COLLEGE OF SCIENCE
THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

Research and Creative Experience for Undergraduates Program (RCEU) Summer 2022

Algorithm-Based Fault Tolerance at Scale

Hayden Estes, Dr. Joshua Dennis Booth
Department of Computer Science

Overview

The rising need for fault tolerant systems is higher than ever due to the introduction of exascale computing. A system's fault tolerance is defined by its hardware, algorithms, and data types. Modern heterogeneous systems are composed of a large number of processors that support a variety of these data types. Understanding these data types' roles within an algorithm is vital to measuring fault tolerance. Exploring various floating-point formats shows how they can impact fault tolerance methods used for the sparse conjugate gradient algorithm (CG). We have determined that the modern Brain Floating Point Format (BFloat16) used in many Google tensor processing units (TPUs) can not be applied successfully, while IEEE32 is utilized and optimized in GPUs (e.g. NVIDIA Tensor Core) can be applied successfully. These findings will make long-running scientific codes that use CG as a solver method able to ensure accuracy with minimal increased run time.

Pictured: Frontier, the world's first exascale supercomputer.
Frontier will experience several faults every day due to its size and complexity,

Background

1. **What are faults?**
   A fault is when a computer performs an operation incorrectly for various reasons.
   a. Hard-faults: Replicable physical faults in computer hardware.
   b. Soft-fault: transitive temporary faults. (e.g., alpha particle, packing pollution, cosmic radiation, etc)
2. **How frequent are faults?**
   The frequency of faults is directly correlated to the complexity of the system.
   a. Faults on an average computer are reasonably rare due to low complexity.
   b. High-performance systems have several millions of processors which heavily increases the frequency of faults.
3. **How can a system be more fault tolerant?**
   Faults are tolerated through use of detection, duplication, and correction.
   A. Detection involves checking the results in real time based on algorithms or generalized methods regardless of the algorithm. (e.g. duplicating calculations)
   B. Correction involves accepting the majority vote, continuing from a checkpoint, etc.

Methodology

Continuing the work of our primary reference, we decided to explore how differing levels of precision may affect the efficiency of fault tolerance. We implemented our own preconditioned CG algorithm so that we had full control over the precision during every mathematical operation. Using this implementation, we supplied a set of SuiteSparse matrices and applied the data types IEEE64, IEEE32, and BFloat16. After confirming our data was correct through duplication and comparing the vectors of the previous work, we were able to analyze our results for any data types' benefits.

Results

Ultimately, we determined that the CG iterations, memory overhead, and overall precision are the key aspects of these data types. Figures 1 and 2 below show characterizing results of our preconditioned CG algorithm.

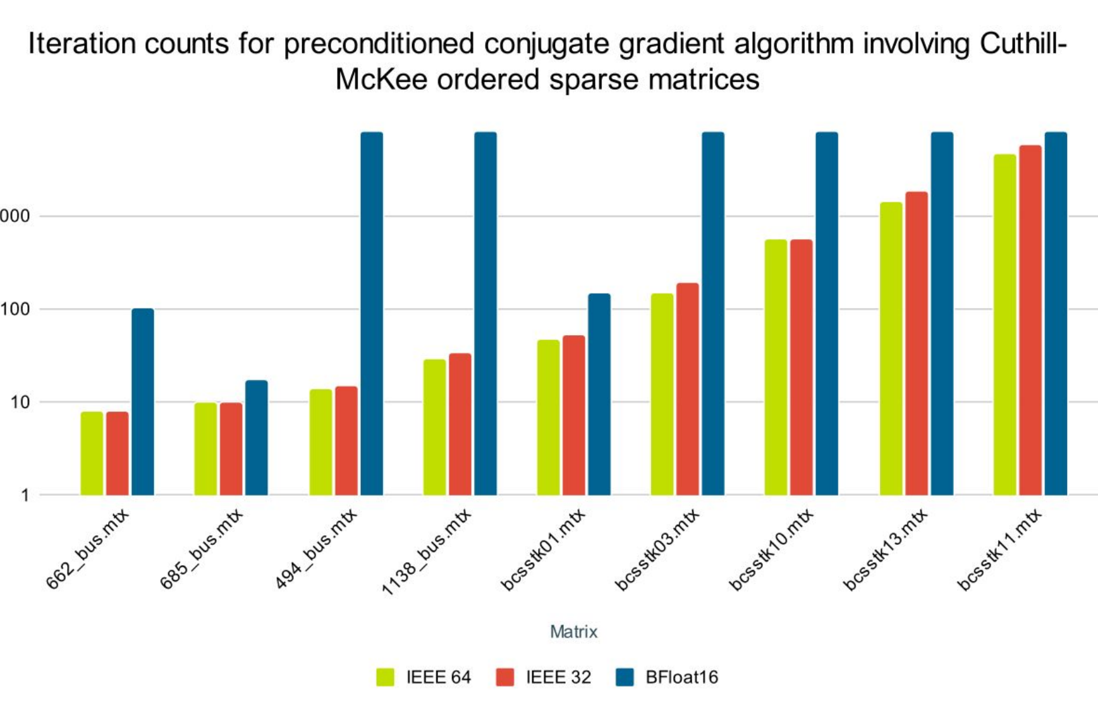
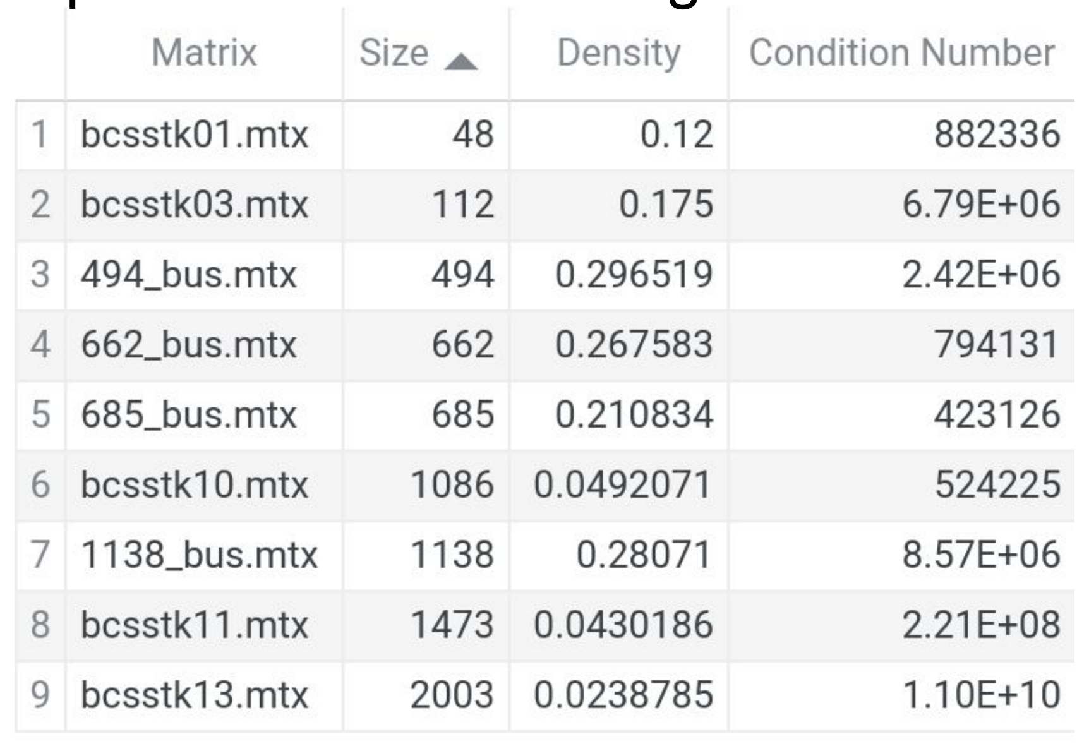


Figure 2: PCG iteration counts with varying precisions
*Note: The iterations have an artificial maximum of 8000



	Matrix	Size ▲	Density	Condition Number
1	bcsstk01.mtx	48	0.12	882336
2	bcsstk03.mtx	112	0.175	6.79E+06
3	494_bus.mtx	494	0.296519	2.42E+06
4	662_bus.mtx	662	0.267583	794131
5	685_bus.mtx	685	0.210834	423126
6	bcsstk10.mtx	1086	0.0492071	524225
7	1138_bus.mtx	1138	0.28071	8.57E+06
8	bcsstk11.mtx	1473	0.0430186	2.21E+08
9	bcsstk13.mtx	2003	0.0238785	1.10E+10

Figure 1: Sparse Matrices with their respective size, density, and condition values

IEEE 32	vs	BFloat16
➔ More precise decimals		➔ Less precise decimals
➔ Large iteration decrease		➔ Large iteration increase
➔ More memory overhead		➔ Less memory overhead

IEEE 32	vs	IEEE64
➔ Less precise data type		➔ Less precise data type
➔ Minimal iteration increase		➔ Large iteration increase
➔ Less memory overhead		➔ Less memory overhead

1. IEEE32 precision is a beneficial alternative to IEEE64 for CG algorithms while still causing a minimal increase in iteration count due to the efficiency increase in fault tolerance methods.
2. BFloat16 is not a viable alternative to IEEE64 or IEEE32 for CG algorithms due to the large increase in iteration counts compared to the fault tolerance efficiency

Impact

Our findings will make CG fault tolerance more feasible due to the higher duplication efficiency with the IEEE32 data type. CG algorithms are used to solve sparse systems of linear equations, so they are included in many other applications such those represented by partial differential equations. This research does not only affect high-performance computing but also all long-running programs including CG algorithms such as climate, electrical circuit, and robotic manipulator simulations.

Key References

1. Manu Shantharam, Sowmyalatha Srinivasmurthy, and Padma Raghavan. 2012. Fault tolerant preconditioned conjugate gradient for sparse linear system solution. In Proceedings of the 26th ACM international conference on Supercomputing (ICS '12). Association for Computing Machinery, New York, NY, USA, 69–78.
2. Berrocal, Eduardo et al. "Lightweight Silent Data Corruption Detection Based on Runtime Data Analysis for HPC Applications." *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*. ACM, 2015. 275–278. Web.
3. Cappello, Franck et al. "Toward Exascale Resilience." The international journal of high performance computing applications 23.4 (2009): 374–388. Web.
4. Sun, Hongyang et al. "Selective Protection for Sparse Iterative Solvers to Reduce the Resilience Overhead." 2020 IEEE 32nd International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). IEEE, 2020. 141–148. Web.

Acknowledgements

Thank you to the *National Science Foundation* for funding this project through NSF Grant 2135310, the *Extreme Science and Engineering Discovery Environment* for access to high performance hardware, and UAH for the *Research and Creative Experience for Undergraduates* program and opportunity.

XSEDE
Extreme Science and Engineering Discovery Environment