Towards a Model to Estimate the Reliability of Large-Scale Hybrid Supercomputers

Elvis Rojas[1,2(✉)], Esteban Meneses[2,3(✉)], Terry Jones[4(✉)], and Don Maxwell[4(✉)]

[1] National University of Costa Rica, San José, Costa Rica
erojas@una.ac.cr
[2] Costa Rica Institute of Technology, Cartago, Costa Rica
emeneses@cenat.ac.cr
[3] Costa Rica National High Technology Center, San José, Costa Rica
[4] Oak Ridge National Laboratory, Oak Ridge, TN, USA
{trjones,maxwellde}@ornl.gov

Abstract. Supercomputers stand as a fundamental tool for developing our understanding of the universe. State-of-the-art scientific simulations, big data analyses, and machine learning executions require high performance computing platforms. Such infrastructures have been growing lately with the addition of thousands of newly designed components, calling their resiliency into question. It is crucial to solidify our knowledge on the way supercomputers fail. Other recent studies have highlighted the importance of characterizing failures on supercomputers. This paper aims at modelling component failures of a supercomputer based on Mixed Weibull distributions. The model is built using a real-life multi-year failure record from a leadership-class supercomputer. Using several key observations from the data, we designed an analytical model that is robust enough to represent each of the main components of supercomputers, yet it is flexible enough to alter the composition of the machine and be able to predict resilience of future or hypothetical systems.

Keywords: Fault tolerance · Resilience · Failure analysis · Failure modelling

M. Malawski and K. Rzadca (Eds.): Euro-Par 2020, LNCS 12247, pp. 37–51, 2020.
https://doi.org/10.1007/978-3-030-57675-2_3

1 Introduction

Large-scale machines provide a valuable tool to push the envelope in many scientific disciplines. From unveiling the mysteries of the universe formation to making sense of the myriad data in the global economy, supercomputers are indispensable. Getting more powerful every year, supercomputers barely keep up with the insatiable need for computing in scientific simulations and data analysis. To maintain the required growth in computing power, hardware engineers have employed increasingly complex and heterogeneous designs. Modern supercomputers assemble an immense amount of processors, accelerators, memory modules, and more parts. The inevitable consequence of such arrangement is a threateningly high failure rate [21,22]. Therefore, it is mandatory to understand the reliability of supercomputers to sustain the rate of scientific discovery.

The last decade has seen several meetings, studies, and reports about supercomputer reliability [3,21,22]. An inter-agency report [21] found that one high priority area was *fault characterization*. As technologies become more complex to provide high scalability, reliability becomes more difficult. Therefore, it is crucial to describe failure types along with their frequency and impact. A meeting on failures at exascale level computing [22] also highlighted the importance of such characterization, but insisted on building strong statistical models for failure analysis and the development of fault tolerant algorithms. Finally, other study [3] recommended exploring future failure characterization paradigms to guide the selection of hardware components for future machines. This paper addresses the concerns of the community by providing a reliability model for supercomputers based on the failure characterization of hardware components.

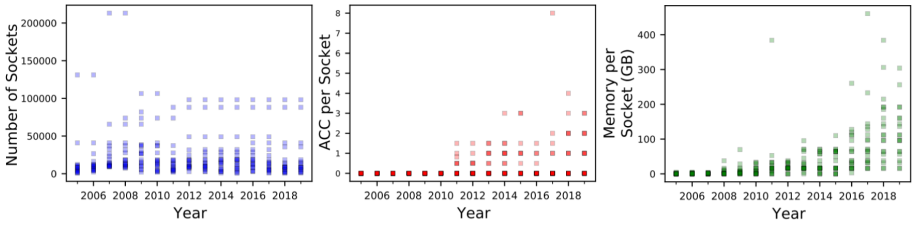


Fig. 1. Features in recent supercomputers. The top 20 supercomputers from the last 15 years show the number of processor sockets has stagnated, but accelerators and memory size per socket continues to increase.

We focus our model on the study of failure rates of three components: processors, accelerators, and memory. Figure 1 shows a historical view of the integration of these components on the top 20 machines of the Top 500 list [25] for the last 15 years. The left part shows the number of processor sockets in each machine. That number increased initially, but stagnated at around 100,000 sockets. However, the number of accelerators per socket has been on the rise, as depicted by the middle figure. The memory size per socket has swelled at a faster pace.

Here then is a list of the contributions of this paper:

- A collection of insights on a five-year failure record of a leadership-class supercomputer is provided in Sect. 2. One of these findings include the trends or epochs in the failure data, automatically detected by an algorithm.
- A whole-system failure model using Mixed Weibull distributions in Sect. 3. Our model outperforms the traditional Weibull distribution and allows the representation of different configurations according to the prominence of a component in the machine: processors, accelerators, or memory.
- Failure rate predictions for different hypothetical exascale machine configurations in Sect. 3.2. Such projections are correlated with power consumption of each configuration to understand the trade-offs between performance and energy.

1.1 Related Work

Several studies have analyzed the behavior of failures in large-scale systems [3–5,11,12,16,18,19,29], including studies that analyzed failures of specific supercomputer components such as GPU [15,23,24] or memory [1,10,20]. This paper also analyzes large-scale system failures, but with the distinctive focus on building a reliability model to understand and project system behavior.

The literature contains studies of modelling the reliability of large-scale systems. In [9], the authors used modelling to examine the impact of failure distributions on application performance. They used the *flexible checkpoint model* (FCM) to determine the application execution time and the optimal checkpoint interval. In [27], the authors developed performance models to predict the application completion time under system failures. Another modelling study [14] analyzed failure traces from five large multi-site infrastructures to model failures and generate failure scenarios. Other researchers modelled the failure behavior using signal analysis theory [6]. They characterized each signal and proposed corresponding models, merging all the information to offer an overview of the whole system. In [7,28], the failure correlation in time and space was analyzed. The time-varying behavior of failures was modelled focused on peak failure periods. The authors characterized the duration of peaks, the peak inter-arrival time, and the duration of failures during peaks. Regarding the space-correlated failures, the model considered groups of failures that occur within a short time interval. With modelling, they found that space correlated failures are dominant in terms of resource downtime in seven of the analyzed systems.

Similar to this paper, all those previous studies have concentrated on modelling failures in large-scale systems. Modelling was used for the following purposes: analyze correlations, predict execution times, compute optimal checkpoint intervals, and analyze performance optimizations. In this paper, we analyze and classify the system failures by component. Based on that, we model failures based on the Mixed Weibull distribution to finally make reliability projections using different system configurations. This paper differs both in the approach and the purpose of modelling. Almost all studies used small datasets with under 1.5 years

or hypothetical extreme-scale systems [9]. We base our results on real-life data from a 5-year failure record from a leadership-class supercomputer.

Another study on modelling failures was described in [8]. They analyzed five years of system logs to model hardware failures of multiple heterogeneous components. They modelled each component and developed integrated failure models given the component usage. They divided the event data into epochs due to missing data in the event log. Before the modelling stage, a statistical analysis of failures was performed. That modelling study differs from our work in multiple dimensions: we analyzed a failure dataset of five consecutive years of a modern hybrid supercomputer, we automatically determined epochs using a time series analysis algorithm, the reliability model was implemented using Mixed Weibull distributions, and we presented a series of failure projections based on different hardware configurations.

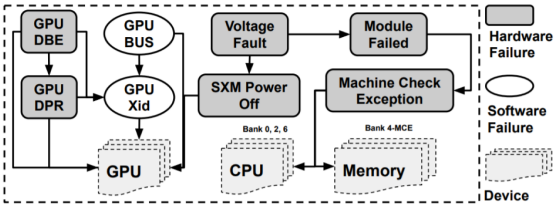
In our previous work [13,17], we developed the process to ingest the raw failure data and derive human understandable information from the Titan failure set. In this work, we introduce an analysis that reveals distinctive epochs of failure rates; we delve into these epochs and uncover a number of interesting observations peculiar to each; and, most importantly, we introduce a new mathematical model that is shown to categorize failures more accurately than other proposed models as determined by the Kolmogorov-Smirnov goodness of fit test.

2 Insights from a Real-Life Hybrid Supercomputer

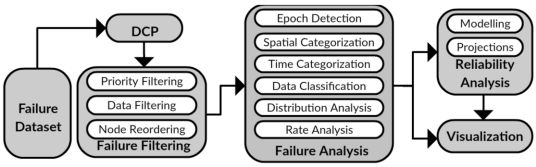
2.1 Failure Dataset

We analyzed failure events from Titan supercomputer. Titan was a Cray XK7 system located at the Oak Ridge Leadership Computing Facility (OLCF) and was one of the earliest supercomputers that used a hybrid architecture (CPU and GPU). It had 18,688 nodes and each node had an AMD 16-core Opteron CPU (299,008 cores in the whole system), an NVIDIA Tesla K20 GPU and 32 GB of main memory. Titan had a peak performance of 17.59 petaFLOPs and by the time it was decommissioned, it was in the ninth position according to the Top500 ranking [25]. Every abnormal incident on Titan was automatically registered into a failure database. The database was automatically constructed by a program designed by the system administrators that used the SEC (simple event correlator) program [26]. Using correlation rules, SEC analyzes output streams from each node and merges multiple reports of the same incident into a single database entry.

In this work, we analyzed five full years of failure events on Titan from 2014 to 2018. In this five-year span, the total number of events in the failure database was 2,663,512. After a filtering stage, the number of events in the failure database was dramatically reduced by 99.78%. The remaining 0.22% of events correspond to what we describe as 5654 *unique failures*, distributed as 565, 649, 1824, 1291, and 1325 events for years 2014–2018, respectively. This massive reduction in the event database is due to the presence of multiple redundant messages and warnings for the same failure.



(a) Failure propagation.



(b) Stages of the methodology.

```
Input: data[] // failure data time series
Input: N  // epoch granularity
Output: epochs[]  // list of meaningful epochs
data_ema[] ← Compute_EMA(data[])  // computing exponential moving average
data_signals[] ← Compute_B&B(data_ema[], data[])  // computing bull&bear signals
fragments[] ← Compute_fragments(data_signals[])  // computing epoch fragments
segments[] ← Compute_segments(fragments[],N)  // computing epoch segments
 // computing linear trend of each segment
foreach s ∈ segments[] do
    e ← Extract_epoch(data[],s)  // extracting original epoch values from data
    line ← Compute_linear_fit(e)  // compute least squares linear fit
    epochs[] ← epochs[] ∪ {e,line}  // adding new epoch
end
```

(c) Algorithm to detect trends in time series.

Fig. 2. Data analysis methodology.

Figure 2a shows this redundancy as the cause-effect dependency of events on the database. These are potential dependencies because in some cases an event can occur in isolation without the occurrence of the preceding event in the graph. For instance, a GPU DBE (double bit error) failure may generate a GPU DPR (double page retirement) failure and then a GPU XID failure (a general software GPU error). In that case, the filtering process only considers the failure with the highest priority (i.e. GPU DBE) and discards the other derived failures. Nevertheless, in some cases, the GPU XID or GPU DPR failures can be generated in isolation. Relevant data on discarded failures is attached to the highest priority failure, to avoid losing information. Figure 2a shows three types of hardware components (GPU, CPU, and memory) and how these components might be affected by system failures.

2.2 Methodology

Figure 2b summarizes the stages of the methodology, implemented as a collection of Python scripts and available at https://github.com/elvinrz/FailureAnalysis.

The first stage, *Data cleaning and Preprocessing (DCP)*, prepares all input files in a consistent format. The second stage, *Failure Filtering*, takes the preprocessed data and performs a series of tasks to filter redundant data. The priority filtering task is used to remove redundant data with less priority or that depends on other events. The others two tasks were used to remove non-significant events, such as heartbeat faults, which are considered as warnings by system administrators. This paper focuses on system failures only. Consequently, user failures were discarded. The total number of user failures removed was 816,826, representing 30% of the total number of events from 2014 to 2018.

The third stage, *Failure Analysis*, uses the filtered data and performs a series of analysis to model the behavior of failure events in the system. We performed data fitting with three distributions (Weibull, Exponential and Lognormal) and we used the Kolmogorov-Smirnov Goodness of Fit Test to determine how close the data fits a statistical distributions. Algorithm 2c presents an adjusted procedure to model the time series trends. The algorithm performs an exponential moving average (EMA) and it uses a least squares polynomial fit to calculate the trend segments. The algorithm outputs trends of failures event segments throughout the years. Therefore, we are able to automatically detect epochs in the data.

The *Reliability Analysis* stage is used to describe the background of the statistical model that was used in this study. We implemented the Mixed Weibull distribution to analyze real and synthetic failure data. In addition, Mixed Weibull distributions were used to perform a series of projections when changing the proportion of the system components (CPU, GPU and memory) to determine the reliability and the power consumption of exascale machines. The *Visualization* stage displays the results of the previous analyses. We plot all necessary visualizations to show categorizations, correlations, and probability distributions.

2.3 Insights

Observation #1*: Most system failures in Titan are processor, accelerator, or memory related.* Table 1 shows the failure distribution of hardware components according to failure classification on Fig. 2a. The failures related to processor, accelerator, and memory represent 92.45% of all failures.

Table 1. Failure count by category and epochs.

Category	Type	E1	E2	E3	E4	%
GPU	XID, DBE, BUS, DPR, SXM P. Off	933	834	1291	1508	80.87
CPU	Machine Check Excep. (Bank 0, 2, 6)	33	9	5	14	1.11
Memory	Machine Check Excep. (Bank 4, MCE)	214	80	124	173	10.49
Total						92.45

Observation #2*: GPU failures are dominant, particularly those associated to GPU memory.* Table 1 shows that 80.87% of all failures are related to GPU. The share of GPU memory failures (DBE, DPR) from the total amount is 52.08%.

Observation #3*: The time series of GPU failures can be divided into four distinctive epochs.* Figure 3 shows the result of using Algorithm 2c on the failure time series of three hardware components. The GPU time series presents four epochs: three blue segments representing a trend to increase failure rate and a single orange segment representing the decrease of failure events. This result is a refinement of a previous composition manually made by experts using the same data [17].

Observation #4*: The time series of processor and memory failures have a single epoch.* The result of applying Algorithm 2c to the CPU and memory failure time series reveals only one epoch for each. Figure 3 depicts this result.

Observation #5*: Epochs 2 and 3 on GPU failure time series reflect abnormal behavior of hardware components.* Table 1 reports the number of GPU failures in each epoch. Epochs 2 and 3 together are composed of 63 weeks (24.23% of total weeks) and contain 46.53% of the GPU failures. In contrast, Epoch 1 is composed of 42% of total weeks and only has 20.4% of GPU failures. Epoch 4 has a similar behavior as Epoch 1. According to the system administrators of Titan, epochs 2 and 3 represent abnormal behavior due to a massive failure of GPU components (Epoch 2) and the replacement of those parts (Epoch 3).

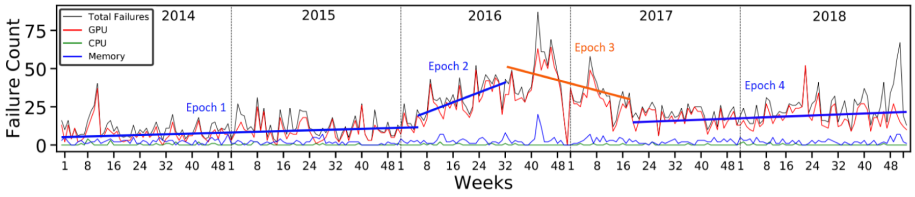
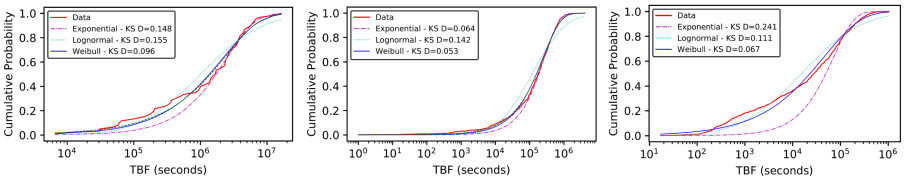


Fig. 3. Failure time series 2014–2018 for the three main hardware components.

Observation #6*: Hardware component failures are statistically independent.* We tested the possible dependence between failures of different hardware components. Figure 2a shows the failure categorization by component. Using a time window of 300 s, we counted if there was a couple of events from different components in the same time window. This analysis was performed before the filtering process to take into account all events of interest (heartbeat faults and user failures were excluded). The results show the total number of correlated failures is 25 out of 5218 total failures. Such a minute portion statistically rules out any correlation and, as a consequence, any dependence.

Observation #7*: Time between failures of processor and memory components follows a Weibull distribution.* We studied the *Cumulative Distribution Function*



(a) CPU failures. **(b)** Memory failures. **(c)** GPU failures.

Fig. 4. Cumulative distribution function for CPU, memory, and GPU failures.

(CDF) of the MTBF data. Three different distributions were tested (exponential, lognormal and Weibull). We used the *Kolmogorov-Smirnov Goodness of Fit Test (KST)* to determine which distribution better models the MTBF data. For CPU and memory components, we used the four epochs to perform the distribution analysis because the failures of these two components were significantly less than the GPU failures. Figure 4 shows the CDF of the three components, and we see that the *Weibull* distribution fits better the MTBF data than *LogNormal* and *Exponential* distributions. Low D values resulting from the KST represent a better fit. In all cases the KST of the *Weibull* distribution was the smallest with values of D equals to 0.096 for the CPU, and 0.053 for the memory. We do not reject the null hypothesis (the data comes from a specific distribution) because the computed D values were lower than the critical values.

Observation #8*: Time between GPU failures follows a Weibull distribution.* For the GPU time series, the four epochs were analyzed and in all epochs the Weibull distribution was the best fit. The KST test resulted in $D = 0.067$ for the GPU at Epoch 1. In light of Observation #5, for the rest of the paper, we use only Epoch 1 data for modelling.

3 Modelling Reliability of Hybrid Supercomputers

3.1 Analytical Model

The theory of reliability provides a rich framework to study, analyze, and model failure data from supercomputers. In the literature, the mean-time-between-failures (MTBF) is a popular metric to describe the reliability of large-scale machines [3,21,22]. To find such value, it is necessary to develop a model. The *reliability function* is the most frequently used function to perform life data analysis. It provides the probability of a component functioning with no failure for an amount of time. It is a function of time and a flexible way to derive the MTBF of a system. A key element to build a precise reliability function for a system is to find an appropriate distribution function for life data analysis. For instance, finding a distribution function for the probability of a component failing at time t. Let us call $f(x)$ this distribution function. Using $f(x)$, we compute $U(t) = \int_0^t f(s)ds$ as the probability of a component failing by time t.

Function $U(t)$ is called the *unreliability function* and it is basically the cumulative distribution function of $f(x)$. Finally, the reliability function $R(t)$ can be derived using $U(t) + R(t) = 1$. In the rest of the paper, we concentrate on finding a function $f(x)$ that best fits the data and provides a precise reliability function for the system. The literature shows that Weibull distribution is a good fit for supercomputer reliability data on time between failures [17,19,24]. The Weibull probability distribution function is given by $f(x;\alpha,\beta) = \frac{\alpha x^{\alpha-1}}{\beta^{\alpha}} e^{-(\frac{x}{\beta})^{\alpha}}$, where α is the *shape* parameter and β is the *scale* parameter. A value of $\alpha < 1$ points to a decreasing failure rate, $\alpha = 1$ means the failure rate is constant (in which case the Weibull distribution equals an Exponential distribution), and $\alpha > 1$ indicates an increasing failure rate. The scale parameter β represents how spread out the distribution is.

Although a Weibull distribution may adequately capture the failure data of *all components* in a supercomputer, it may fall short for modelling scenarios where the behavior of the components differ from one another. For instance, Sect. 2 presented a case where several components show different failure profiles. For those cases, we may resort to a refined probability function, called a Mixed Weibull (or Mixture Weibull) function and defined as:

$$f(x) = w_1 f_1(x) + w_2 f_2(x) + ... + w_n f_n(x) \tag{1}$$

with $w_i > 0$ and $\sum_{i=1}^{n} w_i = 1$. Each f_i is a Weibull distribution function and represents an independent population. Consequently, Eq. 1 models a system where failures come from different, independent families and it becomes an appropriate framework to represent failures of components in a supercomputer, given Observation #6 from Sect. 2. Mixed Weibull models are a better fit for the failure data of supercomputers. Figure 5 shows a comparison of a single Weibull function versus a Mixed Weibull function in fitting the failure data from Epoch 1 of all components of Titan. Figure 5a presents how well the two alternatives fit the failure data using the KST test. The Mixed Weibull models performs better than the single Weibull function, which can be seen in the probability plot of Fig. 5b. We propose using a Mixed Weibull distribution function to model the failures in a supercomputer:

$$f(x) = w_{GPU} f_{GPU}(x) + w_{CPU} f_{CPU}(x) + w_{MEM} f_{MEM}(x) \tag{2}$$

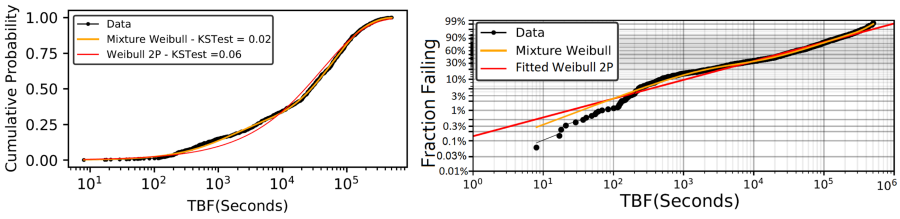
with $w_{GPU} + w_{CPU} + w_{MEM} = 1$. Values for w_i in Eq. 2 will depend on the actual proportion of failure of each component in the supercomputer. To validate our premises, Table 2 presents the results of an algorithm that automatically finds components of a Mixed Weibull distribution on a collection of data assuming there are 3 independent sources of failures. We can see how closely such results match the real proportions of the components.

Table 2. Mixed Weibull and 2P Weibull.

Distribution	Component	Shape	Scale	MTBF (Hours)	Estimate proportion	Real proportion
3 Mixture Weibull	1	0.871	1208	15.66	0.208	0.181
	2	0.934	59666		0.7544	0.791
	3	2.778	289484		0.0376	0.028
Weibull 2P	cpu, gpu, mem	0.62	40634.57	16.33		

3.2 Extreme-Scale Projections

We use Eq. 2 to estimate MTBF values of exascale systems and contrast those values with their corresponding power consumption. We developed different supercomputer configurations taking into account three components: processor, accelerator, and memory. The proportion of these components was varied to reach exascale performance.



(a) Mixture and 2P Weibull CDF. **(b)** Mixture and 2P Weibull probability plot.

Fig. 5. Epoch 1 statistical analysis. We chose Epoch 1 for modelling failures of Titan supercomputer, since Epoch 1 represents an expected failure behavior according to the system administrator. However, the same analysis could have been done using any other epoch.

Assumptions. The previous model analysis of this paper was made with failure data of Titan supercomputer (Epoch 1). The failure data was generated from an AMD 16-core Opteron CPU, an NVIDIA Tesla K20 GPU, and 32 GB of main memory. Nevertheless, to make realistic projections, we updated the CPU and GPU components to the actual time. We used the specifications of the AMD Epyc 7742 and the NVIDIA V100 that have 2.3 TFlops of performance, 225 W TDP and 7.8 TFlops of performance and 250 W TDP respectively. Regarding memory, we only multiply the amount of RAM by a factor depending on the projection.

To project the power consumption of an exascale machine, we considered the power draw of processors and accelerators. These components and the memory determine the maximum power required by a subsystem. Currently, the maximum power consumption of a supercomputer or HPC system is determined by

the sum of the power of its subsystems [2]. Nevertheless, we did not take into consideration the memory power in the projections for its relatively low power consumption. Also, cooling, network, and storage power consumption was left out of the power projections.

Size and Failure Data. All the projections were made based on failure data of Epoch 1 (from January 2014 to February 2016) produced by the CPU (32 failures), GPU(930 failures) and memory (213 failures) components. The failure data of Epoch 1 was the result of the execution of 18,688 nodes with a proportion of 1:1:1(1 GPU:1 CPU:32 GB RAM). For that reason, if we wanted to model a different component proportion we needed to generate synthetic data. The synthetic data was randomly generated for each component, but based on the Weibull shape and scale of the real data to ensure the same failure behavior. To determine the total number of failures (ϕ) required to perform the projections we used the following equation $\phi = \sum_{i=1}^{K}(\pi_i * \delta_i * s * \frac{1}{N})$, where K is the total number of hardware components, π_i is the proportion of component i, δ_i is the number of failures of component i and N is the total number of nodes on a real HPC system. Note that inside the equation the number of sockets s of the new projected system is calculated as $\frac{1\ exaflop}{\sum_{j=1}^{K}(\tau_j * \pi_j)}$, where τ_j is the number of teraflops of component j.

Reliability Measure. We used the MTBF as a metric to measure the system reliability. The system MTBF was calculated based on the mean Weibull and the proportion of the total failure data of each component population. The MTBF values were calculated as the mean of performing 100 times each experiment. Systems with low MTBF are less reliable.

Projections. We developed a series of exascale supercomputer projections varying the component proportions. Tables 3 and 4 show two experiments. One experiment only shows the change of the GPU proportion and the other shows the result of changing the proportion of the three components. We see in Table 3 that when increasing the number of GPUs the MTBF decreases. This is a normal behavior considering that we were increasing the component with the highest failure rate. Titan supercomputer in Epoch 1 had an MTBF of 42.3 h with the proportion 1:1:1 and the same proportion of an exascale machine has 9.98 h. Nevertheless, Titan only had 0.027 exaflops relative to the projected exascale machine. Regarding the power consumption, we see that the simplest proportion (1:1:1) has the highest power consumption because to reach exascale performance it is necessary to use 99,010 sockets. Also, note that with the increase of GPUs the power decreases. Comparing the projection (3:1:1) with real life, we can use Summit supercomputer that has the same component proportion (6 GPUs and 2 CPUs). At this time Summit is the fastest supercomputer in the world according to the Top500 list [25], and has a power consumption of 10 KW with 200 petaFLOPs of performance. We can assume that Summit could reach exascale performance with a size five times larger and this could increase the power consumption to 50 KW that is 24% more than the projected power consumption. Finally, it is important to remark that each of the proportions cor-

responds to real supercomputers: Titan (1:1:1), ABCI (2:1:1), Summit (3:1:1) and Lassen (4:1:1). All those supercomputers are listed on the records of the Top500 list [25].

Table 3. Projections changing the GPU proportion.

Proportion (gpu:cpu:mem)	Sockets	Data proportion (gpu:cpu:mem)	Total failures	MTBF (Hours)	CPU TDP (KW)	GPU TDP (KW)	System TDP (KW)
1:1:1	99010	5.3:5.3:5.3	6228	9.98	22277	24753	47030
2:1:1	55866	6:3:3	6315	9.52	12570	27933	40503
3:1:1	38911	6.3:2.1:2.1	6373	9.36	8755	29183	37938
4:1:1	29851	6.4:1.6:1.6	6344	8.97	6716	29851	36567

Table 4 shows other possible configurations. Although this study was focused on hybrid supercomputers, we also made one projection without GPU with the proportion 0:1:1. This proportion corresponds to Tianhe-2A supercomputer that has only Intel Xeon E5 CPUs. This projection needed 77% more sockets regarding the projection with one GPU (1:1:1), 52% more power consumption and the system reliability decrease with an MTBF of 8.06. This projection can give us an idea of how beneficial could be to implement supercomputers with at least one GPU per CPU. With projections 1:2:1 and 3:1:2 we see how the system can be with more CPUs and memory, respectively. Note that with the proportion with more CPUs the best obtained MTBF value was 12.06. The worst MTBF value was obtained with many GPUs (8:1:1). Such configuration also brings the best power consumption. Also, the memory size increase decreases the system reliability.

As a result of the projections we can conclude that hybrid supercomputers are a good solution to reach exascale performance. Hybrid supercomputers need less hardware and the power consumption is remarkably less than supercomputers without GPUs. Nevertheless, it is necessary to take into consideration that the system reliability could be affected by the increase in the GPU proportion.

Table 4. Projections changing multiple component proportions.

Proportion (gpu:cpu:mem)	Sockets	Data proportion (gpu:cpu:mem)	Total failures	MTBF (Hours)	CPU TDP (KW)	GPU TDP (KW)	System TDP (KW)
0:1:1	434783	0:23.2:23.2	5684	8.06	97826	0	97826
1:2:1	80646	4.3:8.6:4.3	5191	12.06	36291	20162	56452
3:1:2	38911	6.3:2.1:4.2	6821	8.7	8755	29183	37938
8:1:1	15456	6.64:0.83:0.83	6378	7.94	3478	30912	34390

4 Final Remarks

This paper presented the Mixed Weibull distribution function as a more appropriate model for failure characterization and prediction in hybrid exascale supercomputers. Starting from a collection of insights on a five-year failure record of

a leadership-class supercomputer, we built a whole-system failure model using Mixed Weibull distributions. These models allow the failure prediction of different configurations according to the prominence of a component in the machine: processor, accelerator, or memory. In the future, we plan on exploring two avenues of research. First, we will evaluate the Mixed Gamma distribution for modelling failures on supercomputers. Second, we will extend the power consumption model to include missing hardware components (storage, network, cooling, memory) and application characteristics during execution.

Acknowledgment. This research was partially supported by a machine allocation on Kabré supercomputer at the Costa Rica National High Technology Center. Early versions of this manuscript received valuable comments from Prof. Marcela Alfaro-Cordoba at University of Costa Rica.

References

1. Bautista-Gomez, L., Zyulkyarov, F., Unsal, O., McIntosh-Smith, S.: Unprotected computing: a large-scale study of dram raw error rate on a supercomputer. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2016, pp. 645–655, November 2016. https://doi.org/10.1109/SC.2016.54
2. Borghesi, A., Bartolini, A., Lombardi, M., Milano, M., Benini, L.: Predictive modeling for job power consumption in HPC systems. In: Kunkel, J.M., Balaji, P., Dongarra, J. (eds.) ISC High Performance 2016. LNCS, vol. 9697, pp. 181–199. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41321-1_10
3. Cappello, F., Al, G., Gropp, W., Kale, S., Kramer, B., Snir, M.: Toward exascale resilience: 2014 update. Supercomput. Front. Innov. Int. J. **1**(1), 5–28 (2014). https://doi.org/10.14529/jsfi140101
4. Di, S., Gupta, R., Snir, M., Pershey, E., Cappello, F.: Logaider: a tool for mining potential correlations of HPC log events. In: 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pp. 442–451, May 2017. https://doi.org/10.1109/CCGRID.2017.18
5. El-Sayed, N., Schroeder, B.: Reading between the lines of failure logs: understanding how HPC systems fail. In: 2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 1–12, June 2013. https://doi.org/10.1109/DSN.2013.6575356
6. Gainaru, A., Cappello, F., Kramer, W.: Taming of the shrew: modeling the normal and faulty behaviour of large-scale HPC systems. In: 2012 IEEE 26th International Parallel and Distributed Processing Symposium, pp. 1168–1179, May 2012. https://doi.org/10.1109/IPDPS.2012.107
7. Gallet, M., Yigitbasi, N., Javadi, B., Kondo, D., Iosup, A., Epema, D.: A model for space-correlated failures in large-scale distributed systems. In: D'Ambra, P., Guarracino, M., Talia, D. (eds.) Euro-Par 2010. LNCS, vol. 6271, pp. 88–100. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15277-1_10
8. Heien, E., LaPine, D., Kondo, D., Kramer, B., Gainaru, A., Cappello, F.: Modeling and tolerating heterogeneous failures in large parallel systems. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2011, pp. 1–11, November 2011. https://doi.org/10.1145/2063384.2063444

9. Levy, S., Ferreira, K.B.: An examination of the impact of failure distribution on coordinated checkpoint/restart. In: Proceedings of the ACM Workshop on Fault-Tolerance for HPC at Extreme Scale, FTXS 2016, pp. 35–42. ACM, New York (2016). https://doi.org/10.1145/2909428.2909430
10. Li, S., et al.: System implications of memory reliability in exascale computing. In: Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2011, pp. 46:1–46:12. ACM, New York (2011). https://doi.org/10.1145/2063384.2063445
11. Martino, C.D., Kalbarczyk, Z., Iyer, R.K., Baccanico, F., Fullop, J., Kramer, W.: Lessons learned from the analysis of system failures at petascale: the case of blue waters. In: 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 610–621, June 2014. https://doi.org/10.1109/DSN.2014.62
12. Martino, C.D., Kramer, W., Kalbarczyk, Z., Iyer, R.: Measuring and understanding extreme-scale application resilience: a field study of 5,000,000 HPC application runs. In: 2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 25–36, June 2015. https://doi.org/10.1109/DSN.2015.50
13. Meneses, E., Ni, X., Jones, T., Maxwell, D.: Analyzing the interplay of failures and workload on a leadership-class supercomputer. In: Cray User Group (CUG) Conference, May 2015
14. Minh, T.N., Pierre, G.: Failure analysis and modeling in large multi-site infrastructures. In: Dowling, J., Taïani, F. (eds.) DAIS 2013. LNCS, vol. 7891, pp. 127–140. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38541-4_10
15. Nie, B., Tiwari, D., Gupta, S., Smirni, E., Rogers, J.H.: A large-scale study of soft-errors on GPUs in the field. In: 2016 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp. 519–530. IEEE Xplore Digital Library, Barcelona, March 2016. https://doi.org/10.1109/HPCA.2016.7446091
16. Oliner, A., Stearley, J.: What supercomputers say: a study of five system logs. In: 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2007), pp. 575–584, June 2007. https://doi.org/10.1109/DSN.2007.103
17. Rojas, E., Meneses, E., Jones, T., Maxwell, D.: Analyzing a five-year failure record of a leadership-class supercomputer. In: International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD), October 2019. https://doi.org/10.1109/SBAC-PAD.2019.00040
18. Schroeder, B., Gibson, G.: A large-scale study of failures in high-performance computing systems. IEEE Trans. Dependable Secure Comput. **7**(4), 337–350 (2010). https://doi.org/10.1109/TDSC.2009.4
19. Schroeder, B., Gibson, G.A.: Understanding failures in petascale computers. J. Phys. Conf. Ser. **78** (2007). https://doi.org/10.1088/1742-6596/78/1/012022
20. Schroeder, B., Pinheiro, E., Weber, W.D.: Dram errors in the wild: a large-scale field study. In: Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems, SIGMETRICS 2009, pp. 193–204. ACM, New York (2009). https://doi.org/10.1145/1555349.1555372
21. Schulz, M., Lucas, B., Macaluso, T., Quinlan, D., Wu, J.: Inter-Agency Workshop on HPC Resilience at Extreme Scale National Security Agency Advanced Computing Systems, 21–24 February 2012 Coordinating Representatives John Daly (DOD) Bill Harrod (DOE/SC) Thuc Hoang (DOE/NNSA) (2012)
22. Snir, M., et al.: Addressing failures in exascale computing. Int. J. High Perform. Comput. Appl. **28**(2), 129–173 (2014). https://doi.org/10.1177/1094342014522573

23. Tiwari, D., Gupta, S., Gallarno, G., Rogers, J., Maxwell, D.: 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, pp. 1–12. IEEE Xplore Digital Library, Austin, November 2015. https://doi.org/10.1145/2807591.2807666
24. Tiwari, D., et al.: Understanding GPU errors on large-scale HPC systems and the implications for system design and operation. In: 2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA), pp. 331–342. IEEE Xplore Digital Library, Burlingame, February 2015. https://doi.org/10.1109/HPCA.2015.7056044
25. Top500.org: Top500 supercomputing sites (2018). https://www.top500.org/. Accessed 19 Aug 2018
26. Vaarandi, R.: Sec - simple event correlator (2018). https://simple-evcorr.github.io. Accessed 19 Aug 2018
27. Wu, M., Sun, X., Jin, H.: Performance under failures of high-end computing. In: Proceedings of the 2007 ACM/IEEE Conference on Supercomputing, SC 2007, pp. 1–11, November 2007. https://doi.org/10.1145/1362622.1362687
28. Yigitbasi, N., Gallet, M., Kondo, D., Iosup, A., Epema, D.: Analysis and modeling of time-correlated failures in large-scale distributed systems. In: 2010 11th IEEE/ACM International Conference on Grid Computing, pp. 65–72, October 2010. https://doi.org/10.1109/GRID.2010.5697961
29. Zheng, Z., et al.: Co-analysis of RAS log and job log on Blue Gene/P. In: 2011 IEEE International Parallel Distributed Processing Symposium, pp. 840–851, May 2011. https://doi.org/10.1109/IPDPS.2011.83