Using Dark Fiber to Displace Diesel Generators

Aman Kansal
Microsoft Research

Bhuvan Urgaonkar
Pennsylvania State University

Sriram Govindan
Microsoft

Abstract

Cloud providers and other data center operators are using geo-distributed data centers. But these data centers largely continue to employ the same designs as were appropriate for single data centers. These designs are wasteful because they do not take full advantage of geo-distribution. Geo-redundancy can reduce other redundancy at multiple intermediate layers in individual data centers and decrease costs. We discuss options for changing infrastructure design to realize such savings. Our proposal opens up an exciting and novel area of investigation into the design of software that can effectively leverage such platforms.

1 Introduction

We expect Internet services to be available 24/7. To achieve this, data center designers make every component in the data center highly available. For instance, servers have dual power supplies to overcome power supply failure. But this does not protect against failures in the power delivery infrastructure external to the server and to overcome this, data centers use dual cording and redundant power distribution circuitry. This does not protect against utility failures. Power backups (e.g., diesel generators) protect against utility failures, but not against facility wide outages (network backbone connection loss, fire, regional severe weather or earthquakes). Multiple geo-distributed data centers are built to overcome these failures. Since building a single data center to withstand such calamities is not cost effective, geo-distribution is the preferred solution [11].

We claim that *geo-redundancy at the data center level can reduce or obviate other redundancy* within individual data centers; much like app-layer replication via distributed file systems has obviated the need for highly reliable RAID arrays, allowing individual storage servers to use JBOD in data centers. We denote a group of reduced availability data centers that collectively deliver high availability as a Geo-distributed Bunch of Data centers (GBoD). We quantify the reduction in individual data center availability that becomes possible when a geo-distributed collection of data centers acts in concert using fault tolerance mechanisms. We discuss how an individual data center's infrastructural design can be changed to reduce its availability, such as through eliminating diesel generators, and quantify corresponding cost savings. We call attention to new research questions in configuring geo-redundant data centers to optimally manage excess compute capacity, networking costs, and correlation among their failures.

GBoD moves complexity from the underlying hardware to software layers. While recent research has designed fault tolerance mechanisms for applications spanning geo-distributed data centers [4, 6, 7, 13, 15], their techniques assume individual data centers are highly available. We argue why their design choices may break application behavior or degrade performance if naïvely applied to a GBoD. We discuss software design research issues to take advantage of lower cost data centers.

Our key contributions are to (1) quantify the reduction in individual data center availability with GBoD and discuss research problems in designing and configuring such lower availability data centers, and (2) present new research problems in designing system and application software for a GBoD, showing why existing techniques do not suffice.

2 How Much Can GBoD Save?

The first question we address is: by how much can the availability of individual data centers in a GBoD be lowered such that the overall availability goal is met? Knowing this, we can change the data center design to appropriately reduce the cost of individual data centers.

Suppose originally, a single data center with a capacity of S servers, or more generally S application instances, each comprised of appropriate compute servers, storage

nodes, and local area network (LAN) equipment was built, and had an availability of a_0. Typical Tier-4 data centers require $a_0 = 0.99999$, referred to as five 9s availability, and achieve it using battery backup, diesel generators, dual cording, and other redundancy. But Tier-4 design does not protect against large disturbances, such as earthquakes or severe weather, that can cause an entire data center to fail for extended periods [1]. Such downtimes decrease availability to below a_0. To preserve availability, the original capacity is split across n data centers (each with capacity S/n) and m redundant data centers of equal capacity are added. Microsoft and Google [3] already incorporate such geo-redundancy for their own applications. Both Windows Azure and Amazon AWS offer it to others. The data center may have been split into n without adding redundant capacity (i.e., with $m = 0$) to take the application closer to customers for low latency or for co-location with energy generation plants, but that would not increase availability. To preserve availability $m > 0$ must be used, and the cost of redundant capacity (mS/n) be paid for. For instance, Google's advertising backend [6] uses $n+m = 5$.

The required capacity of S servers is available if any n out of the $n+m$ data centers are available. But this is inefficient because the overall availability offered by $n+m$ data centers is higher than the original requirement.We could drop the availability of the individual data centers to a_g and still maintain the overall GBoD availability at a_0 as before[1]. The value of a_g can be computed as follows under some simplifying assumptions. Since the $n+m$ data centers are located far apart to be independent in terms of earthquake faults or weather effects, it is reasonable to assume that they are uncorrelated with respect to other failure modes (i.e., they are far apart on the power grid and network backbone so as not to fail together). Strictly speaking, power failures can be correlated even across large distances (e.g., cascade failures across the grid [5]), but we ignore such complexities for a first order analysis. To ensure that GBoD capacity is same as that of the original data center, we compute the probability, A_g, that n or more of $n+m$ lower availability data centers are available:

$$A_g = \sum_{i=n}^{n+m} \binom{n+m}{i} a_g^i (1-a_g)^{n+m-i}$$

The above equation sums up the probability that any i out of $n+m$ data centers are available, for $i = n$ to $n+m$.

To meet the original availability requirement, we must have $A_g \geq a_0$. Taking $a_0 = 0.99999$, and for some reasonable n,m pairs, some of which are used in Microsoft's applications, yields the a_g in Table 1.

[1]Limited data center capacity may be maintained at a_0 due to application requirements such as transactional guarantees at extremely low latency that GBoD cannot support.

n	m	a_g	Redundant capacity (m/n)
2	1	0.9982	50%
4	1	0.9990	25%
4	2	0.9921	50%
6	1	0.9994	16%
6	2	0.9944	33%
10	1	0.9996	10%
10	2	0.9965	20%
20	1	0.9998	5%
20	2	0.9982	10%

Table 1: GBoD tolerates orders of magnitude higher failure rates ($1 - a_g$) compared to current designs for required availability.

2.1 Inexpensive Data Center Designs

The next question is what practical design changes can reduce availability to a_g and if they reduce cost significantly. We illustrate such design options in the power delivery infrastructure, using realistic failure and recovery rates along with equipment costs to evaluate their cost-availability impact.

Table 2 lists availability and costs for major power related equipment in the data center. Utility power is the primary energy source and its availability calculation includes failures in utility provided substation or transformers. Diesel Generator (DG) backs up utility power and is the most expensive component in the power hierarchy. The Automatic Transfer Switch (ATS) switches the power source from the utility to DG and vice versa. The Uninterrupted Power Supply (UPS) units are required to transition between the utility and DG since DG incurs a start up delay of about 30 seconds. Power distribution units (PDUs) step voltage (480V-240V) and feed power to server racks through circuit breakers. Server racks may include a Static Transfer Switch (STS) for toggling between primary and secondary power lines when servers do not have dual-corded power supplies. Availability is computed using failure/recovery rates available in [12, 16]. Costs are obtained from APC [2].

Infrastructure Equipment	Availability	Cost
Utility	.999872	NA
Diesel Generator	.999599	1 $/W
Automatic Transfer Switch	.999943	10K$
Uninterrupted Power Supply	.999963	0.6 $/W
Power Distribution Unit	.999885	0.3 $/W
Rack ATS (STS)	.999999	0.1$/W
300W Server	NA	5 $/W

Table 2: Availability and cost of major power infrastructure equipment (source [12, 16]). NA: Not available.

Data center designs use multiple configurations of the above components. For instance, a data center that needs

n UPS units to meet its power capacity may use $2n+1$ units in a Tier-4 configuration for high availability. To reduce the reliability from a_0 to a_g, we can change these configurations, by reducing the redundancy in individual components or even eliminating components. We compute the availability for multiple such configurations using Reliability Block Diagrams [16] (RBDs). RBDs provide an effective model for estimating the availability of a network of components connected in a simple topology (series/parallel/standby etc.), assuming component failures are independent. For a typical data center topology, utility and DG are assumed to be in parallel since both can individually power the data center. UPS units are not treated as a parallel power source but are in series with the rest of the power network [9,16], to assist in DG transitions and to suppress the relatively frequent voltage/frequency fluctuations in utility power. Due to the latter function, UPS cannot be removed even if no DG is used. ATS, PDU, and server power supplies (PSUs) are in series with rest the power network. ATS is removed if no DG is used. While we consider data center level UPSs, additional UPS configurations are possible and analyzed in [9].

Table 3 shows cost and availability for some possible configurations, for a 10MW data center. This size is large enough to ensure that all component costs take advantage of economies of scale. The n,m notation represents the redundancy in each component; n is the number required to meet capacity and m units are redundant. The value of n depends on the maximum capacity of a single cost effective unit. For instance, UPS is priced best when purchased in 500kW to 600kW at 5 minutes of energy storage, leading to $n = 20$ for a 10MW data center. DGs are cost effective at 2.5MW, implying $n = 4$, and so on.

C0-C4 reduce the redundancy of components while C5 and C6 eliminate the DG altogether. As we move from C0 to C6, the power infrastructure cost falls by more than 2X. Data centers also store diesel fuel for DGs and refresh it after being stored for some time because the fuel degrades and using degraded fuel can damage the DGs. When the DG is eliminated (configuration C5), this expense can be removed as well. This cost is not accounted for in the table and will lead to additional savings.

C5 is the lowest cost configuration that satisfies the a_g limit for many of the choices in Table 1, while C1 can provide five 9s in a single data center. Table 4 depicts some possible data center designs that combine these configurations into a GBoD of $n+m$ data centers. Designs D1 and D2 are clearly over-available and costs drop significantly if either of D3-D10 is used. All of D3-D10 except D6 and D8 provide five 9s. Designs D3, ..., D10 vary in the redundant ratio, m/n, of servers that is used and the cost of networking n locations. Large online service providers and cloud operators building several 10s of megawatts in data center capacity every year can save substantially through such changes.

Config#	DG	UPS	PDU	Avail.	Infra. Cost
C0	(4,1)	(20,1)	(80,2)	.999999	21.89
C1	(4,0)	(20,1)	(80,2)	.999997	19.39
C2	(4,0)	(20,1)	(80,1)	.999956	19.35
C3	(4,0)	(20,0)	(80,1)	.999235	19.05
C4	(4,0)	(20,0)	(80,0)	.990331	19.01
C5	(0,0)	(20,1)	(80,2)	.998999	9.39
C6	(0,0)	(20,0)	(80,0)	.989199	9.01

Table 3: Cost (in Million USD) and availability for select infrastructure configurations. In an n,m tuple, n represents minimum capacity needed while m represents redundant units.

Design#	(Config#, n, m)	Avail	Power Infra.
D1	(C1,2,1)	0.99999999	28.5
D2	(C1,4,1)	0.99999999	23.75
D3	(C5,2,1)	0.999997	13.5
D4	(C5,4,1)	0.999990	11.25
D5	(C5,5,2)	0.999999	12.6
D6	(C5,10,1)	0.999937	9.9
D7	(C5,10,2)	0.999999	10.8
D8	(C5,20,1)	0.999777	9.45
D9	(C5,20,2)	0.999998	9.9
D10	(C5,20,3)	0.999999	10.35

Table 4: Cost (in million USD) and availability of a 10 MW data center using GBoD with $n+m$ data centers.

3 Research Challenges and Opportunities

3.1 GBoD Infrastructure Provisioning

Although our analysis suggests cost improvements, it is admittedly simplistic, and we need a more comprehensive cost-benefit analysis to tune data center designs.

Network: An obvious source of extra costs in a GBoD is the additional network bandwidth. Compared to a set of highly available data centers, greater network capacity is needed in the GBoD due to more aggressive consistency maintenance and more frequent recovery from failures. These costs must be offset by the savings due to reduced redundancy at individual data centers. Geo-distributed data center operators that have already invested in "dark fiber" (including Google, Amazon, and Microsoft) are well-placed in terms of this trade-off. Also, network costs have historically trended downward, making GBoD more attractive over time.

Inter-datacenter correlations: While we assumed failures to be independent across data centers, actual correlation depends on the locations of GBoD data centers.

As distance between two data centers grows, failures are less likely to be correlated. This, however, comes at the cost of higher latency between the two, suggesting that completely independent failures might be hard to realize. Since the effects of correlated failures are significantly more difficult to model than independent failures, they often lead to conservative designs [17], wiping out part or all of GBoD savings. Collecting detailed information about the failures of various infrastructural elements and their relationships would be crucial both in assessing the eventual efficacy of a GBoD's design and its overall profitability. Evidence suggests that often non-trivial dependencies (e.g., cascade failures [5]) exist among failures even in utilities that are thousands of miles apart. Understanding these complex relationships is by no means easy and is an active area of research. When correlation cannot be eliminated, greater redundancy (higher m) must be used, and we must optimize this trade-off to optimize cost and performance.

Data center Configuration. While we quantified cost and availability for power infrastructure design changes, other options exist as well. Five 9s availability calls for 24/7 support staff to assist in recovery at any time. At lower availability of a_g, 8/5 support suffices. This reduces personnel costs. Servers themselves could be made less reliable such as through lower cost hardware or casing. For instance, a single power supply unit and fan may be used instead of redundant ones. Server hardware repairs such as disk replacements that are performed daily could be batched over a week or more. Server equipment failures have been studied on large-scale data [8, 18, 19, 22], and these insights can be used to analyze cost and availability. Given that servers are refreshed every 3-4 years while infrastructure lasts 12-15 years, reducing the cost of servers may yield greater cost savings for the same loss of availability. A more comprehensive analysis of availability and cost change would identify which design changes are most effective.

Application Heterogeneity: Another key provisioning question arises from the diversity in performance and availability needs of applications [14]. It is likely that an effective GBoD be designed to allow *heterogeneity* both within and across its constituent data centers (e.g., some data centers employing higher availability infrastructure or more IT capacity than others).

3.2 GBoD Software Design

Whereas GBoD might be more cost-effective than traditional data centers, there are significant differences between these two platforms that necessitate a re-think of software design. Some applications such as read-intensive Web servers, for which writes are already relatively slow due to human editing and curating, will likely not notice any performance impact due to the extra latencies in synchronizing the writes across multiple data centers. On the other hand, extremely latency-sensitive applications, such as financial trading co-located in stock-exchange data centers, may never use GBoD due to high revenue loss at high latency. However, many applications lie between these two extremes and can benefit from GBoD with appropriate design changes.

Perhaps the most significant platform difference is the increased latency between servers in different data centers. Even when connected to a high-speed network backbone, latencies in GBoD are far from those in a data center LAN. Typical data center latencies are under a millisecond. GBoD latencies are measured below.

Experiment: We deployed TPC-W [21], an e-commerce benchmark, as an example application in six of Microsoft's geo-distributed data centers (Figure 1). These data centers are connected to a high capacity network backbone. We ran our client in one of these data centers (east-US) eliminating any effects due to congestion or queuing in the edge networks, and measured latencies when requests were served by each of the six different data centers. Figure 2 shows the cumulative distribution of the measured latencies. The figure confirms that latencies increase with distance as expected. These latencies are high enough to affect user experience and revenues [10, 20].

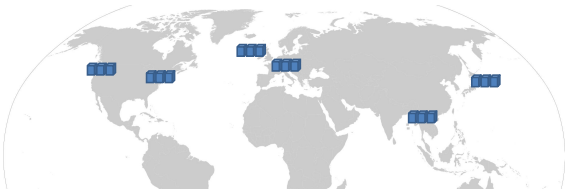

Figure 1: Location of the six geo-distributed data centers used in our experiments.

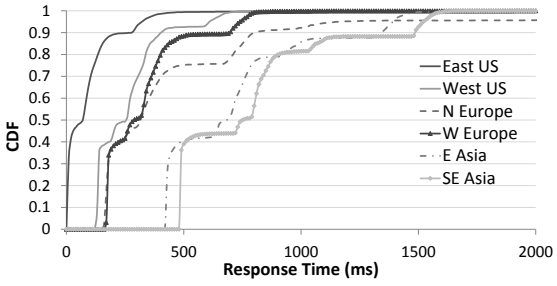

Figure 2: Latencies for six of Microsoft's geo-distributed data centers serving a client in East-US.

Challenges: Fortunately, prior work has studied fault tolerance mechanisms spanning multiple data centers [3,

4, 6, 7, 13, 15]. Their techniques deliver certain application performance guarantees while making carefully selected sacrifices. For instance, Amazon Dynamo [7] sacrifices data consistency (for certain kinds of requests, e.g., browsing) to achieve low latency. Google Spanner [6] and Megastore [3] maintain strong consistency but tolerate higher latency. Windows Azure Storage (WAS) [4] and COPS [15] achieve low latency by making geo-replication asynchronous at increased data risk in case of data center failures. MDCC [13] optimizes latency for replication across data centers by tuning the consistency protocol.

Two challenges emerge when we attempt to migrate existing applications to GBoD using the above techniques. First, existing techniques assume data centers to be highly available and are *not optimized for the reduced cost and availability* ranges of GBoD data centers. For instance, WAS and COPS use synchronous replication within a data center to protect against hardware failures but only asynchronously replicate across data centers since data center failure is assumed to be rare. When failures of entire data centers becomes more frequent (e.g., due to elimination of power backups), the resultant data loss may break application functionality. Strongly consistent systems such as Spanner may see frequent fluctuations in performance due to recovery tasks when failures occur more often. Divergence in weakly-consistent data replicas in Dynamo may become unacceptable when data center failures become more frequent.

Second, the trade-offs in existing techniques *do not transparently apply to every application*. Depending on consistency and latency requirements, the most suitable options among the existing techniques must be selected.

We classify the research in software design needed to realize the GBoD vision into three categories, below.

GBoD Platform Services: GBoD can offer certain platform services to applications. Virtual private networks (VPNs) to connect application instances across multiple data centers for secure networking, with potential bandwidth reservations to reduce queuing delays is one appealing example. A distributed memory cache spread across multiple data centers to maintain application session state, geo-distributed storage [4], key-value stores [7], and relational databases [6] (some of which are already used in large online applications and even offered as cloud services) may be offered with their designs optimized for the reduced availability and higher latencies of a GBoD.

In addition, the OS used on servers in a GBoD must be optimized for lower availability hardware, power, and network. For instance, while file systems typically persist i-nodes immediately to disk, they may lose file stream updates in case of power failures. GBoD servers may require a journaling file system that better recovers lost state on reboot. The network stack used may need similar optimizations. For instance, when a data center becomes unavailable, user requests may have to be redirected by external load-balancers or DNS servers. Existing redirection mechanisms are not optimized for low latency under frequent failures.

GBoD Abstractions: The platform services will not necessarily hide the fundamental platform differences in GBoD, and certain performance limitations will remain. To take advantage of cost efficiencies, applications may additionally require abstractions and design patterns to work with GBoD characteristics. System designers must determine the abstractions and interfaces offered. Latency and jitter in latency may be exposed as indicated in [13]. A GBoD based cloud could also expose differential costs of VM instances with higher availability instances being more expensive, in effect "virtualizing" availability. Research and development experience with the platform will lead to desirable GBoD OS interfaces, such as allowing only asynchronous writes, requiring session state to be explicitly categorized as reliable (geo-distributed) or unreliable (local), or enforcing default failure mode behavior. Design patterns will also aid developers. For instance, a pattern that minimizes write-related round trips in a single request would make an application GBoD friendly. Client side caching and state reclamation in case of data center failure would improve the user experience. We believe that while GBoD characteristics lead to many challenges, a rich set of ideas are available to overcome these challenges and will lead to fruitful research.

Application Configuration: Platform defaults will not always suit all applications. For instance, a travel booking application may use a geo-distributed memory cache to maintain its session state to preserve user progress in case of data center failure. However, the number of data centers across which the state is replicated will affect the latency, reliability, and bandwidth cost. Another application, such as photo sharing, may tolerate loosing session state as long as persistent data is maintained. Different tiers within an application may have different redundancy requirements. A content cache or CDN node remains useful at low availability while an authentication service would not.

Multiple configuration decisions and parameters affect the cost, performance, and availability for an application: number of data centers across which an application is spread out, degree of redundancy for each application component, geographic spread and selection of data centers, bandwidth reserved, and failure or recovery mechanism employed. Automated or human assisted configuration tools offer a major research opportunity for accelerating application development.

References

[1] AMAZON-AWS. Summary of the AWS service event in the US east region. http://aws.amazon.com/message/67457/.

[2] APC Datacenter Equipment. http://www.apc.com/products/category.cfm?id=13&subid=50, 2012.

[3] BAKER, J., BOND, C., CORBETT, J. C., FURMAN, J., KHORLIN, A., LARSON, J., LEON, J.-M., LI, Y., LLOYD, A., AND YUSHPRAKH, V. Megastore: Providing scalable, highly available storage for interactive services. In *Proceedings of the Conference on Innovative Data system Research (CIDR)* (2011), pp. 223–234.

[4] CALDER, B., WANG, J., OGUS, A., NILAKANTAN, N., SKJOLSVOLD, A., MCKELVIE, S., XU, Y., SRIVASTAV, S., WU, J., SIMITCI, H., HARIDAS, J., UDDARAJU, C., KHATRI, H., EDWARDS, A., BEDEKAR, V., MAINALI, S., ABBASI, R., AGARWAL, A., HAQ, M. F. U., HAQ, M. I. U., BHARDWAJ, D., DAYANAND, S., ADUSUMILLI, A., MCNETT, M., SANKARAN, S., MANIVANNAN, K., AND RIGAS, L. Windows azure storage: a highly available cloud storage service with strong consistency. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles* (New York, NY, USA, 2011), SOSP '11, ACM, pp. 143–157.

[5] CHEN, J., THORP, J. S., AND DOBSON, I. Cascading dynamics and mitigation assessment in power system disturbances via a hidden failure model. *Int. J. Elec. Power and Ener. Sys., 27(4):318-326* (2005).

[6] CORBETT, J. C., DEAN, J., EPSTEIN, M., FIKES, A., FROST, C., FURMAN, J. J., GHEMAWAT, S., GUBAREV, A., HEISER, C., HOCHSCHILD, P., HSIEH, W., KANTHAK, S., KOGAN, E., LI, H., LLOYD, A., MELNIK, S., MWAURA, D., NAGLE, D., QUINLAN, S., RAO, R., ROLIG, L., SAITO, Y., SZYMANIAK, M., TAYLOR, C., WANG, R., AND WOODFORD, D. Spanner: Google's globally-distributed database. In *Proceedings of the 10th USENIX conference on Operating Systems Design and Implementation* (Berkeley, CA, USA, 2012), OSDI'12, USENIX Association, pp. 251–264.

[7] DECANDIA, G., HASTORUN, D., JAMPANI, M., KAKULAPATI, G., LAKSHMAN, A., PILCHIN, A., SIVASUBRAMANIAN, S., VOSSHALL, P., AND VOGELS, W. Dynamo: amazon's highly available key-value store. In *Proceedings of twenty-first ACM SIGOPS symposium on Operating systems principles* (New York, NY, USA, 2007), SOSP '07, ACM, pp. 205–220.

[8] GILL, P., JAIN, N., AND NAGAPPAN, N. Understanding network failures in data centers: measurement, analysis, and implications. In *Proceedings of the ACM SIGCOMM 2011 conference* (New York, NY, USA, 2011), SIGCOMM '11, ACM, pp. 350–361.

[9] GOVINDAN, S., WANG, D., CHEN, L., SIVASUBRAMANIAM, A., AND URGAONKAR, B. Towards Realizing a Low Cost and Highly Available Datacenter Power Infrastructure. In *Proceedings of the Workshop on Power Aware Computing and Systems (HotPower)* (2011).

[10] GREENBERG, A., HAMILTON, J., MALTZ, D. A., AND PATEL, P. The cost of a cloud: research problems in data center networks. *SIGCOMM Comput. Commun. Rev. 39*, 1 (Dec. 2008), 68–73.

[11] HAMILTON, J. Inter-datacenter replication and georedundancy. http://perspectives.mvdirona.com/2010/05/10/InterDatacenterReplicationGeoRedundancy.aspx, 2010.

[12] IEEE Recommended Practice for the Design of Reliable Industrial and Commercial Power Systems. *ANSI/IEEE Std 493-2007* (2007).

[13] KRASKA, T., PANG, G., FRANKLIN, M. J., AND MADDEN, S. Mdcc: Multi-data center consistency. *CoRR abs/1203.6049* (2012).

[14] LIMRUNGSI, N., ZHAO, J., XIANG, Y., LAN, T., HUANG, H. H., AND SUBRAMANIAM, S. Providing reliability as an elastic service in cloud computing. In *Communications (ICC), 2012 IEEE International Conference on* (june 2012), pp. 2912 –2917.

[15] LLOYD, W., FREEDMAN, M. J., KAMINSKY, M., AND ANDERSEN, D. G. Don't settle for eventual: scalable causal consistency for wide-area storage with cops. In *Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles* (New York, NY, USA, 2011), SOSP '11, ACM, pp. 401–416.

[16] MCCARTHY, K. APC White paper 75: Comparing UPS System Design Configurations, 2008.

[17] NATH, S., YU, H., GIBBONS, P. B., AND SESHAN, S. Subtleties in tolerating correlated failures in wide-area storage systems. In *NSDI* (2006).

[18] SCHROEDER, B., AND GIBSON, G. A. Understanding disk failure rates: What does an mttf of 1,000,000 hours mean to you? *Trans. Storage 3*, 3 (Oct. 2007).

[19] SCHROEDER, B., PINHEIRO, E., AND 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* (New York, NY, USA, 2009), SIGMETRICS '09, ACM, pp. 193–204.

[20] SCHURMAN, E., AND BRUTLAG, J. Performance related changes and their user impact. In *Presented at Velocity Web Performance and Operations Conference* (2009).

[21] TOTOK, A. TPC-W (Web Commerce) Benchmark. `http://www.cs.nyu.edu/totok/professional/software/tpcw/tpcw.html`, 2005.

[22] VISHWANATH, K. V., AND NAGAPPAN, N. Characterizing cloud computing hardware reliability. In *Proceedings of the 1st ACM symposium on Cloud computing* (New York, NY, USA, 2010), SoCC '10, ACM, pp. 193–204.