mission-critical programs for nearly 7 years.

From a reliability perspective, Titan was a very interesting machine. Its operation was forced to execute three very significant rework cycles, two on the mechanical assembly affecting the PCIe connector from the GPU daughtercard to the motherboard, and one to replace about 11,000 GPU assemblies because of a failing resistor on their printed circuit board. We write primarily about the GPU operation epoch that includes this last rework cycle. This epoch of nearly 6 years includes Titan's most stable and failure free period and contains the most reliable data on GPU operation.

Figure 1 illustrates the chronology of the rework cycles and stable periods as indicated by the number of GPU changes at periodic inventories. The first two rework cycles before 2014 involving blade mechanical assemblies can be characterized as a break-in period on a new system that is first of its kind and pushes many technological boundaries. This early period included extensive field engineering and experimentation, down times, as well as temporal and completeness gaps in inventory runs due to down times. We include the early swap data in this figure only for completeness and to underscore the massive amount of hardware work required to field a world's largest supercomputer. However, as we focus on GPU operation after the second rework cycle we exclude this early period from further analysis. Late 2013 begins a very long period of high reliability and stable operation with very few issues until about

This work was sponsored by the U.S. Department of Energy's Office of Advanced Scientific Computing Research. This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

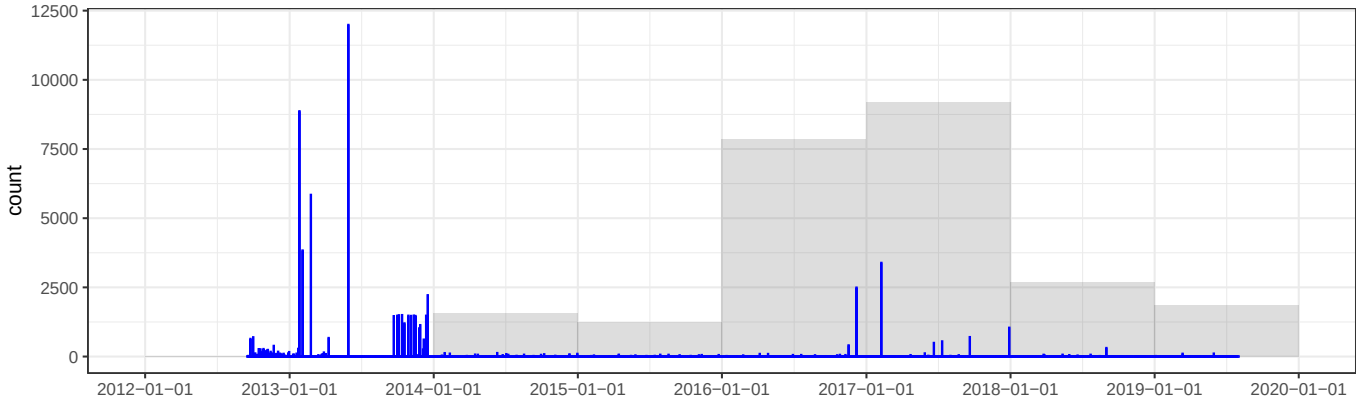

Fig. 1. Volume of GPU swaps detected on Titan at individual inventories (narrow blue) and yearly sum totals for 2014 and later (wide gray) over its entire lifetime. High swap volumes prior to 2014 reflect two major rework cycles on mechanical assembly issues at PCIe connectors that can be characterized as a break-in period. Our analysis in this paper excludes the break-in period and is focused on units in place by the end of 2013 or swapped in during the years that follow.

mid 2016, when GPU failures begin to rise. This results in the final rework cycle, replacing 11,000 GPUs from late 2016 through much of 2017.

Finding the root cause for the 2016 failures took a great deal of testing and the involvement of materials science and microscopy researchers. The failures were traced to a resistor on the GPU circuit board (not the GPU chip itself) due to silver sulfide corrosion. Growth of such corrosion products in ambient air on microelectronics parts is described in [35] and is consistent with failures starting only after a critical amount of corrosion builds up, a situation that matches the experience on Titan.

Throughout this paper, when we refer to replacing the GPU, we refer to the entire circuit board along with its GPU chip. Our analysis focuses on GPU boards that were installed in the second rework cycle and later, essentially units installed near the beginning of 2014 and later. A more precise definition of this analysis cohort is in Sec. IV. Figure 1 is the only one involving data on units removed prior to 2014.

Our data processing and analysis is performed with R [26] and Python [34], and a number of their packages. We make the data as well as our R codes and Python codes to reproduce the analyses and graphics in this paper publicly available (see, [24]). Due to the size of Titan, the data represents over 100,000 collective years of GPU operation, which may be the largest publicly available GPU reliability data set. We include the full original 2012-2019 data set as well as two more data sets resulting from our processing described in Sec. IV of this paper. The analysis codes also contain further minor details and additional analyses. Our hope is that others will use the data for reliability and survival analysis research and go beyond the results presented here.

We begin by describing related work in Sec. II. Our data collection is described in Sec. III and the data cleaning, checking, and GPU lifetime addition process in Sec. IV. Here we develop novel visualizations of GPU lifetimes and errors that were needed to understand the data and to verify our data processing decisions. A time between failures (TBF) analysis is given in Sec. V. Survival analysis (SA) based on Kaplan-Meier modeling (KM) and Cox regression modeling (CR) of relative hazard rates in various locations are in Sec. VI. The data requirements and goals of a TBF analysis and of SA are different: TBF analysis studies error log data to show how GPU reliability affects the machine, whereas SA methods study lifetime data of the GPUs to show how the machine affects their reliability. The application of SA methods is novel in the HPC reliability context. It has roots in biostatistics and epidemiology, where it is used to discover causes of disease. Finally, we discuss our main conclusions and recommendations in Sec. VII.

II. Related Work

Being the largest machine with the most GPUs in the world attracts a lot of attention so particularly during Titan's first few years of operation, many studies were published about different aspects of its reliability. However, none of the studies investigate the atypical failure mode discussed in this paper nor do they consider data over the full lifetime of the system through its decommissioning.

Tiwari et al. [30], [33] analyze types of GPU errors on Titan and on a GPU cluster at Los Alamos National Laboratory. This includes neutron beam experiments at the Los Alamos Neutron Science Center and at Rutherford Appleton Laboratories to further understand GPU soft errors caused by neutron radiation triggered upsets (bit flips and timing errors). Tiwari et al. [32] further analyze several types of GPU errors on Titan, including software/firmware related errors and failures, ECC double-bit errors, as well as GPU "off the bus" and ECC page retirement events. This work observes an ECC double-bit mean-time between errors (MTBE) of about 160 hours, or about one per week, with 86% occurring in GPU memory and the rest in the GPU register file. Their GPU "off the bus" events were caused by a system integration issue that was fixed and not an inherent GPU or GPU memory architecture flaw. Both [33] and

[32] report temperatures observed on some Titan components, which we use in Sec. VI to confirm our interpretation of airflow effects on temperature.

Nie et al. [19]–[21] characterize the soft error behavior of Titan's GPUs in relation to temperature and power consumption to predict its increased occurrence by correlating data in temporal and spatial domains with machine learning models. The work focuses primarily on correctable single-bit errors. Using Titan data from June 2013 to February 2015, Tiwari et al. and Nie et al. do not address the atypical failure mode discussed in this paper, as it did not surface until mid 2016.

Zimmer et al. [36] develop a new job scheduling strategy for Titan to counter the GPU failures that we discuss and to improve system utilization and productivity. The solution uses reordering of the compute nodes for resource allocation, scheduling larger jobs on more reliable nodes and smaller jobs on less reliable nodes.

Ezell [6] creates a general understanding of Titan's interconnect failures using an application to stress test its Gemini interconnect. The work by Kumar et al. [15] analyzes Titan's interconnect faults, errors and congestion events to improve resilience of the interconnects and their congestion resolution methods. The results show that the magnitude of interconnect errors is very high with an uneven distribution across different types of links. They also show that congestion is very frequent and bursty. Gupta et al. [9] investigate the spatial and temporal properties of failures on Titan and their impact on resilience mechanisms with implications for efficient system operation. Gupta et al. [10] also perform a study covering five supercomputers at Oak Ridge National Laboratory, including Titan. The study concentrates on developing an understanding of errors and failure frequencies over multiple generations of supercomputers and over their years of operation. This work resulted in many lessons learned, including that the mean-time between failures (MTBF) can change drastically and non-monotonically over time. Meneses et al. [17] analyze the interplay and workload on Titan using failure and job submission logs. The results indicate that failures depend heavily on workload.

Bautista-Gomez et al. [1] use failure data from Titan to dynamically adapt checkpoint frequency to the current reliability of the system. Tiwari et al. [31] also use failure data from Titan to exploit temporal locality in errors. Temporal clustering of errors allows lazy checkpointing when given error-free time thresholds are reached. Both approaches, Bautista-Gomez et al. and Tiwari et al., aim at dealing with the drastically and non-monotonically changing MTBF of Titan to match current system reliability with an efficient recovery strategy.

Other work in characterizing supercomputer faults, errors, and failures focuses on various other systems deployed in the United States, primarily at Department of Energy laboratories, and around the world. Di et al. [5] develop an in-depth understanding of failure characteristics of the IBM Blue Gene/Q Mira system at Argonne National Laboratory. This work shows that 99.4% of job failures are due to user behavior, such as errors in the code or misconfiguration. Martino et al. [16] characterize the errors and failures on the Blue Waters Cray XE6/XK7 system located at the National Center for Supercomputing Applications of the University of Illinois at Urbana-Champaign. The results show that 74.4% of system-wide outages in Blue Waters were caused by software. Notably, Blue Waters's XK7 partition had the same architecture as Titan, but did not experience the same GPU failure mode detailed in this paper.

III. Data Collection and Preprocessing

Data on Titan GPU lifetimes is constructed from two sources: inventory runs and failure event log records. Two types of failure events were collected for this study: Double Bit Error (DBE) and Off the Bus (OTB). DBE is an error correcting code (ECC) detection in GPU memory, which can correct a single bit flip and detect but not correct a double bit flip. OTB is the loss of host CPU connection to the GPU. While other types of errors were recorded in log files, as is reported in [9], [30], OTB and DBE became dominant in 2016 and were found to be the "signature" event of the GPU board failing resistor and a trigger for GPU replacement.

An inventory of GPU serial numbers and their locations was recorded each time the system boots, typically for software updates and patches but sometimes for hardware swaps, although warm swaps of blades were usually possible. Boots typically occurred once every few days but sometimes even a few times per day. A single inventory takes about a minute and is recorded in a separate file. It can be incomplete if a node response times out but such occurrences were relatively rare. Figure 2 shows that during 2014 and 2015, inventories were far apart, even 56 days once in 2014. We note this to illustrate that exact times for GPU removal from service are not known, the are "censored," and we only get the inventory time information. We discuss the notion of censoring and its implications in Sec. VI.

Initial processing of inventory files checks the Serial Number (*SN*) and Location of each GPU and creates or updates a separate record for each contiguously observed *SN-Location* combination. Resulting GPU data records are of the form shown in Fig. 3, where records for three are shown. Each record starts with a serial number and locations are coded with c*col-row*c*cage*s*slot*n*node*. The location references are

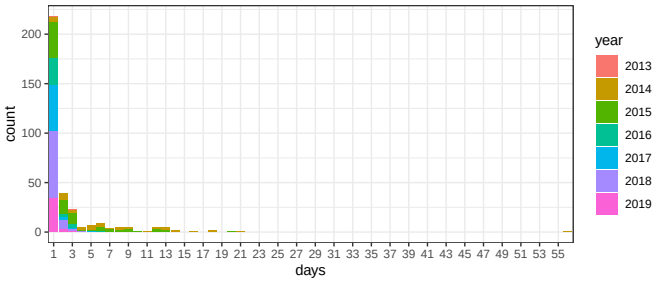

Fig. 2. Inter-inventory times, taken as time intervals between unique GPU insert and remove dates in the data and removing those less than a day.

```
0323812007945 | c17-4c1s3n1 | 09/28/2012 10:29:48 | 02/02/2013 11:32:29
 | c20-6c1s5n2 | 07/23/2019 11:25:33 | 01/20/2020 18:51:10
 | c13-1c1s3n3 | 01/21/2014 10:28:50 | 07/11/2017 18:04:25
 | c0-1c1s3n3 | 10/11/2013 15:57:33 | 10/12/2013 22:09:31
 | c21-1c2s5n0 | 03/19/2013 15:48:11 | 05/29/2013 11:54:11
0325216047736 | c18-4c1s5n1 | 04/09/2017 21:36:19 | 01/20/2020 18:51:10
0323812008856 | c5-4c0s7n0 | 09/30/2012 12:20:00 | 01/25/2013 15:29:58
 | c0-6c1s7n2 | 10/21/2013 14:28:19 | 10/28/2013 17:52:44
 | c3-3c1s5n0 | 05/29/2013 11:54:11 | 05/29/2013 11:54:11
 | c23-6c1s7n2 | 01/21/2014 10:28:50 | 11/02/2018 14:42:34
 | | DBE | 11/02/2018 14:42:34
```

Fig. 3. A few records of raw data produced from inventories and log files that is processed further in our analysis. Note that dates are a mix of EST and EDT.

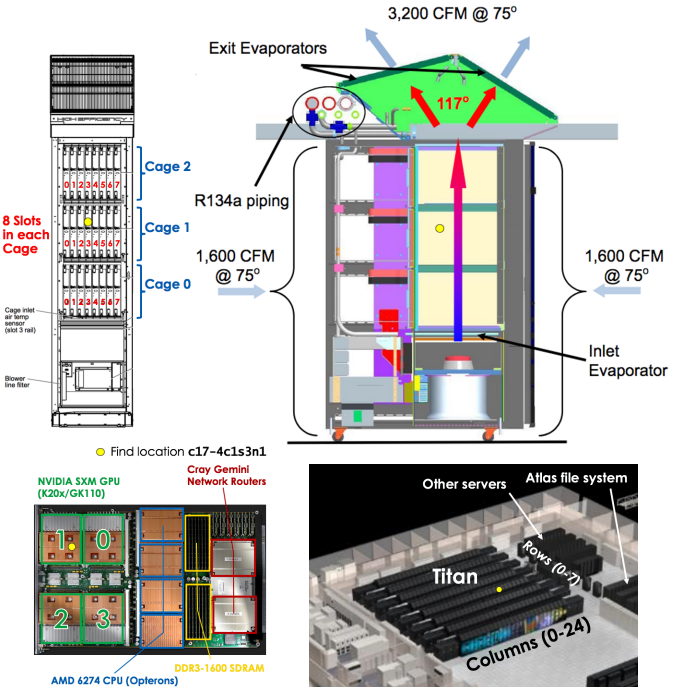

Fig. 4. Titan cabinet (front view and left side view with air flow) and single blade layout (bottom left with *node* numbers) and floor layout (bottom right with *col* and *row* numbers). Cooling airflow is bottom to top within a cabinet, taking ambient air in mostly at the foot and releasing it back out at the top. Cooling fluid is passed through heat exchange evaporators, below Cage 0 and above Cage 2. Ambient air in the room is conditioned at nominal 75° F. The room air system has a dozen intakes near the ceiling and dumps cool air under the floor in several locations. A number of floor tiles are perforated. Yellow dot marks GPU location `c17-4c1s3n1`.

cabinet column (0-24), row (0-7), cage (0-2), slot (0-7), and node (0-3) with respect to the layout shown in Fig. 4. Note that there is an extra empty column of cabinets between columns 10 and 11 to accommodate a few ceiling supports. To aid orientation in the figure, we mark the first location in Fig. 3 `c17-4c1s3n1` with a yellow dot in each view of Fig. 4.

The first GPU record in Fig. 3 shows installation in locations `c17-4c1s3n1`, `c21-1c2s5n0`, `c0-1c1s3n3`, `c13-1c1s3n3`, with periods off the system, and finally in `c20-6c1s5n2`, where it stays until the last inventory file processing run on January 20, 2020. Note that subsequent processing for life spans takes into account that records are not in chronological order and adjusts the last processing run time to "lights out" on Titan on August 1, 2019. The second GPU is installed in location `c18-4c1s5n1`, where it stays until "lights out". The third GPU is first installed in locations `c5-4c0s7n0`, `c3-3c1s5n0`, `c0-6c1s7n2`, and `c23-6c1s7n2`, where a "DBE" is observed on November 2, 2018, and it is not seen again. This is the data set `titan.gpu.history.txt` we make publicly available.

IV. Viewing and Cleaning the Data

As we need to recover durations of GPU operation from this data, correct processing involves time adjustments for switching between daylight saving time and standard time and leap years. We perform this by setting a reference time zone (Eastern time) and converting all date-times from strings into POSIX date-time variables with the R lubridate package [8], which enables appropriate date arithmetic and sensible date constructs for graphs.

As most analysis software relies on rectangular table-like data, we fill the needed repeats of values missing in the raw data (see Fig. 3). To focus our analysis on data after the first two rework cycles in the break-in period, we first reduce the data to the GPUs that were installed near the start of 2014. This was done by removing data for any units with a *remove* date before 2014. After this reduction, there were still six older units remaining, which we also removed to have a clean set of units for the analysis. We do some further processing to handle time overlaps in a tiny fraction (under 0.0007 in GPU life and 0.0002 in location life) of recorded life in the raw records by simply dropping the overlapping lifetimes. A detailed followup of a few of these overlaps found that they are caused by incomplete inventories, where for a moved GPU a node query times out, creating the appearance that a unit is in two locations.

Next, we aggregate into one record per serial number with a total lifetime, the first insert time, the last remove time, and a number of other quantities such as location where the longest time is spent, the proportion of time at the longest location, and the number of DBE or OTB events.

To get some intuition for the GPU lifetimes on Titan, we give two views of 90 randomly selected GPUs and 90 randomly selected locations in Fig. 5. The GPU view visually documents the life of each GPU unit: when it was installed and removed at various locations, its DBE and OTB events, and the last time it was seen. The location view documents the life of a location: when different GPUs were installed and removed, their OTB and DBE events, and whether a removal was the last time the unit was seen on the system. These views were critical to understanding the data and to verifying various data processing decisions.

The third rework cycle was used to label GPUs as *old* batch and *new* batch. The two batches are clearly identifiable in the SN view of Fig. 5 as those first appearing near the 2017 time frame. The *new* batch is formally defined as those with first *insert* date of 2016 or later. More frequent OTB and DBE events are apparent in the old batch. It is also clear from this view that practically all new GPUs stayed at their initial install location whereas the old GPUs were occasionally reinstalled

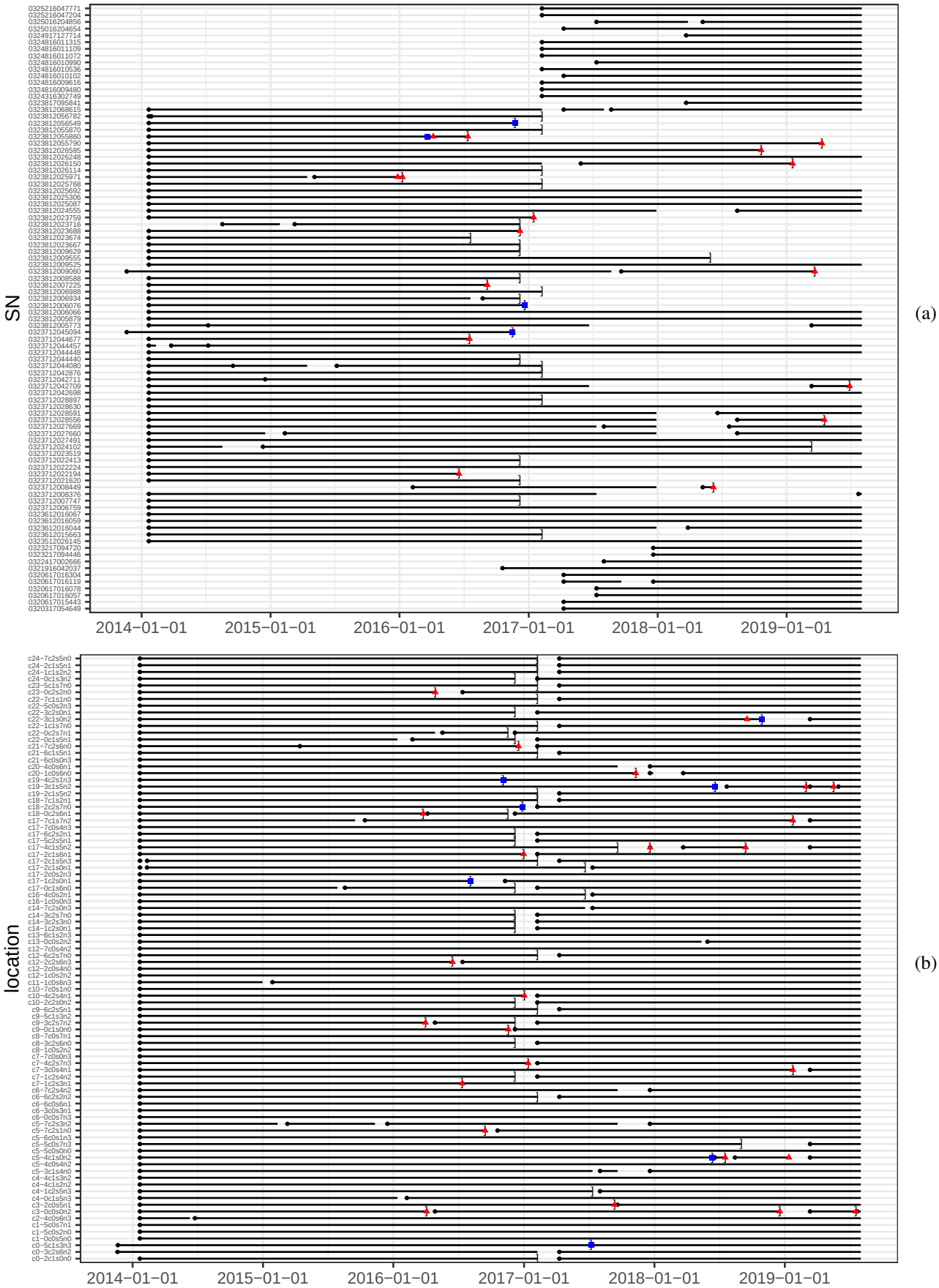

Fig. 5. Serial number view (a) and location view (b) of GPU life and failures. Both SN and locations are randomly selected. Black dots are installs, black lines are lifetimes at installed location, blue squares are OTB events, red triangles are DBE events, and black] are "last seen" events. Such views were critical to understanding the data and to verifying various data processing decisions.

at new locations. Nevertheless, a separate analysis determined that the vast majority of time of the vast majority of units is spent at one location by both the new and the old units.

The location view shows that each location was operational almost all the time with small gaps when GPUs were changed out. It is also notable that OTB and DBE events are associated with a single GPU although four GPUs are together on a blade. The proactive replacements in the rework cycle were done by full blades. Events on single GPUs were usually first swapped for a new blade, the failed GPU was replaced on the blade, and the fixed blade then reused elsewhere. The survival analysis of Sec. VI handles these nuances by appropriate censoring.

V. Time Between Failures Analysis

Using the cleaned data, we analyze the inter-arrival times between the DBE and OTB events. This analysis is done at the device level and at the system level. It provides important insights into the reliability of large-scale machines, where the failure rate of an individual device is significantly different from the overall reliability of the machine.

A histogram of MTBFs measured across GPUs which had at least one failure event is shown in Fig. 6. This is a practical assessment of device reliabilities as opposed to those provided in the device datasheet. The failures are tracked using the *SN* of the GPUs even though a GPU might have been placed at different locations in the machine during its lifetime. The time to the first failure on a device is measured by taking the insert time as the reference point, whereas, a simple difference is taken for subsequent failures. It can be noticed that MTBFs due to DBE and OTB failures of old GPUs are clustered around 2.8 years. This corresponds to the lifetime of most GPUs in the system after accounting for relocations and replacements done in the machine. Almost all of the new batch of GPUs lie below the average since most have a lifetime close to 3 years. We point out that the new batch had a much smaller number of failures (a total of 127 DBEs and OTBs) than the old batch (a total of 5320 DBEs and OTBs). With this high disproportionality, it is difficult to compare device-level

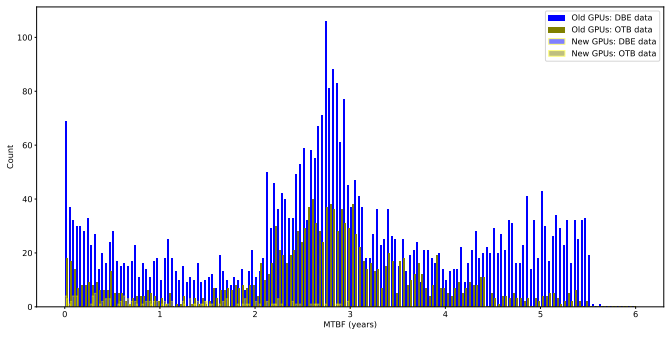

Fig. 6. Distribution of device-level MTBFs due to failures across all GPUs during the lifetime of the machine. The DBE and OTB events are separately plotted to distinguish between the failure pattern of each event type. Data includes both old and new batch of GPUs with the new GPUs having MTBFs scattered between 0 and 3 years as highlighted by yellow highlights in the plot. Each bin in the distribution represents almost 2 weeks.

MTBFs across the new and old batches. On the other hand, it indicates better reliability of the new GPUs as highlighted in the next section.

Apart from the center cluster, many GPUs also have a very low MTBF. This failure characteristic is mainly an artifact from troubleshooting a period of increasing failures in mid to late 2016. After a GPU experienced an OTB or a DBE a second reboot was attempted at the same or possibly different location. Usually, but not always, a second OTB or DBE was generated immediately or possibly after a short time. When moved to a new location, this generates an artifact in our data with an unusually short MTBF because we use the *insert* time as a reference for the first failure on a relocated GPU. This drawback is balanced by the advantage that this way we can discount out of service times which were frequent particularly during early 2017 as can be seen in the location view of Fig. 5.

On the other end of the spectrum, a noticeable portion of GPUs have very high MTBFs and their failures are mostly DBEs. These are the GPUs that see continuous operation in the machine despite its various episodes and see a single DBE event during their lifetime. Apart from this, the distribution of MTBF due to DBE events looks very similar to that due to OTB events. We also note in Fig. 6 that the number of recorded DBE events is much higher than the OTB events. This highlights the targeted replacement of GPUs with OTB events causing a substantial decrease in OTB events.

Now we turn to a system-wide MTBF analysis. Most high-performance computing applications, especially on leadership computing systems such as Titan, use a large fraction of the entire machine in parallel. In the following analysis, failures occurring across all GPUs are consolidated and time between failures is calculated. The old and new GPUs are all considered. At any given time, only a fixed number of GPUs are in the system so any variation in time-between-failures is due to individual device reliabilities.

That said, a single MTBF number does not give an accurate picture of this machine. With many GPU relocations and replacements across the machine over its lifetime, it is best to consider the variability of MTBF across fixed periods. Here, we calculate the mean of quarterly (three months) time-between-failures. Figure 7 shows the considerable change in system-wide MTBF from one period to another. We can see that MTBF tells the different phases that occurred on the machine. Relatively high MTBFs are seen before 2015-Q4, with some quarters not having any OTB events and overall MTBF being determined solely by DBE events. During this phase, we see some MTBF variation from quarter to quarter, but generally, the system MTBF remains higher than one day (33 hours) and reaches as high as 10 days in one instance. An alarming drop in the system MTBF to less than a day, (7.7 hours), is observed in 2015-Q4. A consistent drop in system-wide MTBF is observed starting from 2015-Q2 until 2016-Q4. The corresponding increase in the number of failures during this phase can be seen in Fig. 8, with a peak in 2016-Q4. The usability of the machine comes into question with such low MTBFs and it eventually triggered a phase of installing new

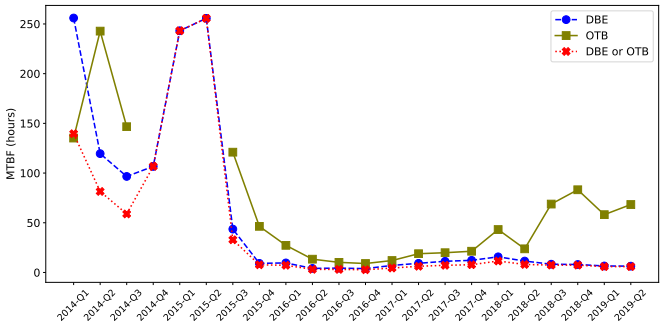

Fig. 7. Variation of system-wide MTBF over the lifetime of the machine. System MTBF is calculated over three month periods to understand the various episodes of the machine. The three month periods are referred to as quarters, i.e., January through March is Q1, and so on. Most GPU replacements started taking place at the end of 2016-Q4 and were completed before 2018-Q1. DBE and OTB events are considered independently as well as together to represent failures occurring on the machine.

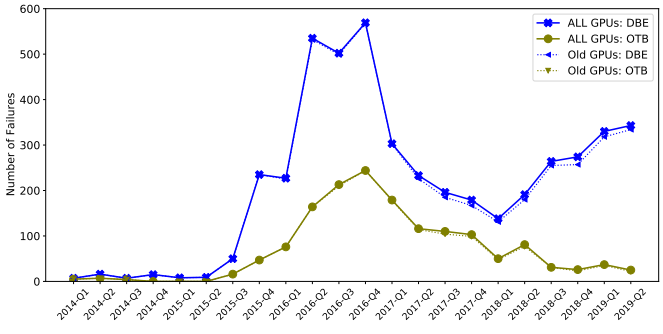

Fig. 8. The number of DBE and OTB failures observed over the lifetime of the machine. A distinction is made between failures on old GPUs to highlight the number of failures on the newer GPUs. The peak failures are seen in 2016-Q4 (813 failures), which marks the commencement of major replacement of GPUs in the machine.

GPUs in the machine.

When GPU replacements start to take place in late 2016, it triggers a slight increase in MTBF. However, this change only lasts until 2018-Q1, when we see another downward trend of system MTBF. Incidentally, 2018-Q1 also marks the completion of all GPU replacements. So the upward trend noted in the period from 2016-Q4 to 2018-Q1 is likely due to phased replacements, each time a portion of the machine being unavailable, thus having a smaller number of GPUs than the full machine in operation. There is no definitive way to incorporate this unavailability of the machine into MTBF analysis, unless we know how the down times were scheduled. The lowest MTBF obtained in 2016-Q4 is 2.7 hours, with subsequent periods having the lowest MTBF of 5.9 hours. With system MTBF less than a day, even slight variations in MTBF make it difficult to reliably run applications despite using failure recovery approaches such as checkpoint restart, as discussed later on in Section VII.

We also separate the DBE and OTB failures in this analysis. After the 2017 replacement of many GPUs, there is a drastic increase in mean-time-between OTB failures, which is also evident in a reduction of OTB failures in Fig. 8. However, the

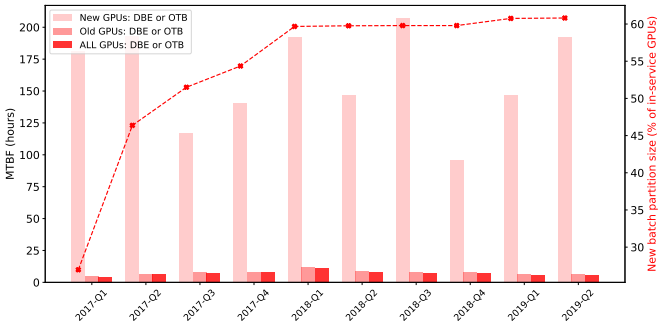

Fig. 9. Variation of system-wide MTBF across hypothetical new and old partitions of the machine after a substantial number of new GPUs were put in service. The dashed line plotted with secondary y-axis shows the size of new partition over time. DBE and OTB events both determine the MTBF. Results highlight the difference in reliability for jobs using the new GPUs as compared to old GPUs, as well as now MTBF is driven by the less reliable old partition despite occupying only a minority portion of the machine.

system MTBF is determined by the weakest link and here the occurrence of DBE events tends to dictate it. Even though the replacements helped to increase the MTBF due to DBE events, the overall system reliability is dictated by the components with the most age in the system. There is a re-emergence of an upward trend towards the end in DBE failures in Fig. 8, which appears to be due to older GPUs in the system.

To better understand the difference in reliability of newer and old portions of the machine, Fig. 9 shows the drastic difference in system MTBF measured in two hypothetical partitions of the machine starting from the time when a major number of GPU replacements had been completed. The Fig. 9 also shows the proportion of the machine with new GPUs over time. The majority of the machine had new GPUs installed starting in 2017-Q3. This new partition of the machine has a 12X better MTBF than the older partition.

For example, in 2018-Q4, the old partition MTBF is about 7.9 hours, whereas the newer partition is 96 hours (4 days). The implications of such a huge disparity on applications running on the system are discussed in Section VII. Moreover, while the old partition is smaller than the new partition, the old partition drives the overall reliability of the machine.

VI. Survival Analysis

Survival analysis (SA) methods [28], sometimes referred to as time to event methods, focus on the effect of the machine on the GPU units. These methods have roots in biomedical statistics, where the effect on the patient is most important. So here we treat the GPUs as patients to determine factors that contribute to failures. SA methods use and combine information across the operational lifetimes of all GPUs. For these analyses, we take apart the location string of each GPU into variables *col, row, cage, slot,* and *node*, and study the influence of the locations on the GPU lifetimes. The construction of a GPU lifetime is more complex than it initially appears because the units are observed only at reboot time, because most units were proactively replaced to prevent failure, and because some

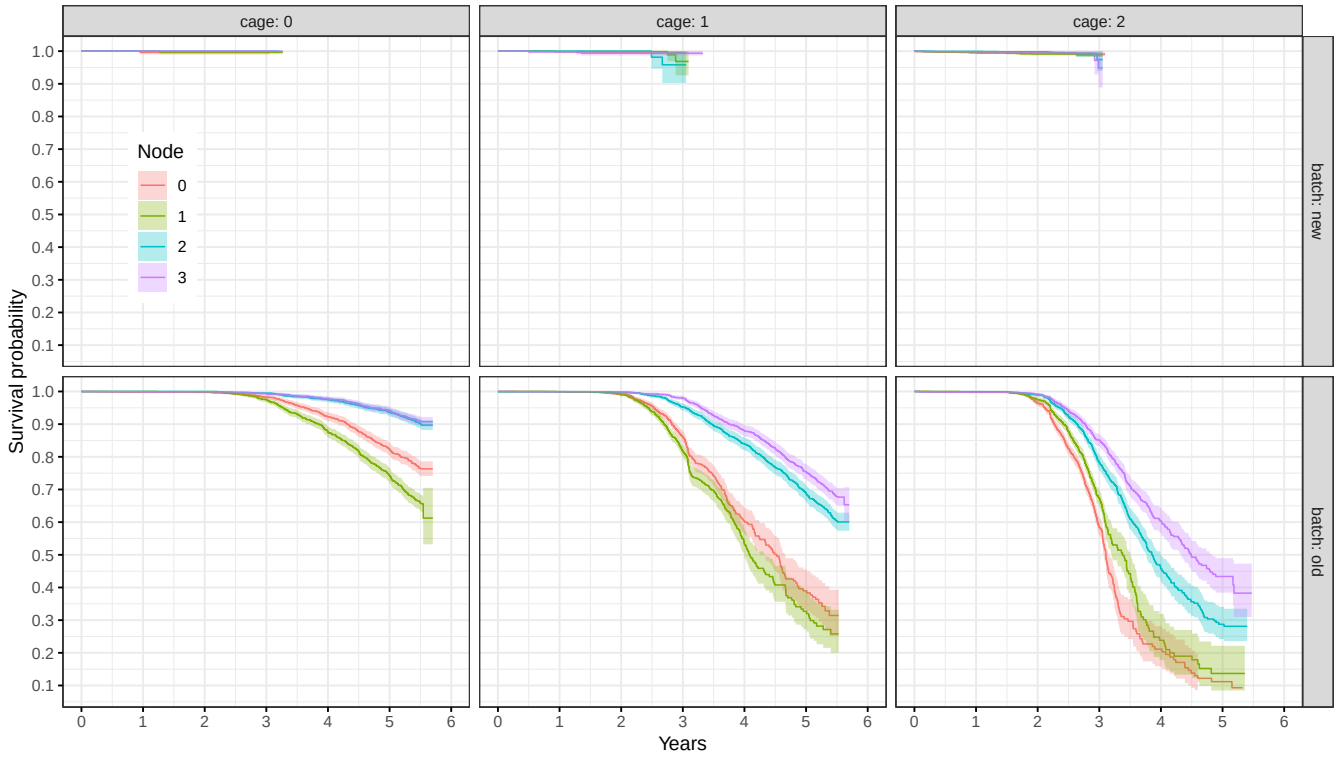

Fig. 10. Comparison of the old and new batches, including the difference of survival probabilities based on `cage` and `node` GPU locations.

units continue in operation after OTB and DBE events (when a second reboot may be successful).

A unit that experiences at least one OTB or DBE event and is removed from the system is considered failed and its operation time until the last seen time is taken as its lifetime. Although most failed units experience one of these events at the last seen time, our definition is not perfect because some units experience OTB or DBE events at a time different from its last seen time. Such units were relatively few so we consider this definition of lifetime as the most pragmatic.

A key concept in survival analysis is censoring, which is about using information from study subjects whose exact failure times are not available or that have not failed. This applies to our study because of proactive GPU replacement before failure, because most units were still in operation when the system was shut down, and also because life spans were recorded only at inventory times. We use censoring concepts on the proactive replacements and on units still in operation at the end. This allows us to use all of the GPU lifetime data, including units that did not fail. But we ignore the inspection time censoring, treating inventory times as exact failure times to reduce the complexity of this analysis. We expect that because of the volume of data and length of operation time, this would not make much difference in our conclusions. However, we are making our data publicly available [24] and expect that others, especially in the survival analysis community, will dive deeper.

Kaplan-Meier survival analysis (KM) [13], [28] starts with computing the probability of survival beyond a given time. It is a nonparametric technique that makes no specific failure model assumptions, such as Weibull, Exponential, etc. The technique is able to use censored observations and can also split the data into subpopulations to compute separate survival curves.

If T is the random variable of a GPU failure time, then its cumulative distribution function $F(t) = Pr\{T < t\}$ gives the probability that a GPU fails by duration t. The survival function is its complement

$$S(t) = Pr\{T \geq t\} = 1 - F(t).$$

It is the probability of being operational at duration t. We use the R packages survival [29] and survminer [14] for the KM analysis, which is reported in Fig. 10. Within each *batch*, separate survival curves are computed for each *cage* by *node* combination. Along with the survival curve estimate, this analysis provides 95% confidence region for survival probability shaded around the curves.

It appears that transport of cooling air provides a complete explanation for relative differences in cage and node survival rates in the old batch. We reach this conclusion with reference to Fig. 4 and relative positions of cages and nodes within a cabinet. Both cage and node differences in survival probabilities can be explained by an inverse relationship with the distance to the bottom of the cabinet, where cooling air is forced through the cabinet to the top. The survival curves can be ordered (*cage 0, cage 1, cage 2*) in decreasing order of survival. As the blades are placed vertically within the cabinet, pairs of

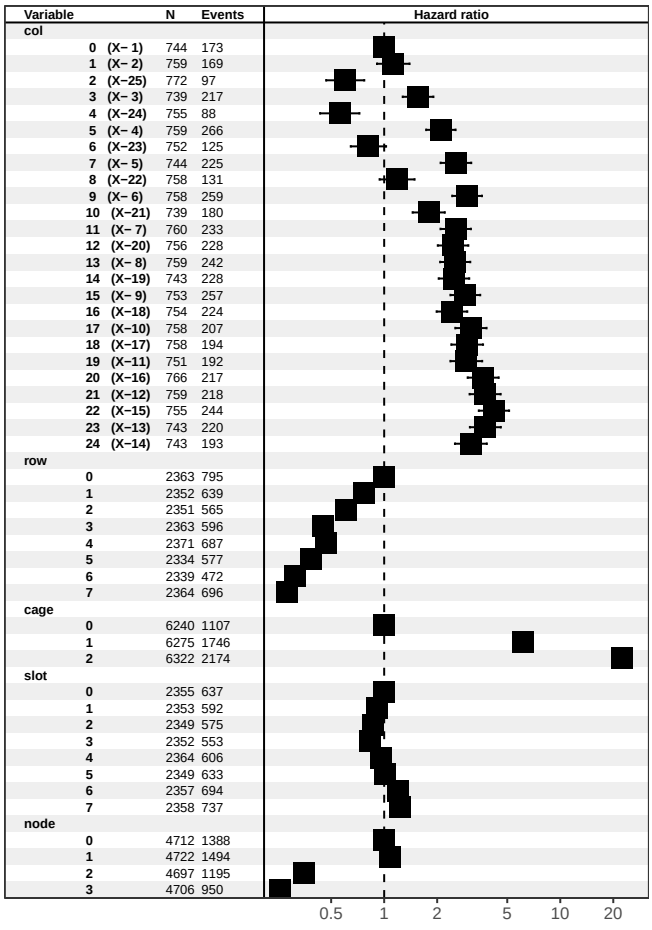

Fig. 11. GPU hazard ratios from Cox regression model on *old* batch. All variables, *col*, *row*, *cage*, *slot*, and *node* are with respect to spatial locations shown in Fig. 4. Notably, the *col* are physical columns of cabinets.

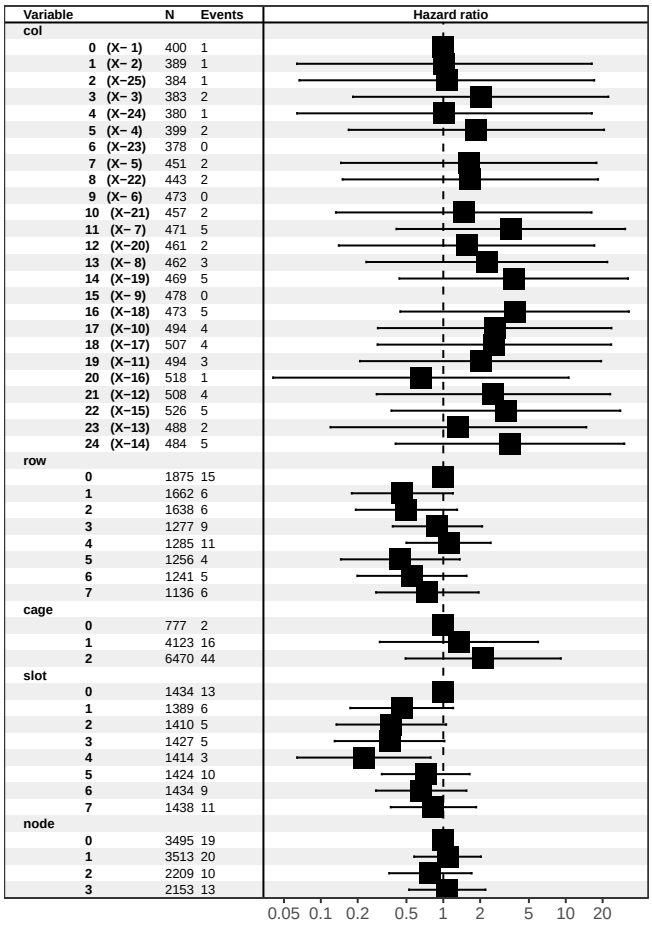

Fig. 12. GPU hazard ratios from Cox regression model on *new* batch.

nodes too experience the same relationship with distance to the bottom, nodes 2 and 3 having lower failure rates than nodes 0 and 1.

An increasing average temperature gradient of about $4°$C per cage is reported in both [30, Fig.10] and [32], independently confirming our airflow temperature interpretation. Findings in [30] also suggest that DBEs may be sensitive to temperature, although their evidence is considered preliminary due to the relatively low number of DBEs in 2014-2015 and high variability of $10°$C to $15°$C in their measured cage replicates.

Very few failures have occurred in the new batch and they are clearly less prone to failure at the 2.5 year mark. There is a slight change at the 3 year mark of cages 1 and 2 but it is also accompanied with a rise in uncertainty and survival is still well above levels in the old batch.

We don't see a "bathtub curve" phenomenon, in fact the opposite is apparent in cages 1 and 2 of the old batch. The slope of the survival curve is related to the hazard rate. There does not seem to be an early "infant mortality" period nor a "wear out" phenomenon at the end. Rather, we see a steeper slope (higher hazard rate) in the middle, associated with the unexpected resistor failures.

To get more comparison power across the locations, we can use a technique that in a sense averages over time. Cox proportional hazards (CPH) regression analysis, can include covariates and estimates relative risk averaged over time based on the covariates [4], [11]. The CPH regression function takes the form

$$h(t) = h_0(t)e^{b_1x_1+b_2x_2...b_kx_k},$$

where x_i are covariates, $h_0(t)$ is the *baseline hazard*, and the b_i are coefficients that measure the impact of the covariates. The quantity e^{b_i} is the hazard ratio (increased average risk over baseline) for covariate i.

CPH is considered a semi-parametric model as there are no assumptions about the shape of the baseline hazard function. Its strongest assumption is that the hazards are proportional. There are a number of tests for this, including graphical diagnostics in the survminer package as well as checking that survival curves for categorical covariates do not cross. We ran these diagnostics and concluded that the hazards for our location categories are approximately proportional. However, survival functions partitioned on *batch* do cross and so we

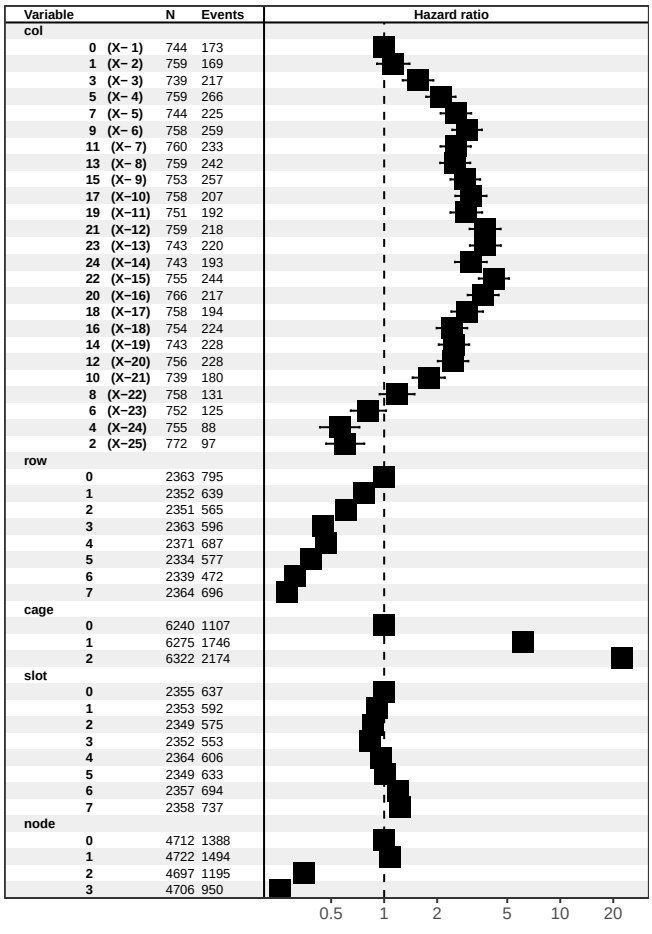

Fig. 13. GPU hazard ratios from Cox regression model on *old* batch, with the *col* variable in interconnect torus X-coordinate order. The torus order removes the peculiar interleaving pattern in Fig. 11 and presents a response that can be explained by an interaction of system cooling and job scheduling.

fit the model to the new and the old batches separately. The results are presented in Fig. 11, and 12.

We find that the average hazard ratios strongly correlate with detailed nuances of the system cooling architecture and are also impacted by job scheduling. The figures give hazard ratios for each of the location variables, giving the ratio for each level compared to its first (0) level (consequently the 0 level is always 1). Due to the exponential nature of the CPH model, the horizontal hazard ratio axis is on a log scale. The estimates include 95% confidence intervals, which too are most reliably computed on a log scale (see, for example, [25]). In the figures, we also include the number of units, N, at risk and the number of events that occurred in each category.

All the factors (*col, row, cage, slot, node*) in the old batch are balanced with respect to the number of units at risk and consequently nearly orthogonal (each level of a factor contains all levels of the other factors) seemingly an almost "designed" experiment nature to this analysis. This is not the case for the new batch, where the cage levels have very different numbers of units at risk. But this also points out that even for the old batch the balance holds only at the outset and as life proceeds, failing units are replaced with new units and the balance degrades because failures are location-dependent.

Our interpretation of correlation with details of the cooling system comes mostly from the hazard ratios for the old batch. The new batch has not had many failure events and the uncertainty bars of nearly all ratios include 1, which is no difference. In the old batch, we see that *cage* has the strongest effect, putting the highest hazard ratio on cage 2, which is consistent with lowest survivals in the KM analysis earlier. Its ratio value near 20 has to be interpreted with caution because it is a time averaged value on a log scale and suffers from the degrading balance mentioned in the previous paragraph. This aspect is not captured by the uncertainty, which accounts for the randomness of the failures but not for the geometric time averaging [12]. Consequently, we interpret the pattern of relative hazards rather than actual ratio magnitudes. More in-depth analysis with penalized estimation methods like [2] can take such balance issues in the exposure history into account and provide time-dependent hazard estimates.

In addition to the strong *cage* and *node* hazard rate differences that increase with distance from the bottom of each cabinet, seen in both KM and CPH results, there is a weaker but peculiar pattern in the *row* and *col* hazard rates. The different behavior in *col* 0-11 from *col* 12-24 is likely due to the presence of additional server systems and the Atlas file system to the right of *col* 24, as can be seen in the floor layout of Fig. 4. The servers used ambient forced air for cooling and did not have additional evaporator heat exchangers, affecting room temperature on the right side of Titan. Anecdotally, this was confirmed by measurements (unfortunately we do not have access to this data) and resulted in biasing more perforated floor tiles on the right side of Titan.

The peculiar interleaving pattern of the first set of columns can be explained by Titan's folded torus interconnect. If we order the columns by their torus X coordinate [6], seen in Fig. 13, the column pattern retains the proximity to other server systems explanation and loses the interleaving pattern peculiarity. Moreover, Titan job scheduler filled temporal and spatial gaps in nodes starting with low torus *X* coordinate. This potentially explains that the low *col* and low *X* coordinate columns have higher hazard due to higher workload than the low *col* and high *X* coordinate columns. Moreover, an impact of scheduling on low *X* coordinate (left) temperature aligns with higher servers-caused room temperature on high *col* numbers (right).

It is possible that the torus Y coordinate can help explain the apparent row effects, but we have difficulty interpreting the row description in [6]. A very minor airflow effect across slots also seems to be present as it mimics faster airflow for the middle slots.

VII. Conclusions

The failure rates of the old batch are not matched by the new batch nor by experience at other facilities with the same type of GPU components, making the failure mode [35] specific

to the component batch installed in the second rework cycle. This unexpected event had considerable impact on operations and on availability of the system. On a positive but ironic note, the corrosion process in the failing resistors made the specific GPU batch act as sensitive instruments for obtaining cumulative trend information from component heat dynamics, giving the current analysis the strong signals observed, and providing lessons learned. Here we discuss the conclusions we can draw from the experience on mitigation, which keeps the system operating at an acceptable level and possibly even restores some of the lost capability, and on long-term planning, which addresses such scenarios in future-generation supercomputers.

A. Mitigation

Mitigation response on Titan included replacing about 11,000 GPUs and changing the job scheduling strategy [36]. This replacement, which amounted to about 59% of Titan's 18,688 GPUs, helped to improve productivity by restoring some of the lost capability, but it was a costly and time-consuming effort. Changing the job scheduling strategy to run larger jobs on more reliable nodes and smaller jobs on less reliable nodes played a crucial role in maintaining productivity at reduced capability. However, jobs running on larger portions of the system, utilizing most or all of its resources, were still impacted by the failing GPUs and corresponding low system MTBF.

The options of replacing failed components and employing reliability-aware resource management may not be available or be cost effective for other supercomputing centers dealing with similar unexpected reliability issues. Replacement components may not be readily available and may have to be manufactured, which can be impossible for a technology that is no longer supported by the original manufacturer. If errors or failures are caused by software, this too can be difficult to fix if the software itself or the deployed version is no longer supported by the vendor.

There are service contract or warranty aspects that may involve more than one manufacturer and it may not be clear cut who has the financial responsibility in a given situation. Did the component fail because it was faulty or because it was placed in a faulty environment? An operational budget of a supercomputing center may not have significant funds to cover replacement or re-engineering costs. Root-cause analysis can be time consuming and require expensive, even research grade expertise, as was needed to find the failing resistors on Titan. On a world class system the same is true of developing a hardware and/or software mitigation strategy. Fixing a problem requires finding and understanding it, which can be difficult in today's complex systems and may require knowledge that only the original manufacturer or vendor possesses.

Reliability-aware resource scheduling also has its limitations, as the network architecture needs to be taken into account. Titan's 3D torus network created more challenges for efficient job allocations than the fat tree network of Titan's successor, Summit [22]. Nodes associated with the same job need to be close to each other on the network for maximum application performance. But node outages or less reliable nodes create resource allocation holes. Separating two lower nodes in each cage 0 on Titan would not create a contiguous partition, yet this group would be the most reliable in 2016 as was shown in the preceding section. Alternatively, partially or completely replacing the aging and failing supercomputer with a new system, even if this solution offers less capability, may be more cost effective in the end.

Other mitigation components for Titan included matching the checkpointing interval of applications with the system's current reliability [1], [31]. The issue here was lack of automated systems to report reliability information and lack of flexibility in application-level checkpoint/restart implementations.

Finally, mitigation is engaged only when we discover there is a problem. Considering the KM analysis shown in Fig. 10, beginning of year 2 (early 2016 for old units) is roughly when the survival probability confidence intervals of nodes in cage 2 begin to separate. Their separation indicates that the nodes in cage 2 are different even after accounting for random variation. A statistical comparison of cages 0 and 2 would likely show an earlier signal. The use of more advanced statistical modeling techniques on future systems, especially when combined with more diagnostic data collection, can lead to earlier detection of a problem and deeper insight into its causes. The statistical data processing, visualization, and modeling tools used in this paper can be scaled to large distributed systems [3], [27] to keep up as data collection on such systems ramps up.

B. Future Systems

The long-term planning component is one of the biggest lessons learned from the Titan reliability experience. Today's supercomputers are designed to deal with expected reliability issues. However, history has shown [7] that unexpected reliability issues do occur and do have a significant impact. Vendors and manufacturers obviously can not mitigate against all possible reliability threats, however, a resilience strategy is needed for future-generation systems that is able to deal with emerging unexpected reliability threats in a reasonable and cost effective way. In addition to mitigation, better support for automated real-time reliability monitoring and reporting is needed, including for root-cause analysis.

Supercomputer center policies regarding reliability monitoring, resource allocation and checkpointing strategies need to be powerful and flexible enough to facilitate mitigation for such unexpected reliability threats. System acquisition contracts may need to include performance requirements for degraded operation, such as a certain percentage of performance capability if the MTBF drops by an order of magnitude.

The reliability issues that Titan experienced had a direct impact on the development and deployment of Titan's successor, Summit, and even on Summit's successor, Frontier. More and better monitoring data is already collected on Summit and discussions about the use of more advanced statistical methods have already started. NVIDIA's GPU management

and monitoring software has been significantly improved. Temperature monitoring has been significantly improved as well.

VIII. Acknowledgments

This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

This work was supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Resilience for Extreme Scale Supercomputing Systems Program, with program managers Robinson Pino and Lucy Nowell.

We are very grateful to four anonymous reviewers who provided detailed and insightful comments on our initial submission, which resulted in a greatly improved presentation and deeper conclusions.

References

[1] L. Bautista-Gomez, A. Gainaru, S. Perarnau, D. Tiwari, S. Gupta, C. Engelmann, F. Cappello, and M. Snir, "Reducing waste in extreme scale systems through introspective analysis," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2016, pp. 212–221. [Online]. Available: https://doi.org/10.1109/IPDPS.2016.100

[2] A. Bender, F. Scheipl, W. Hartl, A. G. Day, and H. Küchenhoff, "Penalized estimation of complex, non-linear exposure-lag-response associations," *Biostatistics*, vol. 20, no. 2, pp. 315–331, Apr. 2019. [Online]. Available: https://doi.org/10.1093/biostatistics/kxy003

[3] W. Chen, G. Ostrouchov, D. Schmidt, P. Patel, and H. Yu, "pbdMPI: Programming with Big Data–Interface to MPI," *R Package*, 2012. [Online]. Available: http://cran.r-project.org/package=pbdMPI

[4] D. R. Cox, "Regression models and life-tables," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 34, no. 2, pp. 187–220, 1972.

[5] 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/Q system," in *49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 2019*, Portland, OR, USA, Jun. 24-27, 2019, pp. 473–484. [Online]. Available: https://doi.org/10.1109/DSN.2019.00055

[6] M. A. Ezell, "Understanding the impact of interconnect failures on system operation," in *Proceedings of the Cray User Group Conference (CUG) 2013*, Napa Valley, CA, USA, May 6-9, 2013.

[7] A. Geist, "How to kill a supercomputer: Dirty power, cosmic rays, and bad solder," *IEEE Spectrum*, Feb.23, 2016. [Online]. Available: https://spectrum.ieee.org/computing/hardware/how-to-kill-a-supercomputer-dirty-power-cosmic-rays-and-bad-solder

[8] G. Grolemund and H. Wickham, "Dates and times made easy with lubridate," *Journal of Statistical Software*, vol. 40, no. 3, pp. 1–25, Apr. 2011. [Online]. Available: http://www.jstatsoft.org/v40/i03/

[9] S. Gupta, D. Tiwari, C. Jantzi, J. Rogers, and D. Maxwell, "Understanding and exploiting spatial properties of system failures on extreme-scale HPC systems," in *45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 2015*, Rio de Janeiro, Brazil, Jun. 22-25, 2015, pp. 37–44. [Online]. Available: https://doi.org/10.1109/DSN.2015.52

[10] S. Gupta, T. Patel, C. Engelmann, and D. Tiwari, "Failures in large scale systems: Long-term measurement, analysis, and implications," in *Proceedings of the* 30^{th} *IEEE/ACM International Conference on High Performance Computing, Networking, Storage and Analysis (SC) 2017*, Denver, CO, USA, Nov. 12-17, 2017, pp. 44:1–44:12. [Online]. Available: https://doi.org/10.1145/3126908.3126937

[11] F. E. Harrell, *Cox Proportional Hazards Regression Model*. Springer International Publishing, 2015, pp. 475–519. [Online]. Available: https://doi.org/10.1007/978-3-319-19425-7

[12] M. A. Hernán, "The hazards of hazard ratios." *Epidemiology*, vol. 21 1, pp. 13–15, 2010. [Online]. Available: https://doi.org/10.1097/EDE.0b013e3181c1ea43

[13] E. L. Kaplan and P. Meier, "Nonparametric estimation from incomplete observations," *Journal of the American Statistical Association*, vol. 53, no. 282, pp. 457–481, 1958. [Online]. Available: https://doi.org/10.2307/2281868

[14] A. Kassambara, M. Kosinski, and P. Biecek, *Drawing Survival Curves using 'ggplot2'*, 2019, r package version 0.4.6. [Online]. Available: https://CRAN.R-project.org/package=survminer

[15] M. Kumar, S. Gupta, T. Patel, M. Wilder, W. Shi, S. Fu, C. Engelmann, and D. Tiwari, "Understanding and analyzing interconnect errors and network congestion on a large scale HPC system," in *Proceedings of the* 48^{th} *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 2018*, Luxembourg City, Luxembourg, Jun. 25-28, 2018, pp. 107–114. [Online]. Available: https://doi.org/10.1109/DSN.2018.00023

[16] C. D. 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 (DSN) 2014*, Atlanta, GA, USA, Jun. 23-26, 2014, pp. 610–621. [Online]. Available: https://doi.org/10.1109/DSN.2014.62

[17] E. Meneses, X. Ni, T. Jones, and D. Maxwell, "Analyzing the interplay of failures and workload on a leadership-class supercomputer," in *Proceedings of the Cray User Group Conference (CUG) 2015*, Chicago, IL, USA, Apr. 26-30, 2015. [Online]. Available: https://www.researchgate.net/publication/276290607

[18] H. Meuer, E. Strohmaier, J. Dongarra, and H. Simon, "Top 500 List of Supercomputer Sites," 2020. [Online]. Available: http://www.top500.org

[19] 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 (HPCA) 2016*, Barcelona, Spain, Mar. 12-16, 2016, pp. 519–530. [Online]. Available: https://doi.org/10.1109/HPCA.2016.7446091

[20] 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 *Proceedings of the* 25^{th} *IEEE International Symposium on the Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS) 2017*, Banff, AB, Canada, Sep. 20-22, 2017, pp. 22–31. [Online]. Available: https://doi.org/10.1109/MASCOTS.2017.12

[21] B. Nie, J. Xue, S. Gupta, T. Patel, C. Engelmann, E. Smirni, and D. Tiwari, "Machine learning models for GPU error prediction in a large scale HPC system," in *Proceedings of the* 48^{th} *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 2018*, Luxembourg City, Luxembourg, Jun. 25-28, 2018, pp. 95–106. [Online]. Available: https://doi.org/10.1109/DSN.2018.00022

[22] Oak Ridge Leadership Computing Facility, "Summit supercomputer," 2020. [Online]. Available: https://www.olcf.ornl.gov/olcf-resources/compute-systems/summit/

[23] ——, "Titan supercomputer," 2020. [Online]. Available: https://www.olcf.ornl.gov/olcf-resources/compute-systems/titan/

[24] G. Ostrouchov, D. Maxwell, R. A. Ashraf, C. Engelmann, M. Shankar, and J. H. Rogers, "Titan supercomputer GPU reliability data 2012-2019 and code to reproduce analysis," DOI, 2020. [Online]. Available: https://github.com/olcf/TitanGPULife

[25] G. Ostrouchov and W. Q. Meeker, Jr., "Accuracy of approximate confidence bounds computed from interval censored Weibull and log-normal data," *Journal of Statistical Computation and Simulation*, vol. 29, no. 1, pp. 43–76, May 1988. [Online]. Available: https://doi.org/10.1080/00949658808811050

[26] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2020. [Online]. Available: https://www.R-project.org/

[27] D. Schmidt, W.-C. Chen, M. A. Matheson, and G. Ostrouchov, "Programming with BIG data in R: Scaling analytics from one to thousands of nodes," *Big Data Research*, vol. 8, pp. 1 – 11, 2017. [Online]. Available: https://doi.org/10.1016/j.bdr.2016.10.002

[28] Terry M. Therneau and Patricia M. Grambsch, *Modeling Survival Data: Extending the Cox Model*. New York: Springer, 2000. [Online]. Available: https://doi.org/10.1007/978-1-4757-3294-8

[29] T. M. Therneau, *A Package for Survival Analysis in R*, 2020, version 3.1-12. [Online]. Available: https://cran.r-project.org/package=survival

[30] D. Tiwari, S. Gupta, J. Rogers, D. Maxwell, P. Rech, S. Vazhkudai, D. Oliveira, D. Londo, N. DeBardeleben, P. Navaux, L. Carro, 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 (HPCA) 2015*, San Francisco, CA, USA, Feb. 7-11, 2015, pp. 331–342. [Online]. Available: https://doi.org/10.1109/HPCA.2015.7056044

[31] D. Tiwari, S. Gupta, and S. S. Vazhkudai, "Lazy checkpointing: Exploiting temporal locality in failures to mitigate checkpointing overheads on extreme-scale systems," in *44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) 2014*, Atlanta, GA, USA, Jun. 23-26, 2014, pp. 25–36. [Online]. Available: https://doi.org/10.1109/DSN.2014.101

[32] 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 *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC) 2015*, Austin, TX, USA, 2015. [Online]. Available: https://doi.org/10.1145/2807591.2807666

[33] D. Tiwari, S. Gupta, J. Rogers, and D. Maxwell, "Experience with gpus on the titan supercomputer from a reliability, performance and power perspective," in *Proceedings of the Cray User Group Conference (CUG) 2015*, Chicago, IL, USA, Apr. 26-30, 2015. [Online]. Available: https://www.osti.gov/biblio/1265578

[34] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[35] O. L. Vargas, S. Valdez, M. Veleva, K. R. Zlatev, W. M. Schorr, and G. J. M. Terrazas, "The corrosion of silver in indoor conditions of an assembly process in the microelectronics industry," 2009. [Online]. Available: https://doi.org/10.1108/00035590910969347

[36] C. Zimmer, D. Maxwell, S. McNally, S. Atchley, and S. S. Vazhkudai, "GPU age-aware scheduling to improve the reliability of leadership jobs on Titan," in *International Conference for High Performance Computing, Networking, Storage and Analysis (SC) 2018*, Dallas, TX, USA, Nov. 11-16, 2018, pp. 83–93. [Online]. Available: https://doi.org/10.1109/SC.2018.00010