# Automating DRAM Fault Mitigation By Learning From Experience

Elisabeth Baseman*, Nathan DeBardeleben*, Kurt Ferreira†,
Vilas Sridharan‡, Taniya Siddiqua ‡ and Olena Tkachenko*
*Ultrascale Systems Research Center, Los Alamos National Laboratory[1]
{lissa, ndebard, otkachenko}@lanl.gov
†Center for Computing Research, Sandia National Laboratories
kbferre@sandia.gov
‡RAS Architecture, Advanced Micro Devices, Inc.
{Vilas.Sridharan, Taniya.Siddiqua}@amd.com

***Abstract*—Current practice for mitigating DRAM hardware faults is to simply discard the entire faulty DIMM. However, this becomes increasingly expensive and wasteful as the price of memory hardware increases and moves physically closer to processing units. Accurately characterizing memory faults in real-time in order to pre-empt future potentially catastrophic failures is crucial to conserving resources by blacklisting small affected regions of memory rather than discarding an entire hardware component. We further evaluate and extend a machine learning method for DRAM fault characterization introduced in prior work by Baseman *et al.* at Los Alamos National Laboratory. We report on the usefulness of a variety of training sets, using a set of production-relevant metrics to evaluate the method on data from a leadership-class supercomputing facility. We observe an increase in percent of faults successfully mitigated as well as a decrease in percent of wasted blacklisted pages, regardless of training set, when using the learned algorithm as compared to a human-expert, deterministic, and rule-based approach.**

## I. Introduction

As the era of exascale computing approaches, the question of system reliability becomes paramount. With extremely large machines, even the least probable failure becomes significantly more likely to occur. The increase in machine size, particularly regarding DRAM and SRAM, is expected to be on the order of a hundred-fold [1]. Recent work by Gottscho *et al.* suggests that faults in hardware components such as DRAM can have serious negative consequences on the applications running on the supercomputing system [2]. Current approaches simply discard entire DIMM units when some threshold of errors are observed, but this will become too costly in the exascale era. Newer machines have the capability to blacklist only certain pages of DRAM which exhibit faults, in order to conserve time, finances, and hardware while still mitigating the existing fault [3], [4]. However, in order to make use of this capability, the machine or system administrator must be able to accurately identify the type of fault occurring in real-time.

In this work, we extend previous research on automatic characterization of DRAM faults [5] by considering production-relevant metrics, studying a larger dataset covering the lifetime of a supercomputing machine, and investigating the use of a variety of training datasets. Our primary findings are:

- A machine learning approach trained on data from an older computing system can out-perform fault mitigation performed by an expert-designed system in terms of percentage of faults successfully mitigated while maintaining the same percentage of wasted healthy pages and remaining bad pages.
- DRAM behavior with regard to faults does not seem to vary significantly over time, even when considering the full lifetime of a machine.

## II. Available Field Data

Cielo was a leadership-class production system at LANL in Los Alamos, New Mexico. Cielo contained 8,500 compute nodes, each with two 8-core AMD Opteron$^{TM}$ processors. Each processor had eight 32KB L1 data caches, eight 512KB L2 caches, and one 12MB L3 cache. Each node contained a total of 32GB of DRAM, from eight 4GB DDR-3 registered DIMMs. In this work, we focus on correctly classifying the DRAM fault modes seen over the lifetime of Cielo. In order to do this, we also take into account the DRAM fault modes seen on Hopper, an older supercomputer located at LBNL in Oakland, California, which had a similar configuration, although with 2,500 fewer compute nodes.

Figure 1 shows the distribution of the fault modes we consider, cumulatively and by individual years over the lifetime of Cielo. We note that the most likely fault mode, regardless of year, is a Single Bit fault. Interestingly, there does not appear to be any clear trend over time regarding composition of the fault data. Similarly, Figure 2 illustrates the percentage of errors in the Cielo dataset which were generated in each year of the lifetime. Cielo was de-commissioned in early 2016, resulting in a very small contribution from its final year.

[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-17-22170 and the Sandia identifier SAND2017-3983C. Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. 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.

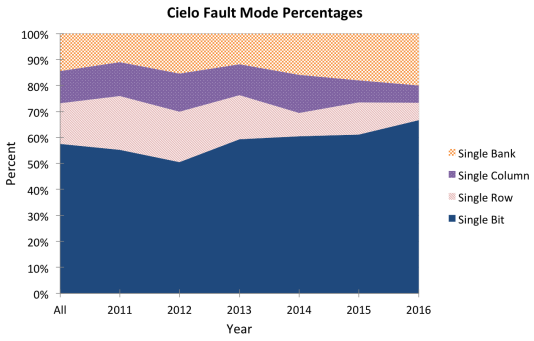



Fig. 1. DRAM fault mode compositions over the lifetime of Cielo.

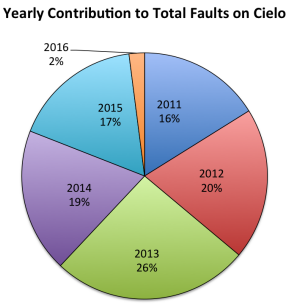



Fig. 2. Yearly contribution to total lifetime DRAM faults on Cielo.

However, we see that all other years contributed a very similar amount of faults, suggesting that there was no significant difference in DRAM behavior as the machine aged.

## III. Current Method

We assume, as in previous work [5], that all DRAM devices experience at most one fault mode over their lifetime. Definitions of the most common fault modes on which we focus are as follows:

- *Single Bit Fault*: All errors from a single DRAM occur within the same row and column.
- *Single Row Fault*: All errors from a single DRAM occur within the same row, but across different columns.
- *Single Column Fault*: All errors from a single DRAM occur within the same column, but across different rows.
- *Single Bank Fault*: The errors from a single DRAM are spread across varying rows and columns.

Other more serious fault modes are extremely rare in the available field data and we do not consider these fault modes in the current work. The current state-of-the-art method for determining fault modes online is to simply apply these deterministic rules, which encode expert assumptions, at each point in time [7].

## IV. Proposed Method

To eliminate the need for costly expert-designed fault characterization systems, we propose the use of machine learning in order to predict fault mode, following previous work [5]. The previous study finds that a machine learning technique which gives excellent fault mode prediction performance after observing only two errors from an individual DRAM is a random forest (a randomized ensemble of decision trees [8]). We extend this work by applying the technique to a much larger dataset covering the lifetime of a supercomputing facility and by investigating the use of a variety of training sets.

The features included in the random forest include only information regarding the location of observed errors, along with timestamps (e.g., frequency of errors observed in the same row, number of errors observed in the same column, etc.). We again follow previous work by calculating these features after two errors have been observed, because this was the earliest point in time that any algorithm was able to achieve satisfactory predictive performance.

We then investigate the usefulness of three different training methods in comparison with the current rule-based method:

1) A random forest trained on error data from only the previous year of Cielo activity
2) A random forest trained on error data from all previous Cielo activity
3) A random forest trained using error data from the entire lifetime of Hopper

Results will provide insight regarding whether or not data from older systems can be used for intelligent fault mitigation in real-time, and whether only very recent data provides enough information to the learning algorithm.

## V. Experimental Setup

For each training method described above, we test the learned model on all of the available Cielo data (except for that used for training), as well as on yearly subsets of the Cielo data. Each random forest model, while trained on a different dataset, consisted of 200 trees with a maximum depth of 6. Each model-test set pair was run 200 times to achieve a sense of variance. We report averages and standard deviations for each of our metrics, described below. Note that the rule-based system is deterministic and therefore did not require repetition.

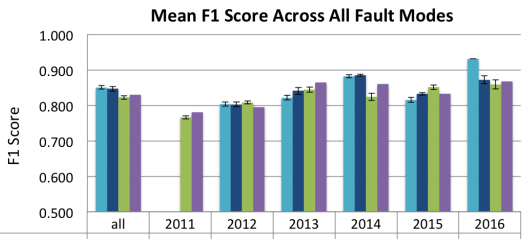
### A. Evaluation

We present four evaluation metrics, investigating separate but related aspects of fault mitigation.

**F1 Score:** F1 Score is a machine learning metric for evaluating predictive performance, ranging from 0.0 (predictive performance worse than random guessing) to 1.0 (perfect predictive performance) [9]. It uses the following relationship between precision and recall:

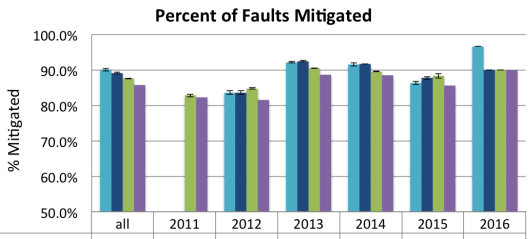
$$f1 = 2 \times \frac{\texttt{precision} \times \texttt{recall}}{\texttt{precision} + \texttt{recall}}$$

**Percent Mitigated:** Even in cases where a fault mode is mis-predicted, the prediction may still result in a successful mitigation. For example, if a fault is truly a single bit fault, but the prediction is for a single row fault, the result would be blacklisting the entire row, which would include the corrupted




| | all | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 |
|---|---|---|---|---|---|---|---|
| Random Forest, trained on previous Cielo year | 0.851 | | 0.804 | 0.822 | 0.882 | 0.816 | 0.931 |
| Random Forest, trained on all previous Cielo | 0.847 | | 0.803 | 0.841 | 0.886 | 0.832 | 0.872 |
| Random Forest, trained on Hopper | 0.822 | 0.766 | 0.808 | 0.845 | 0.824 | 0.851 | 0.860 |
| Rule-Based | 0.831 | 0.781 | 0.796 | 0.864 | 0.860 | 0.832 | 0.867 |

Fig. 3. F1 score (predictive performance) for each method.




| | all | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 |
|---|---|---|---|---|---|---|---|
| Random Forest, trained on previous Cielo year | 90.1% | | 83.6% | 92.2% | 91.5% | 86.4% | 96.7% |
| Random Forest, trained on all previous Cielo | 89.1% | | 83.6% | 92.5% | 91.7% | 87.7% | 90.0% |
| Random Forest, trained on Hopper | 87.6% | 82.8% | 84.8% | 90.5% | 89.6% | 88.3% | 90.0% |
| Rule-Based | 85.7% | 82.3% | 81.6% | 88.7% | 88.5% | 85.6% | 90.0% |

Fig. 4. Percent of faults successfully mitigated by each method.

single bit. To address this, we report the percent of faults that are successfully mitigated by the prediction method, regardless of the correctness of the prediction.

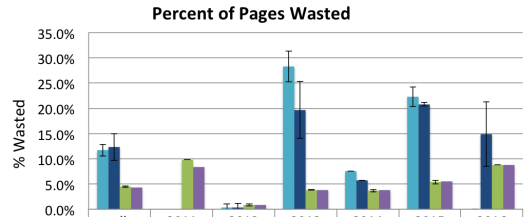
**Percent Wasted:** When an algorithm mis-predicts a fault mode, it may incorrectly blacklist pages that are in fact still healthy. We refer to these incorrectly blacklisted pages as waste, and report the percentage of pages blacklisted by the algorithm that were actually healthy.

**Percent Remaining:** When an algorithm mis-predicts a fault mode, it may blacklist fewer pages than were actually required to successfully mitigate the fault. In this case, some portion of the affected pages remain in use by the system. We report the percentage of unhealthy pages that remain in use by the system for each prediction method.

Since the mitigation strategy is based on the number of pages in memory affected by the particular fault mode, we make some generic assumptions regarding the dimensions of the DRAM (which in practice would vary based on the particular hardware specifications). We assume that a row is a single page wide while a column fault affects on the order of 10 pages [10]. Therefore, mitigation costs for each fault mode are 1 page for single bit or single row faults, and 10 pages for single column or single bank faults.

## VI. Results

Figure 3 shows mean (weighted) F1 scores across all fault modes for the random forest approach and the rule-based approach, by year. A higher F1 score indicates better general predictive performance. On average, the random forest models trained on previous Cielo data tend to give slightly better predictive performance than models learned on Hopper data or the rule-based system. However, considering performance over time shows no clear advantage to any of the learned models, although we do see that in general the learned models tend to out-perform the rule-based system. All the learned models, regardless of training set, tend to perform at least as well as the rule-based system, indicating that we may be able to save human architect effort by relying on models learned from previous systems. However, while the F1 metric provides information regarding accuracy of specific

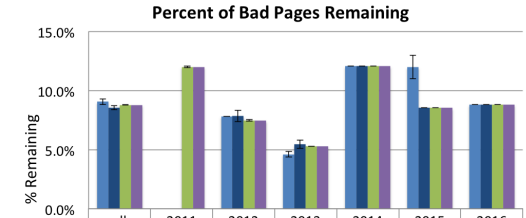



| | all | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 |
|---|---|---|---|---|---|---|---|
| Random Forest, trained on previous Cielo year | 11.7% | | 0.3% | 28.3% | 7.6% | 22.3% | 0.0% |
| Random Forest, trained on all previous Cielo | 12.3% | | 0.4% | 19.7% | 5.7% | 20.8% | 14.9% |
| Random Forest, trained on Hopper | 4.5% | 9.8% | 0.9% | 3.8% | 3.7% | 5.4% | 8.8% |
| Rule-Based | 4.3% | 8.4% | 0.9% | 3.8% | 3.8% | 5.5% | 8.8% |

Fig. 5. Percent of healthy pages wasted by each method.

predictions, it may not be the most meaningful in terms of real-time fault mitigation in production.

Figure 4 shows the percent of faults, from all modes, successfully mitigated by the rule-based and learned approaches. All learned approaches consistently mitigate more faults than the rule-based system, except in the final year which contained significantly less data (recall Figure 2). On average across all years, we find that random forests trained on only data from




| | all | 2011 | 2012 | 2013 | 2014 | 2015 | 2016 |
|---|---|---|---|---|---|---|---|
| Random Forest, trained on previous Cielo year | 9.1% | | 7.8% | 4.6% | 12.1% | 12.0% | 8.8% |
| Random Forest, trained on all previous Cielo | 8.6% | | 7.9% | 5.5% | 12.1% | 8.5% | 8.8% |
| Random Forest, trained on Hopper | 8.8% | 12.0% | 7.5% | 5.3% | 12.1% | 8.5% | 8.8% |
| Rule-Based | 8.8% | 12.0% | 7.5% | 5.3% | 12.1% | 8.5% | 8.8% |

Fig. 6. Percent of bad pages remaining for each method.

the previous year of Cielo data mitigate the greatest percentage of faults, followed by models trained on all previous Cielo data, and finally followed by models trained on all available Hopper data. All three of these approaches successfully mitigate more faults than the expert-designed rule-based system, even when taking into consideration the standard deviation of the model outputs.

Figure 5 presents the percentage of pages blacklisted by each approach that were in fact healthy (wasted pages). Interestingly, yearly and on average, random forests trained on data from Cielo (whether only the previous year or all available previous data), consistently waste significantly more pages of memory than models trained on Hopper or the rule-based system. Overall, random forest models trained on Hopper data appear to waste around the same amount of memory as the rule-based system. While the cause of this behavior is unclear, it seems that using a larger dataset from a previous supercomputing system causes the models to be more conservative in their blacklisting of pages, while using a smaller dataset, although it is taken from the current system, tends to wildly over-estimate the number of blacklisted pages required to successfully mitigate faults. Recall that all three learned models out-performed the rule-based system in terms of mitigated faults, so the over-estimation of blacklisted pages is not necessarily helping the models learned on Cielo to successfully mitigate more faults than those learned from Hopper. The exception to this behavior is the performance of models trained on only the previous year of Cielo data, which in 2016 do not blacklist any healthy pages. This may be due to the significantly smaller size of the dataset for 2016.

Figure 6 shows the percentage of bad pages still remaining in use after each methods has attempted to mitigate faults. Surprisingly, all the learned methods perform very similarly to the rule-based system, tending to miss very nearly the same number of bad pages. The only exceptions to this behavior are in the random forests trained on Cielo data from only the previous year in 2013 and 2015, with 2013 showing a slight improvement over the other models, and 2015 showing an extremely inferior performance.

Considering our observation that all models perform similarly regarding the number of remaining bad pages in the system, we are free to evaluate the approaches using only three of our metrics: F1 score, percent of faults mitigated, and percent of healthy pages wasted.

Although we do observe some variability in all our metrics over time, interestingly there does not appear to be any clear time-dependent trends. This implies that in addition to models trained on an older system being sufficiently accurate for application to a new system, the learned model may only need to be re-trained extremely infrequently, or only in cases where hardware that is significantly different enough to warrant new features is used.

## VII. Conclusion

We find that machine learning can out-perform the state-of-the-art expert-designed method for DRAM fault mode classification on metrics that consider production-relevant measurements. While all the learned models tended to leave the same percentage of bad pages remaining in memory as the expert system, they successfully mitigated a significantly greater percentage of faults. In addition, models trained on data from Cielo, the machine on which they were intended to be used, tended to waste many more healthy pages than models trained on an older system. While the cause of this behavior is not clear, it appears that training on a larger dataset from an older system may cause the model output to have better performance than training on a newer but much smaller dataset. We also find no clear trends over time in any of the evaluation metrics, suggesting that there was no major change in DRAM behavior across the lifetime of Cielo. We recommend further study on the use of machine intelligence for handling hardware faults, including implementing the strategy on a current system in order to obtain performance metrics.

## Acknowledgments

The authors thank Sean Blanchard, at LANL and the Ultrascale System Research Center, for domain expert input. The authors also 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.

## References

[1] K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, K. Hill, J. Hiller *et al.*, "Exascale computing study: Technology challenges in achieving exascale systems," *Defense Advanced Research Projects Agency Information Processing Techniques Office (DARPA IPTO), Tech. Rep*, vol. 15, 2008.

[2] M. Gottscho, M. Shoaib, S. Govindan, B. Sharma, D. Wang, and P. Gupta, "Measuring the impact of memory errors on application performance," *IEEE Computer Architecture Letters*, 2016.

[3] M. Gamell, D. S. Katz, H. Kolla, J. Chen, S. Klasky, and M. Parashar, "Exploring automatic, online failure recovery for scientific applications at extreme scales," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Press, 2014, pp. 895–906.

[4] D. Tang, P. Carruthers, Z. Totari, and M. W. Shapiro, "Assessment of the effect of memory page retirement on system RAS against hardware faults," in *Dependable Systems and Networks, 2006. DSN 2006. International Conference on*. IEEE, 2006, pp. 365–370.

[5] 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 *Dependable Systems and Networks Workshop, 2016 46th Annual IEEE/IFIP International Conference on*. IEEE, 2016, pp. 250–253.

[6] B. Schroeder, E. Pinheiro, and W.-D. Weber, "DRAM errors in the wild: a large-scale field study," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 37, no. 1. ACM, 2009, pp. 193–204.

[7] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi, "Feng shui of supercomputer memory positional effects in DRAM and SRAM faults," in *High Performance Computing, Networking, Storage and Analysis (SC), 2013 International Conference for*. IEEE, 2013, pp. 1–11.

[8] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine learning*, vol. 40, no. 2, pp. 139–157, 2000.

[9] D. M. Powers, "Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation," 2011.

[10] 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.