Physics-Informed Machine Learning for DRAM Error Modeling

Elisabeth Baseman*, Nathan DeBardeleben*, Sean Blanchard*, Juston Moore§,
Olena Tkachenko¶, Kurt Ferreira†, Taniya Siddiqua‡, and Vilas Sridharan‡
*Ultrascale Systems Research Center, Los Alamos National Laboratory[1]
§Advanced Research in Cyber Systems, Los Alamos National Laboratory
Email: {lissa, ndebard, seanb, jmoore01}@lanl.gov
¶New Mexico Consortium
Email: otkac001@fiu.edu
†Center for Computing Research, Sandia National Laboratories
Email: kbferre@sandia.gov
‡RAS Architecture, Advanced Micro Devices, Inc.
Email:{Taniya.Siddiqua, Vilas.Sridharan}@amd.com

***Abstract*—As the scale of high performance computing facilities approaches the exascale era, gaining a detailed understanding of hardware failures becomes important. In particular, the extreme memory capacity of modern supercomputers means that data corruption errors which were statistically negligible at smaller scales will become more prevalent. In order to understand hardware faults and mitigate their adverse effects on exascale workloads, we must learn from the behavior of current hardware. In this work, we investigate the predictability of DRAM errors using field data from two recently decommissioned supercomputers: Cielo, at Los Alamos National Laboratory, and Hopper, at Lawrence Berkeley National Laboratory. Due to the volume and complexity of the field data, we apply statistical machine learning to predict the probability of DRAM errors at previously un-accessed locations. We compare the predictive performance of six machine learning algorithms, and find that a model incorporating physical knowledge of DRAM spatial structure outperforms purely statistical methods. Our findings both support expected physical behavior of DRAM hardware as well as providing a mechanism for real-time error prediction. We demonstrate real-world feasibility by training an error model on one supercomputer and effectively predicting errors on another. Our methods demonstrate the importance of spatial locality over temporal locality in DRAM errors, and show that relatively simple statistical models are effective at predicting future errors based on historical data, allowing proactive error mitigation.**

I. Introduction

Today's supercomputers / high performance computers (HPC) use a combination of commodity and specialized hardware and software to achieve extremely high performance at a reasonable cost. Common HPC workloads require massive memory capacities; the top 5 supercomputers in the world in November 2017 had between 710 terabytes and 1.3 petabytes of main memory [1]. In the next decade, and likely beyond, supercomputers are expected to more than quadruple in computational power. Memory capacities are estimated to achieve in the tens to hundreds of petabytes on these future systems, further increasing the need to understand the nuances of memory errors and faults. Capacities of this scale require advanced error correction to mitigate a near constant rate of errors from a variety of environmental (transient) and intrinsic (permanent) sources. These sources include, but are not limited to, terrestrial neutrons [2], temperature [3], alpha particles [4], voltage fluctuations [5], and aging [6], [7].

Ultimately, fault tolerance impacts the supercomputer's reliability as presented to users. Users expect systems to be deterministic, but errors cause random behavior. Errors that can be detected but not corrected cause the supercomputer nodes to crash, while rare undetected errors *could* cause silent corruption of calculations and potentially undermine the integrity of science done on the systems. It is important to understand that in this work we are studying DRAM *errors* instead of their underlying *faults*. Many errors can come from a single fault and understanding the mapping from errors to faults is extremely valuable and has implications for hardware design [8], [9]. In this work we instead apply modeling techniques to the raw, unprocessed, data that comes from the system; namely, errors.

Understanding the way in which errors and faults manifest in main memory (DRAM) impacts the ways in which hardware designers implement error detection and correction mechanisms. In this work, we model both the transient environmental errors and permanent intrinsic sources using a data-driven approach. The primary contributions of our work are:

- Identification of features relevant to a statistical model of DRAM errors and underlying physics

[1]This work was performed at the Ultrascale Systems Research Center (USRC) at Los Alamos National Laboratory, supported by the U.S. Department of Energy contract AC52-06NA25396. The publication has been assigned the LANL identifier LA-UR-LA-UR-18-23333 and the Sandia identifier SAND2018-8118C. Sandia National Laboratories is a multi mission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. AMD, the AMD Arrow logo, AMD Opteron, and combinations thereof are trademarks of Advanced Micro Devices, Inc. Product names used in this publication are for identification purposes only and may be trademarks of their respective companies.

978-1-5386-8398-9/18/31.00©2018*IEEE*

- Comparison of a variety of predictive machine learning models for determining the likelihood of a DRAM location to experience an error
- Development of a computationally efficient physics-based machine learning model for spatial modeling of DRAM errors
- Demonstration of error model transferability between HPC implementations

II. Related Work

Many published studies have examined failures in production cloud and HPC systems [10], [11], [12], [13], [14], [15], [16], [9], [17], [8], [18], [19]. These studies typically focus on characterizing failures and trying to understand the underlying reliability of the large-scale system.

Additional studies have attempted to use these failure characterization studies to predict eminent failures as well as to investigate important spatial and temporal correlations amongst failure events [20], [21]. Schroeder and Gibson studied failure event logs from a number of high-performance clusters and discovered day-of-the-week and hour-of-the-day temporal correlations between failure events [10]. Yigitbasi *et al.* found similar correlations [22]. Liang *et al.* exploited the spatial correlation between failure events in a BlueGene/L system to predict 80% of memory and network failures [23]. In addition, Fu and Xu used a spherical correlation model to develop two algorithms to cluster different events in a system according to their spatial and temporal correlations [24]. They applied a neural network predictor to predict the time-between-failures in HPC systems. Oliner, Kulkarni, and Aiken discovered a structure of causal influence between the components of HPC systems by calculating time-lagged cross-correlation and analyzing how the signals from different system components deviate from their normal behavior [25]. Goudarzi *et al.* investigated the power of cross- and partial-correlations to observe conditional correlations in HPC failure event data and used information theory to understand the fundamental predictive power of HPC failure data [26]. Baseman *et al.* used discriminative machine learning models to predict which type of fault a DRAM device is experiencing, based on the time series information of errors collected from the device [27], [28]. Lastly, Gupta *et al.* found a spatial component of failures does indeed exist [29].

Our work distinguishes itself from existing studies in several ways. First, our study analyzes the entire lifetime of data from a recent leadership-class system and contains a much larger corpus of data than previous studies. Second, this work focuses specifically on fine-grained DRAM errors (a significant and increasing source of failure on these systems) and their locality trends, unlike previous work which focused on node-level failures. Finally, our present work focuses on quantitative statistical models with the ability to encode physical relationships to learn locality trends from empirical data, rather than relying on human expert-elicited parameters.

III. Available Data

Our data comes from two supercomputers in the U.S. Department of Energy (DOE): Cielo at Los Alamos National Laboratory (LANL) and Hopper at Lawrence Berkeley National Laboratory (LBNL). Cielo was a 8,944 node Cray supercomputer with 286 TB of DDR3 DRAM memory. Hopper had 6,384 nodes with 212 TB of DDR3 DRAM memory.

In both cases, correctable DRAM memory errors were logged by the system software. Cielo's error correction was slightly more elaborate Chipkill (single symbol correct, double symbol detect) while Hopper's correction was single symbol detect. This was a design decision due to LANL's higher altitude ($\sim$8,000 feet) compared to LBNL at roughly sea level.

All memory errors include a timestamp, the compute node that identified the error, and a "syndrome" which we decode to determine high fidelity location information about the error event. This error location includes the DRAM channel, lane, rank, bank, row, column, and number of bits that were corrected. For Cielo, we have over 700 thousand correctable DRAM error events for four and a half years from July 2011 to March 2016. This constitutes roughly the full operational lifetime of Cielo. For Hopper, we have the first 22 months of operation from April 2011 through February 2013. This includes over 400 thousand correctable error events.

IV. Potential Models

Given a set of observed error locations in a DRAM device, and a proposed access location, we aim to estimate the probability that accessing this new location will result in an error. Due to the sheer volume and complexity of our DRAM field data, we turn to statistical machine learning techniques in order to model DRAM error behavior in a data-driven way. We compare the performance of a variety of machine learning models on this error likelihood estimation task, ranging from purely statistical out-of-the-box models (which do not encode any spatial knowledge) to more nuanced and hand-crafted, physics-informed models. What follows is a brief description of each of the machine learning models we examine.

A. Purely Statistical Approaches

1) Random Forest (RF): Random forests are supervised machine learning classifiers that use an ensemble of decision trees built from a single training dataset [30]. The advantage of a random forest as compared to a single decision tree is that it preserves the variance of a single tree while decreasing the bias — i.e., each decision tree in the ensemble is allowed to overfit as long as the number of trees is large. For this reason, random forests frequently work quite well for a variety of tasks without much parameter tuning. Each decision tree within the random forest randomly selects a subset of available features to consider at each split, so creating a sufficiently large forest guarantees that each feature will be considered at least once. For our problem, we use a random forest to predict the probability of a given DRAM location to produce an error, given the locations and frequencies of previous errors observed on the same DRAM device.

2) Boosted Trees (BT): The boosted decision tree algorithm is similar to a random forest in that it is a supervised ensemble classifier consisting of decision trees, but the boosting algorithm changes the behavior of the ensemble [31]. When creating an ensemble of learners (in this case, decision trees) using the boosting algorithm, we start by letting each tree have rather weak performance, but on subsequent trees we increase the weighting of training samples which were miss-classified by the previous tree. This method frequently results in slightly better classification performance than a random forest because a boosted ensemble has the ability to correct itself as it is trained, whereas each decision tree within a random forest is trained independently. Again, a boosted ensemble of decision trees involves very few parameters to tune, mainly the number of learners in the ensemble, and so is easy to use out-of-the-box. We train our boosted ensemble to predict the probability of a DRAM location experiencing an error given the locations and frequencies of previously observed errors on the same DRAM device.

3) Support Vector Machine (SVM): Support vector machines (SVMs) are another type of supervised machine learning classifier, but which have more nuanced parameters [32]. SVMs project the training dataset into a higher dimensional space using a kernel function. This function can be chosen by hand, and in our work, we choose the commonly used radial basis function kernel. In this higher dimensional space, the decision surface between possible classification outcomes becomes linear, and the SVM learns this surface. The result is then projected back down into the original dimensional space, and thus the SVM can learn a nonlinear decision surface relatively efficiently. In our work, the SVM is trained to predict whether a given DRAM location will produce an error given previous error locations and frequencies on the same DRAM device.

4) Neural Network (Multilayer Perceptron, MLP): A multilayer preceptron (MLP), which is a simple neural network, is a supervised machine learning classifier based on the architecture of the brain, which can learn a nonlinear decision surface via backpropagation [33]. While these neural networks, particularly deep neural networks, have seen numerous successes in recent years, they are difficult to tune and often require significant engineering of their features and architectures for optimal performance. We use a small MLP model to predict whether a given DRAM location will experience an error given the previous error locations and frequencies on the same DRAM device.

5) Naïve Bayesian Classifier (NB): The Naïve Bayesian classifier (NB) is an efficient machine learning model that uses only marginal statistics of its training data. It is a directed probabilistic graphical model which assumes independence between all features and exploits Bayes' Theorem for learning and inference [34]. Although simple, Naïve Bayes has been shown to be surprisingly effective, and is especially known for its usefulness in spam detection and filtering [35]. We use a Naïve Bayesian Classifier to predict whether a given DRAM location will experience an error given the previous error locations and frequencies on the same DRAM device.

B. Physics-Informed Approach: Markov Random Field (MRF)

We develop a statistical machine learning model for DRAM errors which also makes use of spatial knowledge of the hardware via a Markov random field model. Markov random fields (MRFs) are undirected probabilistic models which encode dependencies between features [32]. One of the simplest versions of an MRF is a grid structure, known as an Ising Model in physics. MRF models, originally used for magnets and for spin models, are especially effective at learning spatial correlations (such as neighbor, row, and column errors in DRAM). In our MRF model, we learn how strongly an error experienced in one location effects its surrounding locations. Learning in MRFs is accomplished by fitting weights such that the lowest energy state corresponds to the most physically preferable configuration, where the energy of a configuration is given by:

$$H_{\texttt{MRF}} = -\sum_{i,j,i\neq j} (a_{i,j}x_i x_j + r_{i,j}x_i x_j + c_{i,j}x_i x_j) \quad (1)$$

where X is a given configuration of settings for all nodes in the grid, i, j index all pairs of nodes in the grid configuration, $a_{i,j}$ is the corresponding weight between nodes i and j if i is directly adjacent to j and 0 otherwise, $r_{i,j}$ is the corresponding weight between nodes i and j if i and j are in the same row and 0 otherwise, and $c_{i,j}$ is the corresponding weight between nodes i and j if i and j are in the same column and 0 otherwise. Then, it is well known that the probability of a particular given setting of all nodes, X, is:

$$P(X) = \frac{e^{H(X)}}{Z} \quad (2)$$

with

$$Z = \sum_X e^{-H(X)} \quad (3)$$

where Z is known as the *partition function*. This model is similar to the layout of a DRAM device, with its rows and columns, and architects expect there to be dependencies between spatially-related DRAM locations. For example, we know that DRAM devices are connected across rows and columns, the effects of which can be seen in errors observed in the field [36]. Therefore, if one DRAM location experiences an error, we would expect the probability of adjacent locations to experience an error to increase (even in the case of a random impact from a cosmic ray, as a ray travels through a device it would be more likely to affect directly adjacent locations), as well as the probability of an error in locations in the same row and column to increase due to the physical architecture of the device. For these reasons, we include undirected dependencies across rows and columns within the Ising model grid to obtain an "adapted Ising model" version of a simple MRF. The structure of this model is shown in Figure 1.

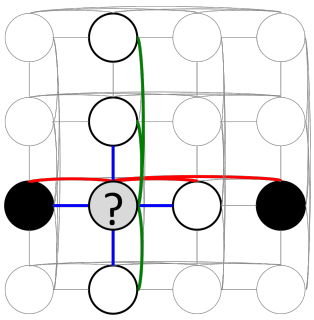

Fig. 1. The structure of our physics-informed adapted Ising model, a special case of a Markov Random Field. In this undirected probabilistic graphical model, we assume dependencies between adjacent locations and dependencies across rows and columns, as corresponds to the physics of DRAM devices. In the shown example, we are trying to predict whether the node with the question mark will experience an error. The other darkened nodes have already experienced an error. The nodes outlined in bold are involved in the probability calculation. Blue edges indicate relationships weighted with a "directly adjacent" weight, green edges indicate relationship weighted with a "column" weight, and red edges will be weighted with a "row" weight.

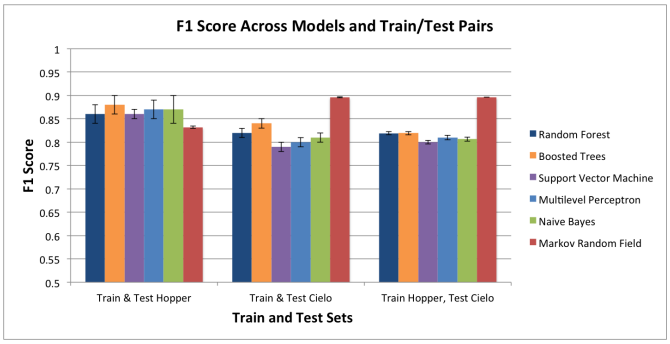

Fig. 2. Average F1 scores with standard deviation for each model (higher is better). Note that MRF appears to be the most general model.

In this MRF model, we have incorporated domain knowledge regarding some physical assumptions about DRAM devices into our statistical model. We can now use Equation 2 combined with our observed real DRAM data in order to learn the appropriate adjacent, row, and column weights, and then to infer the probability of a given DRAM location to experience an error given the previous error locations observed from the same DRAM device. Exact learning and inference procedures in Markov random field models would be computationally prohibitive in real-time. Therefore, due to the complexity of our partition function, we accomplish learning and inference via a logistic regression approximation to achieve a maximum pseudolikelihood estimate following Hunter [37]. This approximate approach makes our MRF model feasible for large-scale data and for real-time analysis.

V. Experimental Setup

For each of our six potential models (random forest, boosted trees, support vector machine, multilayer perceptron, naïve Bayes, and Markov random field), we investigate performance on three distinct train/test pairs: train and test on data from Hopper, train and test on data from Cielo, and train on Hopper test on Cielo. Because Hopper was an older machine than Cielo, these three train/test pairs not only allow us to evaluate the usefulness of each model on two different HPC machines — they also allow us to test the generality of each model. That is, if a model performs well when trained on an older system and tested on a newer system, this will mean that the model has learned something universal about DRAM error behavior, rather than overfit to a specific machine's architecture.

A. Features

For all models other than MRF, we generate our dataset using DRAM error location information from Hopper and/or Cielo. We take a sample of DRAM locations across time, and for each location we extract the total number of errors seen up to and including the current time in the same row, same column, to the top, to the left, to the right, and below the chosen location. We then also extract the average frequency at which we have seen errors in each of these relative locations. These counts and frequencies then become our features, and our ground truth as to whether the chosen location did ultimately experience an error becomes our label. A detailed description of the feature extraction process and specific features is available in prior work [27], [28].

For the MRF model, we also generate our dataset using DRAM error location information from Hopper and/or Cielo (depending on the train/test set pairing we are using). However, because this model encodes some knowledge of the underlying physics of the DRAM, we encode our features slightly differently. For each DRAM, we build up a final picture of which locations experienced errors; in the final array snapshot of the DRAM, locations which experienced an error are assigned a 1 for `TRUE` while locations which did not experience an error are assigned a 0 for `FALSE`. We then use a randomly selected subset of these arrays in the MRF model to learn our weights for the undirected network. We then randomly select locations from the held out DRAM arrays and use our trained model to predict whether the selected location will experience an error given the rest of the relevant DRAM snapshot.

B. Evaluation

For each of our models combined with each train/test set pair, we report average F1 score, which is the harmonic mean of precision and recall. For F1 score, a value of 1.0 indicates perfect predictive performance, while 0.5 indicates performance as good as random guessing. We also report average false positive and false negative rates.

VI. Results

Figure 2 shows average F1 scores with standard deviation. Note that across all train/test sets, among models that do not have knowledge of physics, the random forest and boosted trees models perform best. As expected, among these models, scores are slightly lower when trained on an older system and tested on a new system. However, these scores are still quite high, indicating that the models are indeed learning something

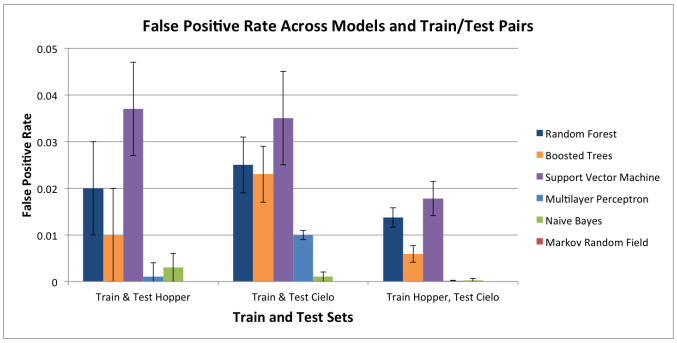

Fig. 3. Average false positive rates with standard deviation for each model (lower is better). MRF achieves nearly zero false positives.

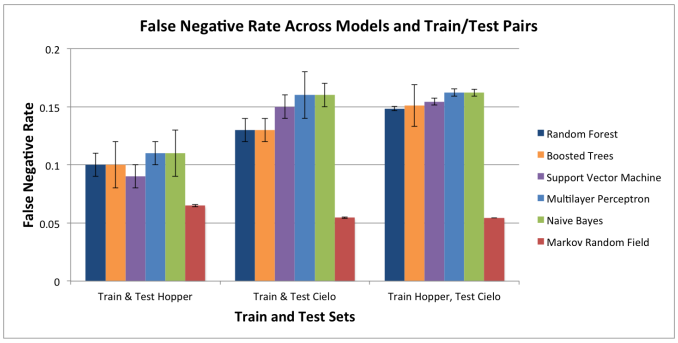

Fig. 4. Average false negative rates with standard deviation for each model (lower is better). MRF achieves significantly fewer false negatives.

general about DRAM behavior. However, we observe that the MRF model does as well at predicting the newer system when trained on an older system as it does when trained on the newer system. In fact, the MRF model does very well when trained and tested on two different systems, indicating that by giving the model access to observed data as well as some basic knowledge about the physics architects expect, we can create a general and accurate model of DRAM behavior.

Figure 3 shows false positive rates with standard deviation for each model for each train/test set pairing. This represents how frequently each model incorrectly predicts that a DRAM location will experience an error. Note that for this metric SVMs are our worst performers, while MLPs, NB, and MRFs perform very well. Models that do not have underlying physical knowledge are able to achieve near zero false positive rates, even when trained and tested on different systems. However, the MRF models all achieve near zero or exactly zero false positives regardless of train and test sets. Clearly given access to some physical knowledge as well as observed data allows the MRF statistical model to understand that errors are relatively rare.

Figure 4 shows false negative rates with standard deviation for each model for each train/test set pairing. That is, Figure 4 represents how frequently each model incorrectly predicts that a DRAM location is safe to access. Under this metric, we observe that random forests and boosted trees tend to be our best performer among physics-ignorant models. However, we again see that MRF, the physics-aware model, performs significantly better, even on temporally distinct train and test sets. In fact, the performance when trained and tested on different systems is nearly the same as when trained and tested on the same newer system. This suggests that giving the model the ability to learn about the strength of physical correlations in the DRAM devices gives a non-trivial performance improvement.

VII. Conclusion

We find that statistical machine learning models can predict the likelihood that a given DRAM location will experience an error with excellent predictive performance (up to an F1 score of 0.89 +/- 0.01). In our comparison of a variety of models, we find that among models which are ignorant of any underlying physics, random forests and boosted trees achieved the highest F1 scores and lowest false negative rates, while multilayer perceptrons and naïve Bayes classifiers achieve the lowest false positive rates. However, a Markov random field model which encodes a basic understanding of underlying physical connections between DRAM locations and is then trained using observed data performs best on all three metrics, even when trained on an older supercomputing system and tested on a newer system. In fact, using a model which has a very basic representation of some physical relationships between locations gives better performance than models which only rely on statistics, even when those physics-ignorant models are given extra information regarding temporal locality of errors. For these reasons, our work points to the usefulness of spatial locality over temporal locality in the analysis of DRAM errors. In addition, because our Markov random field model is approximated by a logistic regression classifier, the model is simple and fast enough to be potentially used in production, even on board future memory controllers.

Further work includes investigating other Markov random field configurations to further confirm the underlying physics within DRAM devices — this would allow researchers and architects to either confirm or question that the errors observed do indeed agree with expected physical behavior. In addition, these models could be used as architecture design tools to accurately simulate expected error/fault behavior in a new DRAM layout.

Acknowledgements

The authors thank John Shalf from Lawrence Berkeley National Laboratory, who provided the Hopper field data. Please see “Memory Errors in Modern Systems: The Good, The Bad, and The Ugly” for more on the LANL/SNL/LBNL/AMD collaboration on supercomputer memory errors [8].

References

[1] “TOP500 - November 2017,” https://www.top500.org/lists/2017/11/, accessed: April 16, 2018.

[2] N. Seifert, “Radiation-induced soft errors: A chip-level modeling perspective,” *Found. Trends Electron. Des. Autom.*, vol. 4, no. 2–3, pp. 99–221, Feb. 2010. [Online]. Available: http://dx.doi.org/10.1561/1000000018

[3] S. Jagannathan, Z. Diggins, N. Mahatme, T. D. Loveless, B. L. Bhuva, S. J. Wen, R. Wong, and L. W. Massengill, "Temperature dependence of soft error rate in flip-flop designs," in *2012 IEEE International Reliability Physics Symposium (IRPS)*, April 2012, pp. SE.2.1–SE.2.6.

[4] T. C. May and M. H. Woods, "Alpha-particle-induced soft errors in dynamic memories," *IEEE Transactions on Electron Devices*, vol. 26, no. 1, pp. 2–9, Jan 1979.

[5] V. Chandra and R. Aitken, "Impact of technology and voltage scaling on the soft error susceptibility in nanoscale cmos," in *2008 IEEE International Symposium on Defect and Fault Tolerance of VLSI Systems*, Oct 2008, pp. 114–122.

[6] E. H. Cannon, A. KleinOsowski, R. Kanj, D. D. Reinhardt, and R. V. Joshi, "The impact of aging effects and manufacturing variation on sram soft-error rate," *IEEE Transactions on Device and Materials Reliability*, vol. 8, no. 1, pp. 145–152, March 2008.

[7] D. Rossi, M. Omaa, C. Metra, and A. Paccagnella, "Impact of aging phenomena on soft error susceptibility," in *2011 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, Oct 2011, pp. 18–24.

[8] V. Sridharan, N. DeBardeleben, S. Blanchard, K. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi, "Memory errors in modern systems: The good, the bad, and the ugly," in *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS '15. New York, NY, USA: ACM, 2015, pp. 297–310. [Online]. Available: http://doi.acm.org/10.1145/2694344.2694348

[9] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi, "Feng shui of supercomputer memory: Positional effects in DRAM and SRAM faults," in *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '13. New York, NY, USA: ACM, 2013, pp. 22:1–22:11. [Online]. Available: http://doi.acm.org/10.1145/2503210.2503257

[10] B. Schroeder and G. Gibson, "A large-scale study of failures in high-performance computing systems," in *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*, 2006, pp. 249–258.

[11] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM errors in the wild: a large-scale field study," *Commun. ACM*, vol. 54, no. 2, pp. 100–107, Feb. 2011. [Online]. Available: http://doi.acm.org/10.1145/1897816.1897844

[12] X. Li, K. Shen, M. C. Huang, and L. Chu, "A memory soft error measurement on production systems," in *2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference*, ser. ATC'07. Berkeley, Calif., USA: USENIX Association, 2007, pp. 21:1–21:6. [Online]. Available: http://dl.acm.org/citation.cfm?id=1364385.1364406

[13] X. Li, M. C. Huang, K. Shen, and L. Chu, "A realistic evaluation of memory hardware errors and software system susceptibility," in *Proceedings of the 2010 USENIX conference on USENIX annual technical conference*, ser. USENIXATC'10. Berkeley, Calif., USA: USENIX Association, 2010, pp. 6–20. [Online]. Available: http://dl.acm.org/citation.cfm?id=1855840.1855846

[14] A. A. Hwang, I. A. Stefanovici, and B. Schroeder, "Cosmic rays don't strike twice: understanding the nature of dram errors and the implications for system design," in *Proceedings of the 17th international conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XVII. New York, NY, USA: ACM, 2012, pp. 111–122. [Online]. Available: http://doi.acm.org/10.1145/2150976.2150989

[15] N. El-Sayed, I. A. Stefanovici, G. Amvrosiadis, A. A. Hwang, and B. Schroeder, "Temperature management in data centers: why some (might) like it hot," in *Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on Measurement and Modeling of Computer Systems*, ser. SIGMETRICS '12. New York, NY, USA: ACM, 2012, pp. 163–174. [Online]. Available: http://doi.acm.org/10.1145/2254756.2254778

[16] T. Siddiqua, A. Papathanasiou, A. Biswas, and S. Gurumurthi, "Analysis of memory errors from large-scale field data collection," in *Silicon Errors in Logic - System Effects (SELSE), 2013 IEEE Workshop on*, 2013.

[17] V. Sridharan and D. Liberty, "A study of DRAM failures in the field," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '12. Los Alamitos, Calif., USA: IEEE Computer Society Press, 2012, pp. 76:1–76:11. [Online]. Available: http://dl.acm.org/citation.cfm?id=2388996.2389100

[18] C. Di Martino, Z. Kalbarczyk, R. K. Iyer, F. Baccanico, J. Fullop, and W. Kramer, "Lessons learned from the analysis of system failures at petascale: The case of blue waters," in *International Conference on Dependable Systems and Networks*, 2014.

[19] L. Bautista-Gomez, F. Zyulkyarov, O. Unsal, and S. McIntosh-Smith, "Unprotected computing: A large-scale study of dram raw error rate on a supercomputer," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '16. Piscataway, NJ, USA: IEEE Press, 2016, pp. 55:1–55:11. [Online]. Available: http://dl.acm.org/citation.cfm?id=3014904.3014978

[20] A. Gainaru, F. Cappello, M. Snir, and W. Kramer, "Fault prediction under the microscope: A closer look into hpc systems," in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, ser. SC '12. Los Alamitos, CA, USA: IEEE Computer Society Press, 2012, pp. 77:1–77:11. [Online]. Available: http://dl.acm.org/citation.cfm?id=2388996.2389101

[21] A. Gainaru, F. Cappello, J. Fullop, S. Trausan-Matu, and W. Kramer, "Adaptive event prediction strategy with dynamic time window for large-scale hpc systems," in *Managing Large-scale Systems via the Analysis of System Logs and the Application of Machine Learning Techniques*, ser. SLAML '11. New York, NY, USA: ACM, 2011, pp. 4:1–4:8. [Online]. Available: http://doi.acm.org/10.1145/2038633.2038637

[22] N. Yigitbasi, M. Gallet, D. Kondo, A. Iosup, and D. Epema, "Analysis and modeling of time-correlated failures in large-scale distributed systems," in *2010 11th IEEE/ACM International Conference on Grid Computing*, Oct 2010, pp. 65–72.

[23] Y. Liang, Y. Zhang, A. Sivasubramaniam, M. Jette, and R. Sahoo, "Bluegene/l failure analysis and prediction models," in *Proceedings of the International Conference on Dependable Systems and Networks*, ser. DSN '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 425–434.

[24] S. Fu and C.-Z. Xu, "Exploring event correlation for failure prediction in coalitions of clusters," in *Proceedings of the 2007 ACM/IEEE conference on Supercomputing*, ser. SC '07. New York, NY, USA: ACM, 2007, pp. 41:1–41:12.

[25] A. Oliner, A. Kulkarni, and A. Aiken, "Using correlated surprise to infer shared influence," in *Dependable Systems and Networks (DSN), 2010 IEEE/IFIP International Conference on*, 2010, pp. 191–200.

[26] A. Goudarzi, D. Arnold, D. Stefanovic, K. B. Ferreira, and G. Feldman, "A principled approach to hpc event monitoring," in *Proceedings of the 5th Workshop on Fault Tolerance for HPC at eXtreme Scale*, ser. FTXS '15. New York, NY, USA: ACM, 2015, pp. 3–10. [Online]. Available: http://doi.acm.org/10.1145/2751504.2751506

[27] E. Baseman, N. DeBardeleben, K. Ferreira, S. Levy, S. Raasch, V. Sridharan, T. Siddiqua, and Q. Guan, "Improving dram fault characterization through machine learning," in *46th International Conference on Dependable Systems and Networks*, 2016.

[28] E. Baseman, N. DeBardeleben, K. Ferreira, V. Sridharan, T. Siddiqua, and O. Tkachenko, "Automating dram fault mitigation by learning from experience," in *47th International Conference on Dependable Systems and Networks*, 2017.

[29] S. Gupta, D. Tiwari, C. Jantzi, J. Rogers, and D. Maxwell, "Understanding and exploiting spatial properties of system failures on extreme-scale hpc systems," in *2015 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, June 2015, pp. 37–44.

[30] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[31] Y. Freund and R. E. Schapire, "A desicion-theoretic generalization of on-line learning and an application to boosting," in *European conference on computational learning theory*. Springer, 1995, pp. 23–37.

[32] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[33] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, p. 533, 1986.

[34] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[35] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[36] V. Sridharan and D. Liberty, "A study of dram failures in the field," in *High Performance Computing, Networking, Storage and Analysis (SC), 2012 International Conference for*. IEEE, 2012, pp. 1–11.

[37] D. Hunter, "Estimation in ergms," in *ERGM Workshop 2006*.