Compression of Linear Algebra Data Structures to Reduce Memory Access Latencies

Thesis Proposal

College of Saint Benedict/Saint John's University

Neil Lindquist

October 2018

Compression of Linear Algebra Data Structures to Reduce Memory Access Latencies

Neil Lindquist

Approved By:

Mike Heroux
Thesis Advisor
Scientist in Residence

Robert Hesse
Faculty Reader
Associate Professor of Mathematics

Jeremy Iverson
Faculty Reader
Assistant Professor of Computer Science

Imad Rahal
Chair, Department of Computer Science

Bret Benesh
Chair, Deparment of Mathematics

1 Statement of Purpose

The goal of this project is to improve the performance of certain parts used in some scientific computations. These calculations include problems such as modeling fluid flow, chemical processes and electromagnetism. The specific parts being optimized are iterative solvers for sparse, linear equations. The majority of the time spent in these types of solvers is spent moving data from main memory to the processor where the actual arithmetic is done [21]. So, by compressing the largest pieces of this data, this project will try to reduce the time spent waiting and thus improve the overall efficiency. This improvement can allow running important computations more often and with more detail.

As a more detailed description of the computations being improved, a basic description of this type of linear solver follows. The specific problem this work focuses on is solving systems of linear equations. A simple example of a system of linear equations is finding values for x and y such that

$$5x + 3y = 10$$
$$3x + 3y = 6$$

are both true. Some of the linear system used in real-work problems, such as representing a set of physics equations, can become very large, with possibly hundreds of millions of equations [7]. Not only does this large size result in long times to solve the problem, it means that the problem has to be stored in the slow-to-access main memory instead of the quick-to-access caches. Accessing main memory can take over 100 times as long as a single arithmetic operation [14]. So, reductions in the amount of memory used to store key components of the problem have the opportunity to provide significant improvements in performance, even if it increases the actual calculations needed. Specifically, data compression schemes with very simple decoding algorithms are being experimented with to find the most effective way to store the key data structures of the solvers.

2 Proposal Summary

This project is working at finding compression methods that improve performance by reducing memory access latencies. The High Performance Conjugate Gradient (HPCG) benchmark is being used as the codebase to test compression methods with [10]. This code is modified to utilize different compression methods, then is used to test the compression's performance. Because HPCG is a benchmark that uses a sparse, iterative linear solver to test performance, it provides both an implementation of a sparse, iterative linear solver (specificly Conjugate Gradient) and measures of performance for both the overall solver as well as the major components. Compression techniques being used include single precision floats, SZ compression and ZFP compression for floating point data and Elias Gamma coding, Elias Delta coding, SZ compression, Huffman coding for integers [9, 23, 11, 17].

3 Preliminary Outline

At the highest level, the thesis will have a vary simple outline. First, it will provide enough information to understand the goals, test problem, compression methods used and the interactions there of. Next will be the actual test results and analysis there of. Finally, the paper will present any conclusions discovered in the test results as well as ways the work can be extended in the future. The first section will cover a number of relevant topics. Firstly, the relevance and goal of this work will be presented. Next, the current state of the art for this type of computations will be described, including both the computational aspects and the underlying mathematics. After that will be a description of the code base used to create the test results. Then, a short discussion of restrictions on the compression and data access patterns will be provided. Finally, information on the various compression methods will be provided. The other two overall sections will be simpler. The test results will largely contain tables of times and other metrics, but will also contain analysis of the results. The concluding section will summarize the results and conclusions of the work as well as propose future projects that builds on these results.

4 Preliminary References

[1] Hartwig Anzt, Björn Rocker, and Vincent Heuveline. Energy efficiency of mixed precision iterative refinement methods using hybrid hardware platforms. *Computer Science - Research and Development*, 25(3):141–148, 2010.

[2] Marc Baboulin, Alfredo Buttari, Jack Dongarra, Jakub Kurzak, Julie Langou, Julien Langou, Piotr Luszczek, and Stanimire Tomov. Accelerating scientific computations with mixed precision algorithms. *CoRR*, abs/0808.2794, 2008.

[3] Alfredo Buttari, Jack Dongarra, Jakub Kurzak, Piotr Luszczek, and Stanimir Tomov. Using mixed precision for sparse matrix computations to enhance the performance while achieving 64-bit accuracy. *ACM Trans. Math. Softw.*, 34(4):17:1–17:22, July 2008.

[4] Alfredo Buttari, Jack Dongarra, Julie Langou, Julien Langou, Piotr Luszczek, and Jakub Kurzak. Exploiting the performance of 32 bit floating point arithmetic in obtaining 64 bit accuracy (revisiting iterative refinement for linear systems). In *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, SC '06, New York, NY, USA, 2006. ACM.

[5] Alfredo Buttari, Jack Dongarra, Julie Langou, Julien Langou, Piotr Luszczek, and Jakub Kurzak. Mixed precision iterative refinement techniques for the solution of dense linear systems. *Int. J. High Perform. Comput. Appl.*, 21(4):457–466, November 2007.

[6] Wei-Fan Chiang, Mark Baranowski, Ian Briggs, Alexey Solovyev, Ganesh Gopalakrishnan, and Zvonimir Rakamarić. Rigorous floating-point mixed-precision tuning. *SIGPLAN Not.*, 52(1):300–315, January 2017.

[7] Timothy A. Davis and Yifan Hu. The university of florida sparse matrix collection. *ACM Trans. Math. Softw.*, 38(1):1:1–1:25, December 2011.

[8] James Demmel, Yozo Hida, William Kahan, Xiaoye S. Li, Sonil Mukherjee, and E. Jason Riedy. Error bounds from extra-precise iterative refinement. *ACM Trans. Math. Softw.*, 32(2):325–351, June 2006.

[9] S. Di and F. Cappello. Fast error-bounded lossy hpc data compression with sz. In *2016 IEEE International Parallel and Distributed Processing Symposium*, pages 730–739, May 2016.

[10] Jack Dongarra, Michael Heroux, and Piotr Luszczek. Hpcg benchmark: a new metric for ranking high performance computing systems. Technical Report UT-EECS-15-736, Electrical Engineering and Computer Sciente Department, Knoxville, Tennessee, November 2015.

[11] P. Elias. Universal codeword sets and representations of the integers. *IEEE Transactions on Information Theory*, 21(2):194–203, March 1975.

[12] Samuel A. Figueroa. When is double rounding innocuous? *SIGNUM Newsl.*, 30(3):21–26, July 1995.

[13] Dominik Göddeke, Robert Strzodka, and Stefan Turek. Performance and accuracy of hardware-oriented native-, emulated-and mixed-precision solvers in fem simulations. *Int. J. Parallel Emerg. Distrib. Syst.*, 22(4):221–256, January 2007.

[14] Georgios Goumas, Kornilios Kourtis, Nikos Anastopoulos, Vasileios Karakasis, and Nectarios Koziris. Performance evaluation of the sparse matrix-vector multiplication on modern architectures. *The Journal of Supercomputing*, 50(1):36–77, Oct 2009.

[15] Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1996.

[16] J. D. Hogg and J. A. Scott. A fast and robust mixed-precision solver for the solution of sparse symmetric linear systems. *ACM Trans. Math. Softw.*, 37(2):17:1–17:24, April 2010.

[17] David A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the Institute of Radio Engineers*, 40(9):1098–1101, September 1952.

[18] Claude-Pierre Jeannerod and Siegfried M. Rump. Improved error bounds for inner products in floating-point arithmetic. *SIAM Journal on Matrix Analysis and Applications*, 34(2):338–344, 2013.

[19] John Jenkins, Eric R. Schendel, Sriram Lakshminarasimhan, David A. Boyuka, II, Terry Rogers, Stephane Ethier, Robert Ross, Scott Klasky, and Nagiza F. Samatova. Byte-precision level of detail processing for variable precision analytics. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 48:1–48:11, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[20] Toyohisa Kaneko and Bede Liu. On local roundoff errors in floating-point arithmetic. *J. ACM*, 20(3):391–398, July 1973.

[21] Orion Sky Lawlor. In-memory data compression for sparse matrices. In *Proceedings of the 3rd Workshop on Irregular Applications: Architectures and Algorithms*, IA3 '13, pages 6:1–6:6, New York, NY, USA, 2013. ACM.

[22] Xiaoye S. Li, James W. Demmel, David H. Bailey, Greg Henry, Yozo Hida, Jimmy Iskandar, William Kahan, Suh Y. Kang, Anil Kapur, Michael C. Martin, Brandon J. Thompson, Teresa Tung, and Daniel J. Yoo. Design, implementation and testing of extended and mixed precision blas. *ACM Trans. Math. Softw.*, 28(2):152–205, June 2002.

[23] P. Lindstrom. Fixed-rate compressed floating-point arrays. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2674–2683, Dec 2014.

[24] Y. Saad. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd edition, 2003.

[25] Tran-Thong and Bede Liu. Floating point fast fourier transform computation using double precision floating point accumulators. *ACM Trans. Math. Softw.*, 3(1):54–59, March 1977.

[26] Eiji Tsuchida and Yoong-Kee Choe. Iterative diagonalization of symmetric matrices in mixed precision and its application to electronic structure calculations. *Computer Physics Communications*, 183(4):980–985, apr 2012.