Algorithm-Based Fault Tolerance at Scale

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

Introduction

- ➔ The rising need for fault tolerant systems is higher than ever due to the introduction of exascale computing.

- ➔ This research explores how different floating-point formats can impact the efficiency for fault tolerance for conjugate gradient algorithms.

- ➔ These findings will make long-running scientific codes that use CG as a solver method able to ensure accuracy with minimal increased run time.

What are faults?

- A fault is when a computer performs an operation incorrectly for various reasons.

- There are hard-faults and soft-faults
  - Hard-faults are physical faults in the computer's hardware that cause unexpected behavior and are often replicable.
  - Soft-faults are created for numerous external reasons, including alpha particles, packing pollution, and cosmic radiation.

How frequent are faults?

- ➔ On an average computer, faults are reasonably rare.
- ➔ However, High-performance systems can have numerous hard-faults and soft-faults happen every 24 hours.

(https://www.psc.edu/bridges-2-begins-production-operations)

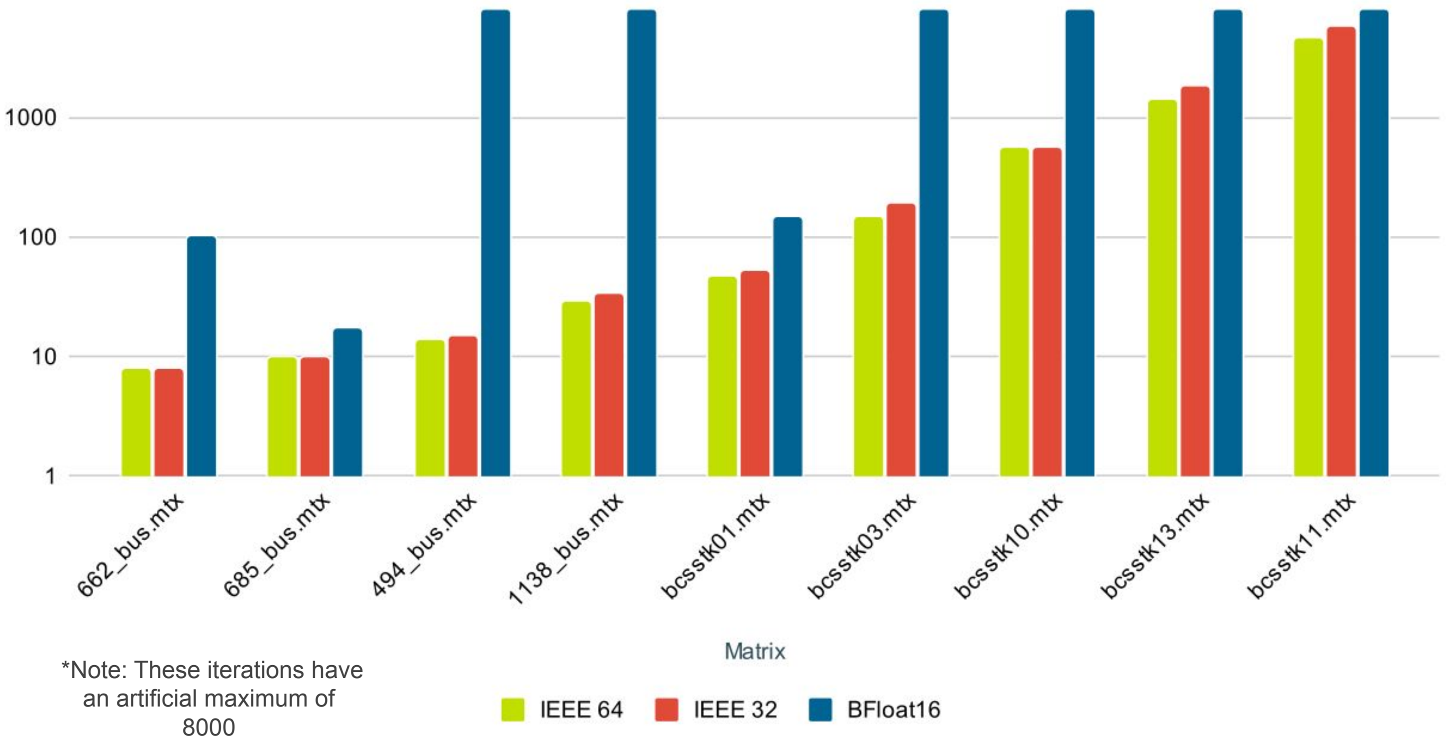
How are faults tolerated?

- Faults are detected by verifying calculated values in real-time.
- Some programs use specialized algorithms for their specific problems or generalized verification methods such as duplication.
- We target an algorithm-based fault tolerance method used for the sparse conjugate gradient algorithm (CG).

Our process

- ➔ We implemented our own preconditioned conjugate gradient algorithm
  - ◆ This allows us to keep the precision of our algorithm for every mathematical operation
- ➔ Using this implementation, we supplied 20 unique sparse matrices and applied the algorithm in each of the separate data types (including mixed precision)
- ➔ After confirming our data was correct, we were able to analyze for any of the various data types available benefits

Iteration counts for preconditioned conjugate gradient algorithm involving Cuthill-McKee ordered sparse matrices



*Note: These iterations have an artificial maximum of 8000

IEEE32 vs BFloat16

- More precise data type
- Large PCG iteration decrease
- More memory overhead

- Less precise data type
- Large PCG iteration increase
- Less memory overhead

IEEE32 vs IEEE64

- Less precise
- Minimal PCG iteration increase
- Less memory overhead

- More precise data type
- Minimal PCG iteration decrease
- More memory overhead

Our Conclusions

1. The IEEE32 precision is a beneficial alternative to IEEE64 for PCG algorithms while still causing a minimal increase in iteration count due to the efficiency increase in fault tolerance methods.

2. The Google Brain Float is not a viable alternative to IEEE64 or IEEE32 for PCG algorithms due to the large increase in iteration counts compared to the fault tolerance efficiency

Impact

1. Our findings will make CG fault tolerance more feasible due to the higher duplication efficiency with the IEEE32 data type.

2. This research does not only affect high-performance computing but also all areas that use long-running programs with CG algorithms.

3. CG algorithms are not only used in high-performance computing, they are also used to solve sparse systems of linear equations and included in many other applications such those represented by partial differential equations.

Key References

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.

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.

Cappello, Franck et al. “Toward Exascale Resilience.” The international journal of high performance computing applications 23.4 (2009): 374–388. Web.

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.

Thank you to…

1. The National Science Foundation for granting us our funding

2. The Extreme Science and Engineering Discovery Environment

3. UAH for the RCEU Program and opportunity