Abstract—Cloud datacenters must ensure high availability for the hosted applications and failures can be the bane of datacenter operators. Understanding the what, when and why of failures can help tremendously to mitigate their occurrence and impact. Failures can, however, depend on numerous spatial and temporal factors spanning hardware, workloads, support facilities, and even the environment. One has to rely on failure data from the field to quantify the influence of these factors on failures. Towards this goal, we collect failures data along with many parameters that might influence failures from two large production datacenters with very diverse characteristics. We show that multiple factors simultaneously affect failures, and these factors may interact in non-trivial ways. This makes conventional approaches that study aggregate characteristics or single parameter influences, rather inaccurate. Instead, we build a multi-factor analysis framework to systematically identify influencing factors, quantify their relative impact, and help in more accurate decision making for failure mitigation. We demonstrate this approach for three important decisions: spare capacity provisioning, comparing the reliability of hardware for vendor selection, and quantifying flexibility in datacenter climate control for cost-reliability trade-offs.

I. INTRODUCTION

Large-scale cloud datacenters (DC) are critical to many of our computing needs, both for hosting internet-scale applications such as web search and social networking that require multi-megawatt compute capacity, and for the increasingly attractive public clouds offering Infrastructure/Platform/Software-as-a-Service to consolidate many online small and enterprise applications. With thousands of IT components including compute, storage, and network devices, and other power and cooling equipment that supports the IT components, there is a high likelihood of failure [19, 34, 35]. Cloud downtime results in both lost revenue and user dissatisfaction for the hosted applications. Consequently, applications demand stringent availability Service Level Agreements (SLAs) from cloud DCs.

Given the high cost and user dissatisfaction associated with failures, DC operators spend considerable effort to reduce failures and/or mitigate their impact through techniques in hardware (e.g. hot spares, hardened devices), software (e.g. replication, migration, checkpoint-recovery,) and operations (e.g. pro-active maintenance, system rejuvenation). All these techniques have a non-trivial cost, and some may also degrade performance. For instance, Tier-4 DCs at 99.999% availability cost 2.3X as much per rack compared to a Tier-1 DC at 99.9% availability [16]. Reducing such costs and potential performance overheads in failure mitigation requires a critical investigation of what can go wrong, when it may go wrong, and why. Insights from such investigations have ramifications for DC design and long term planning as well as daily operations to help enhance utilization, performance, and availability while getting the maximum value out of the infrastructure investments. Section II identifies some of these important decisions that could benefit from such understanding.

Multiple factors – spatial (location of the DC, rack within the DC), temporal (time, season), hardware (vendor, server type, age), workload (computation/ storage/ communication intensities), environmental (temperature, humidity, altitude), and facility type (cooling system, packaging, power delivery and backup system) – influence failure characteristics. It is essential to consider these factors holistically when making decisions, rather than considering only the factor(s) one wishes to control or change, as the latter can lead to erroneous conclusions. For instance, as we will show, instead of basing spare provisioning using aggregate failure characteristics, considering different factors that correlate with server failures, can lead to significant savings. Similarly, studying correlations of only the decision variable (e.g. hardware type, temperature) to failures without including other factors can mis-estimate the influence of those variables on failures, leading to potentially erroneous decisions.

Since physical models for the effects of all such factors on various types of failures are not available, failure mitigation decisions must be driven by data from the field, and that data needs to encompass the multitude of factors listed above. Prior studies that have used real world failure data [15, 41, 42, 44, 49, 52] target only specific types of failures (e.g. memory, disk failures, etc.) and use the analysis for a limited set of correlations (e.g. studying the effect of temperature on disk failures) without considering broader decisions such as spare capacity planning. The data is often limited to a small subset of factors that affect failures (typically computer-systems centric), and does not capture extraneous issues (e.g. power density, external temperatures, humidity). There are a few studies [26, 43, 45, 47] that are more generic, studying at an aggregate level the failure events in large scale supercomputers and DCs. These focus on either aggregate failure rates or limited failure types, but do not develop a systematic understanding of how multiple factors jointly affect different types of failures at varying spatial and temporal granularities. These works are confined to a single DC and do not capture the effects of factors that differ across various DCs. Section III discusses related work in more detail.

Contributions: In this paper, we demonstrate that a holistic consideration of multiple factors that correlate with failures is important for cloud operators to make decisions related to cost-effective failure mitigation.

We present a detailed characterization of failures in production cloud DCs as a function of environment, datacenter proper-

©2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. DOI:
ties, workload properties, hardware types, components’ age and rack level power ratings. We use data from two cloud-scale production DCs for this investigation. Both these DCs host tens of thousands of servers running private and public cloud applications. The two DCs are situated at different geographic locations, and they differ in their external environment (weather, altitude), and internal environment (cooling, packaging, and power backup). The data we use was collected over 2.5 years across multiple parameters encompassing hardware, workload, environment, and facilities, at fine temporal and spatial resolutions.

Analyzing cloud-scale DC operations data is a “big data” problem - where should we start? how do we study the relationship among factors? how do we isolate which is important? We build a framework to systematically conduct such analysis on this data to aid failure related decision-making. We show evidence that multiple factors correlate with failure rates, and that their combinations can create more complex interactions. As a result, our framework concurrently considers multiple factors (e.g., environment, workload categories, server types, DC geographic location and cooling scheme) that affect failures using Classification and Regression Trees (CART) and Partial Dependence Analysis. The analysis helps us to identify parameter relevant to failures and their interactions, at the server level as well as in disks and memory.

Based on the above analysis, we glean additional insights into the data, and how they can guide decisions for optimizing the DC’s total cost of ownership (TCO). In particular we apply the framework to study: (i) provisioning of spares, (ii) comparing the reliability of server hardware across vendors, and (iii) quantifying the range of environmental parameters in the DC under study for cost-reliability trade-offs. Some of the results include:

- We show that spare provisioning cost can be reduced by more than half compared to aggregate characteristics based provisioning, by grouping racks based on factors influencing their failures and differentially provisioning for each group.
- We show that provisioning spares at component-level (memory/hard drive) as opposed to entire server-level can further reduce the over-provisioning cost by over 40%.
- We show that the cost of handling failures (both spare provisioning and maintenance/repair) can vary from one hardware configuration to another by as much as 4x. We find that existing approaches grossly underestimate or overestimate the variance across hardware configurations due to their inability to isolate failures resulting from other influencing factors.
- We find that temperature/humidity set points may need to be customized for each DC and failure type. For a certain DC/failure-type combination studied in this paper, our model identifies that the combination of temperatures above 78°F and relative humidity below 25% is detrimental to reliability – 50% increase in failure rate of hard disks.

II. MOTIVATION AND GOALS

DCs hosting both private (catering to in-house applications) and public (hosting third-party applications) clouds, must deal with a wide range of failures. Failures affect hardware - computing equipment (servers with their memory and storage units), power and cooling infrastructure, as well as the configuration, management, virtualization, monitoring, networking and other hosting support software [19, 35]. To achieve high availability despite failures, DCs employ expensive failure mitigation techniques, ranging from software solutions (such as pro-active monitoring, application isolation, and load balancing), to redundant hardware components (in servers and support facilities), and even geo-replication across DCs to handle natural calamities [35]. We wish to aid DC designers and operators to manage and mitigate failures in a better and cost-effective way.

In this paper, we specifically focus on failures related to the compute infrastructure (i.e., servers, storage and network hardware), to help make critical decisions for the cloud infrastructure. There are several important decisions in planning, building, and operating the infrastructure (either one time, or on a periodic/ongoing basis at different time scales) to realize the application SLAs, both for performance and availability, at the lowest possible TCO:

- **Capital Expenditure (CapEx) related decisions**: There are numerous procurement and provisioning decisions such as: How many spares are required to achieve the availability SLA offered to applications in the overall DC and within each server rack or at other relevant granularities? Is it better to refresh or keep spares at the entire server granularity, or at the granularity of individual components (e.g. DRAM, disk, etc.)? Are servers with comparable performance specifications (e.g., CPU speed, memory size) from different vendors also commensurate from the availability perspective, and based on that, which vendor’s product would be more effective to procure given their costs? Should one opt for more reliable (more expensive) servers and components or more spares? Should spares be maintained for each class of applications separately, or is it better to have a shared pool?

- **Operational Expenditure (OpEx) related decisions**: There are also several operational decisions for better managing cost-reliability trade-offs in DCs, such as: Given environment (temperature, humidity, etc.) control for ideal operation can be expensive, how much leeway do we have to stay from the recommended settings to save costs without significantly affecting reliability? Is it better to replace a server/component, as opposed to servicing it? Can proactive (load migration [10], software rejuvenation [20], etc.) and/or reactive (checkpoint and restart, fail-overs [7], etc.) techniques to failure mitigation be employed to defer equipment refresh while still adhering to application SLAs? Which vendor’s product has lower repair costs? Can root causes of failures be identified, and perhaps even predicted, so as to efficiently plan for repair/service?

An understanding and analysis of failures from the field is invaluable to a whole range of planning and operational issues for the infrastructure provider. An exhaustive treatment of all the above questions is well beyond the scope of this paper. Instead, we specifically focus on three of the important questions and provide a systematic approach to answering them using field data from two cloud-scale DCs:

- Q1. How many spare servers should be maintained to meet a workload’s availability mandate? Is it better to keep server spares or component (DRAM, disk) spares? As far
as possible, we would like to maintain these spares for each rack of the workload, to handle the failures within that rack, given the communication penalties when relocating applications or virtual machines to other racks [51].

- **Q2.** Are some vendor products (denoted SKU for Stock Keeping Units) more reliable than others? How do we use this information in procurement decisions?
- **Q3.** How much leeway, and what range of operation can we allow for environmental control knobs like temperature, humidity, pressure and air-flow, without significantly compromising reliability?

To our knowledge, this is the first study to answer the first two questions stated above using large-scale production data. For the third question, while the impact of temperature on (disk) reliability has been studied before [13, 23, 44], one of the new insights from our investigation is that studying the impact of a single parameter (e.g. temperature or SKUs) individually on failures does not suffice because there is a plethora of factors that interact in complicated ways. As we will show, multi-factor interactions make single parameter studies less accurate, resulting in inefficiencies including over/under-provisioning of equipment and mis-estimation of acceptable temperature ranges. Instead, a coordinated multi-parameter analysis of interactions is needed to provide answers for such questions, and we explain this approach in Section V.

### III. Related Work

**Datacenter failure analysis:** Prior literature has analyzed failures of large scale computing systems in production environment from DCs [5, 11, 12, 14, 19, 26, 30, 37, 43, 45, 47, 50, 54, 55] to component hardware such as disk/flash failures [9, 29, 33, 41, 44, 46, 48], memory failures [49, 52] and network failures [15, 22, 42]. Prior studies have analyzed the impact of environment such as temperature [13, 23, 41, 52], cooling [28], and physical location of DCs (altitude) [52].

In this work, we analyze hardware failures in datacenters as they directly affect hardware availability incurring significant repair/replacement costs. Our analysis aims to answer new failure related questions, affecting datacenter provisioning and management decisions, that are not considered in prior works. Another key difference from above works is that we consider the interplay of multiple failure factors allowing us to more effectively make important decisions.

**Approaches to failure analysis:** Several statistical and machine learning techniques [31] like classification and regression trees [25], Bayesian [17], Nearest Neighbor [26, 39] and predictive modeling [32] have been used for failure predictions for an application, individual server components, supercomputers, and server clusters. We draw our data analysis techniques from a similar repertoire of statistical and machine learning methods, and end up using techniques based on our requirements, such as addressing non-linearities, multiple types of data variables including numerical and categorical, and jointly accounting for multiple factors.

### IV. Data Description

In this paper, we adopt a data driven approach from real facilities to show how a DC operator can use such information to answer the questions raised earlier.

We have collected data from two production cloud DCs (denoted as DC1 and DC2), each containing thousands of servers, together with the power distribution and environmental control equipment supporting these facilities. The data spans a period of more than 2.5 years. This data covers both static (invariant over time) information, as well as dynamic (temporal) information, that is sampled at fine time scales and at different spatial (from individual server, to rack and DC scales) resolutions, including external ambient conditions. The extent and diversity of the collected data provides sufficient statistical significance to the results, together with supporting our rationale for the need for multi-parameter/factor based analysis of such data for decision making. Table III summarizes important parameters relevant to this study and their value range.

**DC Environment:** The two DCs differ in their geographic location, cooling, packaging and availability design (see Table I). DC1 uses adiabatic cooling where-as DC2 uses a more traditional Heating, Ventilation, and air-conditioning (HVAC) based cooling system1. DC1 uses containers [19] for packaging the infrastructure where-as DC2 uses a co-location [1]. Similarly, DC1’s power infrastructure is designed to guarantee three nines of availability where-as DC2 is designed for higher five nines of availability. Both serve in-house and third-party applications, and cover different classes of workloads. Servers in large DCs are typically organized in a spatial hierarchy, from a DC at the top, each having rows of racks which in turn house server chassis. The servers span several different configurations, or SKUs. SKUs that are compute intensive have more servers per rack (>40) and fewer hard disk drives (HDD) per server (≈20). In contrast, storage SKUs have less servers per rack (≈20) and more HDD per server.

<table>
<thead>
<tr>
<th>Facility</th>
<th>Packaging Design</th>
<th>Availability</th>
<th>Cooling</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC1</td>
<td>Container</td>
<td>3 nines</td>
<td>Adiabatic</td>
</tr>
<tr>
<td>DC2</td>
<td>Colocated</td>
<td>5 nines</td>
<td>Chilled water</td>
</tr>
</tbody>
</table>

1 Adiabatic cooling [2] leverages evaporation mode of cooling. This approach compresses gas into a liquid form with a mechanical pump, expelling the heat created, and then lets the liquid expand again, cooling a controlled mass of air and absorbing the heat around. Adiabatic cooling yields high energy efficiency and proves effective in warm, dry climates, but has a major drawback of the need for a large amount of water usage. HVAC cooling systems on the other hand use a refrigerant and an evaporator. Heat dissipated from the IT environment is pumped to the outdoor environment using this circulating flow of refrigerant. This mode of cooling has the advantage of lower capital expenditure (CapEx) to maintain, but incurs large operational power cost.
be monitored for different kinds of information, e.g. separate sensors for measuring inlet and outlet air temperature of a rack. A building management system (BMS) is responsible for the collection and monitoring of the sensor data, and triggering specific actions like alarms, when any of the sensor values exceed the normal threshold range.

**Failure Tickets:** A common reporting mechanism, called RMA (Return Merchandise Authorization) tickets, is used in industry for detection and identification of hardware and software failures, and towards assessing SLA guarantees. We use these RMAs for tracking and quantifying different failures in the DCs. Typically a failure ticket is initiated with the onset of a failure as detected by a DC management framework such as Autopilot [21]. An operating engineer investigates the root cause of this RMA ticket, and if it is a hardware fault, the ticket is resolved by replacing the faulty component. The ticket could also be a false positive, meaning no specific error is identified. We use only the true positives in our analysis. A failure ticket has a *description* field that provides a brief description of the nature of the fault, in addition to the time of occurrence, resolution, repeat count and other relevant comments.

**Fault types:** The *fault description* field of the RMA tickets can be in one of four main categories:

- **Hardware failure:** These include faults in physical components like disk, memory, processor, power strip, network device etc., and are resolved using repair or replacement. We consider only these physical hardware failures to answer the three questions posed in Section II.
- **Software failure:** These capture failures related to OS, applications, and services. The resolution involves software fixes like re-imaging, re-installation, etc.
- **Boot failure:** Failed reboots or PXE boot failures result in boot failure tickets.
- **Others:** These include tickets that do not fall in the above categories due to lack of additional information.

Table II presents the break-up of RMA tickets for the two datacenters DC1 and DC2 respectively.

<table>
<thead>
<tr>
<th>Category</th>
<th>Failure Type</th>
<th>DC1%</th>
<th>DC2%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software</td>
<td>Timeout failure</td>
<td>31.27</td>
<td>38.84</td>
</tr>
<tr>
<td></td>
<td>Deployment failure</td>
<td>13.95</td>
<td>14.56</td>
</tr>
<tr>
<td></td>
<td>Node/Agent crash</td>
<td>2.89</td>
<td>3.05</td>
</tr>
<tr>
<td>Boot</td>
<td>PXE boot failure</td>
<td>10.53</td>
<td>13.81</td>
</tr>
<tr>
<td></td>
<td>Reboot failure</td>
<td>1.25</td>
<td>0.19</td>
</tr>
<tr>
<td>Hardware</td>
<td>Disk failure</td>
<td>18.42</td>
<td>11.23</td>
</tr>
<tr>
<td></td>
<td>Memory failure</td>
<td>5.29</td>
<td>1.85</td>
</tr>
<tr>
<td></td>
<td>Power failure</td>
<td>1.59</td>
<td>3.83</td>
</tr>
<tr>
<td></td>
<td>Server failure</td>
<td>2.84</td>
<td>1.21</td>
</tr>
<tr>
<td></td>
<td>Network failure</td>
<td>2.52</td>
<td>0.65</td>
</tr>
<tr>
<td>Others</td>
<td>Others</td>
<td>9.41</td>
<td>10.77</td>
</tr>
</tbody>
</table>

**TABLE II:** Classification of failure tickets.

Software failures are the most common cause in both DCs, similar to the systems studied in [5, 19], contributing to 45-55% of the total failures. Software timeout is the leading cause resulting in 30-40% of the total failures, followed by deployment related failures. Crashes account for only 3% of the total failures, which also concurs with [5].

Boot failures account for 12-14% of the total failures in both DCs, with PXE boot failure being the dominant cause.

Hardware failures account for about 20-30% of the total failures. Disks are the leading cause for RMA generation in this category followed by memory and power related failures. This is similar to the hardware replacement trends observed in [46]. Other causes account for around 10% of the failures.

In this work, we specifically focus on hardware failures to answer the questions posed in section II, as they result in substantial server downtimes, and also incur significant repair/replacement costs in our environments.

**V. METHODOLOGY**

In this section, we describe general methodologies that can be leveraged to answer the questions discussed in Section II. While the specific analysis techniques themselves have been adopted from well-studied rigorous statistical approaches, the key point to note is that answering such questions requires a holistic consideration of the numerous parameters (both static and dynamic, that may be hardware, software and facilities/environment related), rather than simple correlation studies based on a small subset of parameters as have often been used in prior research to draw conclusions [35, 49, 52, 54]. Otherwise, gross errors can result in poor decisions.

**Categories of Questions:** Before discussing approaches to answer the questions, note that most of the questions identified in Section II (including the specific three studied in this paper) fall into one of two categories:

- **Cat. 1:** Characterizing and studying aggregate behavior (of failures) when employing mitigation techniques: For instance, taking Q1, understanding overall failure behavior (how often? how correlated? etc.) could be used to figure out how many spares to provide for fail-overs. Similarly, aggregate behaviors on predictability (say patterns in error occurrences) could also be used to optimize dynamic mitigation techniques such as replication and migration [10, 40].
- **Cat. 2:** Characterizing and quantifying influences of specific decision variables on failures: For instance, Q2 requires a study of the impact of the SKU on failure rates. Similarly, Q3 involves studying the impact of temperature on failures.

**Metrics:** When answering these questions, there are two main metrics\(^2\) that we will consider to study the effect of failures on procurement and operational decisions:

- **RMA/Failure generation rate (λ):** As explained earlier, we use RMAs to detect failures, and its generation rate can be tracked at different spatial (from DC to Racks, Servers and even components of a server such as memory and disk) and temporal granularities (minutes, hours, days, weeks, months). We can also separately collate it by the workloads hosted on the servers where the failures occurred.

- **Number of devices with failures over a duration (μ):** This metric tracks number of devices that are concurrently unavailable due to failure. This metric could be computed at different spatial and temporal resolutions. This is distinct from the average failure rate, since this also captures the duration of the failure and indicates how correlated the failures are. Correlations become important in many decisions, including server provisioning, e.g. one spare may suffice when two

\(^2\)In all presented graphs, results for these metrics are normalized with respect to their maximum value.
servers do not fail at the same time but more may be needed to handle simultaneous (temporally correlated) failures.

We will next show that approaches which may not consider the interaction of multiple factors, may fall short when answering questions in either categories.

A. Insufficiency of single factor approaches

We define a single factor (SF) approach to failure analysis as one which uses only the characteristics of failure metrics and their relationship with a decision variable, without considering the numerous factors that impact failure occurrences.

Many prior works [5, 26, 41, 43, 46, 49, 54], which have studied such aggregate behaviors of failures or the relationship of certain control parameters such as capacity, usage, age, temperature, etc. on failures fall under this approach. While this approach has its value for certain tasks, we discuss below some pitfalls in decision making when using it. We then present results showing that multiple factors are related to failure occurrences in our data set, giving evidence of the insufficiency of single factor based approaches for answering the type of questions we are interested in.

Consider a Cat. 1 question of provisioning spares based on $\mu$. One could consider a single factor approach that simply takes an aggregate distribution of $\mu$ over time based on a single factor (say workload) across the DC without regard to the SKU, DC/rack, temperature/humidity/air-flow variations in space or time, etc. A (CDF) distribution of $\mu$ across different racks for a given workload, might look like the solid black CDF curve in Fig. 1. Provisioning spares for the 95-th percentile of this distribution might be quite far from the mean of $\mu$. The x-axis point $e$ denotes the fraction of spare servers required overall. On the other hand, it is possible that a certain set of servers in a specific part of the DC running on certain SKU is experiencing a higher $\mu$ over time, while the bulk of other servers are in a lower $\mu$ region. The gray CDF curves show the CDFs of the low and high $\mu$ groups. While the means for these groups are different, the variance around their individual means is lower. These groups require $g_1$ and $g_2$ fraction of spare servers, respectively within each group. Using the spare fraction $g_1$ for bulk of the server population and a spare fraction of $g_2$ for the small set will result in a lower overall failures for the three cases. Histograph the number of failures for each SKU and use that to base vendor selection for the DC. It is possible that such an approach may suggest that $SKU_x$ is considerably more reliable than $SKU_y$, making a compelling reason to procure servers from the first vendor. However, the reason that lower failures were observed for $SKU_x$ than for $SKU_y$ may not be due to more resilient hardware used in $SKU_x$ but because of other factors such as the application hosted on those servers had smoother disk access patterns, or that most of those servers were placed in a cooler portion of the DC. The effect of influences other than the decision variable must be eliminated or minimized before any meaningful conclusions can be drawn.

B. Evidence of Multi-factor Influence

To further motivate the insufficiency of approaches that uses only the failure data or captures only the effect of a decision variable, we show that failures can be influenced by multiple factors. In the interest of space, only a subset of the factors are shown. In this section, we present the failure generation rate ($\lambda$) at the spatial granularity of rack and temporal granularity of a day, i.e. the number of failures per rack per day, and refer to them as failure rate.

Spatial Features: Fig. 2 shows the mean (and sd) of failure rate in different regions of the two DCs. As can be seen, failure rates vary considerably across and within different DCs (due to possibly different cooling technologies, design, and other environmental factors apart from any other differences within the DCs). In general, regions of DC1 shows higher failure rate than DC2. We observe similar variations within each DC region when grouped according to rows of racks and individual racks (not shown here).

Temporal Features: Fig. 3 shows the relationship between the failure rates and the day-of-week effect. In general, mean failure rate is high on weekdays. This effect is expected as a result of variations in workload demand over the week. Fig. 4 shows the failure rates for various months of a year. We see an increase in failures (both mean and sd) in the second half of the year which may be due to external variations (like seasonal change induced environmental effects).

Environmental Features: Fig. 5 plots the relationship between average relative humidity (RH) on the day of failure and the failure rates. As can be seen, there is a notable variation in failure rates for lower humidity operating points. Temperature influence is not shown here, as it is extensively studied later in section VI.

Workload Features: Fig. 6 shows failure rates for workloads
factor analysis, and below we give details on the specific ones statistical and data mining models can be used for such multi-factors in our data set, grouped into different categories. Many categories. Table III summarizes some of the failure related multi-factor approach that jointly considers all relevant factors Hence rather than simply selecting the most important factor, a in the above results), but they can also interact with each other. It is not just that multiple factors influence failures (as observed of factors is likely to ignore the influence of many other factors The bottom line is that considering any one or a small number of factors is likely to ignore the influence of many other factors on failures, some of which may in fact be over-riding the effects of the considered factors.

C. Multi-Factor (MF) Approach

It is not just that multiple factors influence failures (as observed in the above results), but they can also interact with each other. Hence rather than simply selecting the most important factor, a multi-factor approach that jointly considers all relevant factors is needed when trying to answer questions from the two categories. Table III summarizes some of the failure related factors in our data set, grouped into different categories. Many statistical and data mining models can be used for such multi-factor analysis, and below we give details on the specific ones that we use for answering questions from the two categories.

**Hardware Features:** Fig. 7 shows marked differences in mean and sd of failure rates for different hardware SKUs. Similarly, Fig. 8 shows that racks with higher power ratings (> 12KW), on an average, report higher failure rates. We also plot the failure rate of the equipment as a function of its age (Fig. 9). Even though we do not see the tail of the well-known bath-tub curve in this data, we do notice that new equipment tends to have higher failures.

The bottom line is that considering any one or a small number of factors is likely to ignore the influence of many other factors on failures, some of which may in fact be over-riding the effects of the considered factors.

**MF for Cat. 1 Questions:** For such questions, we use Classification and Regression Trees (CART) [8] because this technique is non-parametric and captures non-linearities. It can model both numeric and categorical data seen in our dataset. It also naturally maps to our requirement to split a population (say servers) into groups with similar failure properties.

A CART tree is formed by a collection of rules that best split the data set into different observations based on the estimated variable. “Best” is characterized using metrics such as Gini Impurity that measures the homogeneity of the estimated variable within the split data. The splitting process is recursive and performed in a top-down manner (from root node to child nodes in the CART tree) and stops when no further gain can be made or pre-set stopping rules are met. The parameters can then be ranked based on the goodness of the splits. We use the rpart package [3] in R to build the CART models.

Finally, a grouping of the population will be reached that is desirable to answer the posed question. For instance, if we are interested in the metric of rack failure rates, the CART tree would consider the different features that would best describe the resulting failure rates for a group of racks, creating branches accordingly and dynamically figuring out both the number of groups as well as the racks within each group. We

---

Redundant/correlated factors: If redundant/correlated factors are present, CART still uses the best split to identify the tree [8]. However, the redundant/correlated factors are also included in computing the relative importance of factors.
use the notation Metric ∼ X₁, X₂, ..., Xₙ, to denote a call to this procedure, where Xᵢs are features, and Metric (e.g. λ, µ) is the decision variable for tree construction.

In this work, we perform descriptive analytics and CART works well for our dataset for the above reasons. However, if the one needs to predict failures, i.e. a binary value for Metric, CART by itself will not be sufficient for our dataset. As failed devices are a minority when compared to non-failed devices over the entire observation period, one may need preprocessing to balance these two sets (e.g. [6, 25]). An exhaustive treatment of the pros and cons of different such techniques is beyond the scope of this paper.

MF for Cat. 2 Questions: In the second class (e.g. Q2 and Q3), our goal is to quantify the dependence of failures on specific decision or control variables such as which vendor to buy from, or what is the margin on temperature and humidity that we can exploit to lower operating costs. As explained earlier, we should try to remove the impact of other factors when studying the influence of these specific issues.

We use techniques similar to partial dependence plot [18] and normalize the effect of all observed parameters (Table III) other than the parameter of interest We use the notation Metric ∼ X₁, N(X₂), ..., N(Xₙ), to denote this procedure call, where Xᵢs are features, and Metric (e.g. λ or µ) is the decision variable for which the influence of X₁ is being studied. This represents a path from the root to a leaf in the tree where X₁ is the leaf node and N(X₂), ..., N(Xₙ) represents the fixed values of other factors observed at this node. We quantify the confidence in the model, by checking if after normalization, the influence of this parameter is significant.

We use partial dependence based approach to quantify the dependence on the parameter of interest, accounting for observed parameters. It provides preliminary insights on the questions we are interested in. However, it neither accounts for unobserved factors nor represents true causality. Our attempts at using the state-of-the-art causal estimation from observational data (Pearl’s do-calculus [36]), proved to be a mismatch for our dataset as it cannot handle non-linear dependencies and nominal data types (i.e. categorical without any implicit ordering such as SKU and DC). We leave such an effort as a part of our future work.

VI. Evaluation

We now answer the three questions posed in Section II using the multiple factor (MF) model and compare the results with that of the single factor (SF) models.

Q1: Provisioning spares to meet availability targets

To determine the spare capacity in the presence of failures, we answer the two sub-questions of Q1 from Section II: Q1-A quantifies the number of spare servers and Q1-B quantifies the degree of redundancy in individual components to meet a workload’s availability requirements.

We define the availability SLA for a workload as the percentage of servers that needs to be available to that workload at all times, using 100%, 95% and 90% as example desired availability fractions. We assume application software is designed to handle failures, implying that the workload can fail-over to the spare resources by leveraging well known load redirection [40] and workload migration techniques [10]. The results show the improved over-provisioning capacity estimates on using the MF model compared to the SF models. Additionally, the results also illustrate the rich insights that MF provides in determining the key features that affect over-provisioning.

Q1-A: How many spare servers should be maintained to meet a workload’s availability mandate?: To answer this, we determine the spare capacity in terms of the number of additional servers per rack required for different workloads to meet their target availability. We find rack to be the natural granularity for provisioning spares for two main reasons, (i) Rack is typically the smallest granularity of server provisioning for applications hosted in large cloud datacenters [38]. (ii) From a performance perspective, applications or virtual machines are often affinity to the same rack to minimize network communication penalties [51]. The over-provision requirement for each rack is dependent on µ, i.e. the count of devices with concurrent failure over a period. We compare three different approaches to determine the spare capacity:

(a) Lower Bound (LB): This approach uses the raw failure rate distribution to compute the number of server spares, separately for each rack, to meet certain availability SLA. Of course, the failure characteristics cannot be measured before a rack is deployed because the impact of failure factors such as environmental or workload parameters is not available before deployment and measurement of those characteristics. So we treat this approach as a theoretical lower bound. For the evaluation, we pretend that the entire failure data measured for each rack was magically available before deployment and compute the spare capacity using this future data as a point of comparison. No practical approach would be able to perform better than this.

(b) Single Factor approach (SF): Given that the spare provisioning question only differentiates between availabilities for workloads, we study the relationship between µ and the workload category. This approach represents many prior works that provision spares based on the CDF of an estimated or measured failure rate for all the equipment of a given factor type (e.g. container facility in [53]). Here, this approach provides a uniform fraction of spares in each rack of given workload type using a CDF of µ for each workload type, since it cannot differentiate the racks based on other factors.

(c) Multiple Factor approach (MF): Rather than treat all
racks of a workload the same, we use the multi-factor model (using CART) described in Section V to identify the different influential feature clusters with unique provisioning needs, with the number of spares possibly varying across the clusters. This approach helps provision redundancy in new racks based on which cluster they fall into, where the clusters are determined based on previously measured data.

We evaluate the efficacy of these approaches by comparing the fraction of servers over-provisioned to meet the availability mandate of the workload, and quantify the savings using TCO analysis. We present the results for two workloads with diverse failure characteristics, namely W1, which is compute intensive, and W6, which is storage intensive.

Fig. 10 shows that the over-provisioned capacities determined by MF are much lower than those computed using the SF approach (less than half the over-provisioned capacity for the SLA of 100% availability) and in fact, very close to the lower bound capacity for both the compute and storage workloads across the three availability mandates. This is because MF is able to identify multiple rack clusters with unique and diverse over-provisioning needs –10 clusters with over-provisioned capacity ranging from 2% to 50% for the compute workload and 5 clusters with over-provisioned capacity ranging from 2% to 85% for the storage workload as shown in Fig.s 11a and 11b. This is in sharp contrast to the single factor approach which cannot extract the variations across multiple features, resulting in a conservative one-size-fits-all provisioning across all racks of the workload.

The spare capacity estimated by the SF for the compute workload is nearly half that of the storage workload (especially at stringent availability SLAs) as shown in Fig. 10. In contrast, the spare capacity estimated by the MF approach for the storage workload is almost the same and is in fact slightly lower than that of the compute workload. The reason for this disparity can be directly inferred from Fig. 11, which shows that the variance between compute workload MF clusters is much smaller than the variance across the storage workload MF clusters (2-50% for compute vs. 2-85% for storage). It is evident that the MF approach is able to reduce the spare capacity for storage workload, by providing differential spares across individual clusters.

Also, we find that temporal multiplexing of failure events within a rack further reduces over-provisioning, i.e. failures that are non-overlapping in time could potentially be handled by the same spare. We illustrate this by comparing the spare capacity determined for a coarser daily granularity in Fig. 10 with a finer hourly granularity in Fig. 12. The figures show

![Fig. 10: Over-provisioning requirement determined by the 3 approaches across 3 availability SLAs (daily granularity).](image)

![Fig. 11: Over-provisioning requirement determined by MF.](image)

![Fig. 12: Over-provisioning requirement determined by the 3 approaches across 3 availability SLAs (hourly granularity).](image)

<table>
<thead>
<tr>
<th>SLA</th>
<th>Daily-W1</th>
<th>Daily-W6</th>
<th>Hourly-W1</th>
<th>Hourly-W6</th>
</tr>
</thead>
<tbody>
<tr>
<td>90%</td>
<td>0.52%</td>
<td>3.77%</td>
<td>5.00%</td>
<td>2.70%</td>
</tr>
<tr>
<td>95%</td>
<td>2.60%</td>
<td>11.23%</td>
<td>7.23%</td>
<td>8.60%</td>
</tr>
<tr>
<td>100%</td>
<td>14.60%</td>
<td>35.66%</td>
<td>22.33%</td>
<td>36.37%</td>
</tr>
</tbody>
</table>

TABLE IV: Relative savings in TCO by using MF over SF.

that the spare capacity determined by MF, drops by almost half (20% to 10%) as we move from daily to hourly provisioning while that of the single factor remains the same.

We estimate the consequent impact of over-provisioning on TCO (using [24]). The relative savings in TCO by using MF over the SF are presented in Table IV. Savings as high as 14-36% are estimated for 100% availability SLA.

More importantly, the CART trees of MF approach also gives additional insights to understand how different clusters are formed. In the interest of space, we only summarize the main findings rather than give detailed data or investigate the root-cause behavior. We find that age, power rating and SKU type are the key factors in the formation of the storage workload clusters. In particular, we find that devices that are either very old or very young require more spares (the well-known bathtub behavior [46]). On the other hand, the compute workload has a different relative order of importance where spatial features like the datacenter have a higher dominance than SKU and power rating. Specifically, the top level branches are the datacenters, and the variations within each datacenter are captured based on Age, SKU, and power rating.

**Q1-B: Component level over-provisioning:** Rather than keeping spares at the server level, it can sometimes be more cost-effective to keep spares for the individual components that fail within the server. Component over-provisioning is cost-effective when done at aggregate scale [27], say at the rack-level, as opposed to doing it at the server level, primarily due to the multiplexing effects (i.e. any server in the rack could use the component) and discrete provisioning granularity (i.e. we may be able to add components like memory as only 16GB DIMMs rather than 1KB at a time). Emerging cost-based arguments [27] call for dis-aggregated components across servers, and novel rack architectures are being proposed to facilitate this.

In this section, we consider failures in two key server components, hard disk and memory, that significantly impact server reliability [19, 34], and determine the required spares in these components for satisfying a desired availability SLA. Hardware failures other than hard disk and memory are handled using server spares. To illustrate the best case provisioning improvement from using component spares, we consider over-
provisioning at an aggregate rack-level.

We compare the merits of spare provisioning between the component approach vs. the server level approach by obtaining relative cost of a single unit of these three spare resources from a commercial server-cost estimation tool [4] —Ratio of cost of server, hard-disk and memory is 100:2:10. We assume a 16 GB memory DIMM and a 1 TB HDD as the granularity of spare provisioning for the cost calculation. We determine the overprovisioning capacity for these three resources using the three approaches discussed in Section VI, namely (a) Lower Bound (LB), (b) Single Factor (SF), and, (c) Multi Factor (MF). In the interest of space, we present the over-provisioning capacity results for only the 100% availability mandate at the daily granularity in Fig. 13.

Fig. 13 indicates a clear benefit in provisioning spares at component level as opposed to doing it at just the server level with MF. Interestingly, the result seems counter-intuitive for the Single Factor (SF), especially for W1, with the over-provisioned capacity at the component-level exceeding that determined at the server-level. As the Single Factor (SF) approach separately provisions spares for each of these resources based on their corresponding peak failure rate, it results in a conservative sum of peak provisioning across these resources. But, in the MF approach, the cost of provisioning component-level spares for 100% availability SLA is lower than server-level provisioning computed for lower availability SLAs. Also, the compute workload has a more pronounced reduction of 40% where-as storage has only 10% reduction. Component level provisioning will result in TCO savings of 4.3% and 1.1% for compute and storage workloads respectively, when compared to server level provisioning using the MF.

Similar to the observations about server-level spares, the clusters for component-level spares for compute workload W1 are primarily influenced by spatial features such as the datacenter. For the storage workloads, hardware features such as power rating, age and SKU are ranked higher than other features.

Q2: Are some SKUs more reliable than others?

We answer this question (Q2 in Section II) by ranking different configurations of server vendors based on their reliability. We use rack SKU as a proxy for a specific combination of server models and vendors. SKU reliability impacts DC cost in two aspects (i) CapEx is impacted by the number of spares that needs to be provisioned to handle failure of servers belonging to a SKU, and (ii) OpEx is affected by the average frequency of maintenance/repairs associated with server failures belonging to a SKU. We use two metrics to capture the impact on CapEx and OpEx: (i) peak failure rate ($\mu_{\text{max}}$) of a SKU to determine spare capacity, and (ii) average failure rate ($\lambda$) of a SKU to determine service frequency. The failure metrics were computed for spatial granularity of a rack and temporal granularity of a day. We compare single factor (SF) approach that only considers the impact of decision variable (i.e. SKU) on failures with a multi-factor (MF) approach.

Fig. 14 shows the peak and average failure rate for 4 representative SKUs using the single factor model, normalized to their respective maximum. These 4 SKUs, denoted S1 through S4, fall into two categories: S1 and S3 are storage SKUs, while S2 and S4 are compute SKUs. Across these 4 SKUs, we find several contrasting failure characteristics and their implications on capacity and maintenance: (i) SKU S2 with highest average failure rate –10X of SKU S4’s– suggests higher service rates. (ii) SKU S3, with highest peak failure rates –1.4X of SKU S4’s– suggests higher spare capacity and hence higher CapEx. (iii) SKU S4 with lowest peak and average failure rates suggest lowest spares and service requirements. In summary, the single factor model indicates SKU S4 as the most reliable and S2 as the least reliable with 1.18X higher peak failure rate and 10X higher average failure rate.

We next show that the single factor model is limited in that it does not capture the partial dependence of failure rate on server SKUs in the presence of other factors, resulting in mis-interpretation and poor decision-making.

On the other hand, a multiple factor (MF) model such as $\lambda \sim SKU, N(DC), N(RatedPower), N(Workload),$ $N(CommissionYear)$ that normalizes the effects of other factors can isolate insights that pertains to just the SKU type. Fig. 15 presents the results for the two specific SKUs, S2 and S4, with marked difference in their reliability insights than suggested by the single factor approach. Note that, the relative ordering between the two compute SKUs are the same in both approaches. However, the SF approach grossly overestimates the impact of SKU S2 in its the average failure rate –10X the failure rate of S4 as opposed to just 4X determined by the MF model. The MF approach achieves this improvement by filtering out the noise in the failure rate observation associated with a SKU from other factors. As a result, the determined failure rates have a significant drop in variation (up to 50%) compared to the SF approach as seen from the error bars in Fig. 14 and Fig. 15.
An operator who pays a high cost premium for procuring S4, presuming that it has a much lower failure rate (1/10th) suggested by the SF model, may actually be making an error since the marginal impact of the SKU itself on the failure rate (1/4th) is actually much lower. And as a result, the operator may only procure S4 if the premium on cost is lower than originally thought acceptable. To illustrate this, we perform TCO analysis for two scenarios where the cost ratio of S4 to S2 is 1X and 1.5X respectively. When priced equally, both approaches estimate savings over 21% in using SKU S4 and their difference in estimation is only 3.9%. However, when S4 costs 1.5X as much as S2, then SF estimates 2.3% savings on TCO where-as MF estimates a loss of 3.2%. Paying a higher premium for S4 is not cost effective in this scenario as it will not lead to any cost savings.

Q3: How do environmental settings affect failures?

We answer question Q3 (Section II) by studying the relationship between environmental settings such as temperature, relative humidity, etc. and failure rates ($\mu$). Though we focus on temperature in the interest of space, the methodology applies to other environmental factors as well. We first use the SF model considering only the influence of temperature ($T$) on server failures. Fig. 16 shows the failure rate of racks grouped according to their average operating temperature on the day of failure. This plot suggests there is less variation in the mean of the failure rates among different groups identified by temperature range, but there is a high variation within each group.

We then use the MF approach to show how other influencing features can be abstracted to identify the true correlation of temperature to failures. Temperature is identified as a key factor for hard disk failures in MF approach. To illustrate this, we plot the disk failure rate for different ranges of operating temperature in Fig. 17. It shows a clear trend in hard disk failure rate with increase in operating temperature. We further delve in to the classification tree to focus on environmental correlations to hard disk failures by normalizing other factors such as age, SKU, workload, power rating, etc.

Fig. 18 shows a representative classification identified by the MF which presents several interesting insights: (i) Datacenter seems to have some influence on failures. These DCs differ in various aspects including cooling (see Sec. IV) which affects temperature and relative humidity. As we can see, HDD failures in DC1 show correlation to temperature and relative humidity, where-as DC2’s disk failure rate seem to be relatively unaffected with temperature and RH variations. (ii) For DC1, MF classification tree identifies temperature at 78°F as a splitting criteria with 50% increase in HDD failures rate($\mu$) when the DC operates above this temperature. (iii) While operating above 78°F, the model further identifies RH at 25% as a sub-branch splitting criteria, with additional increase of 25% in failure rate when operating below this RH. It is important to note that while setting the temperature and RH as identified by the MF can reduce failure rate and hence reduce the spare capacity cost, it may in turn increase the OpEx from adhering to the temperature/RH bounds. While DC operators can leverage the MF to identify the control knob settings for achieving desired availability targets, a more extensive analysis (considering cost of environment control) is required to minimize overall TCO. (iv) Temperature and RH exhibit interesting failure rate variation behavior. While temperature lower than 78°F results in lower variation in the failure rate, higher operating temperature and lower RH results in higher variation in failure rates. It shows that while some might like it hot (as observed in prior research [13]) within the group, others like it cold. Such key insights are possible because of the holistic evaluation of multiple features, compared to studying the influence of one or a few small subset of features at a time.

VII. CONCLUDING REMARKS

Understanding and mitigating failures is critical for the successful operation of datacenters. However, these failures can depend on many factors that it becomes very difficult to figure out where to even begin as in many of today’s “big data” problems. Many studies use aggregate behavior, or at best correlations between a small set of parameters for decision making and/or studying influences. In this paper, using a data from two production cloud-scale datacenters, we have shown the pitfalls of this approach. Instead, we need a systematic way for selecting and considering the different factors holistically when a datacenter operator is making important provisioning and operational decisions. We have demonstrated this approach with three important decision-making questions - how many spares to provision? which vendors should we procure from? what ranges to use for the various environmental knobs? Beyond helping make better decisions with these questions, the approach also gives additional insights on why the results behave in a certain way, as opposed to a black-box approach of looking at the results. There are several opportunities for leveraging and extending this framework in the future for other important issues including prediction of datacenter failures for pro-active maintenance and alleviation techniques.

VIII. ACKNOWLEDGMENTS

This work has been supported in part by NSF grants 1302225, 1302557, 1213052, 1439021, 1629129, and 1526750.
REFERENCES


