CS575 - Final Project

Damian Franco

Plan and Goals

Numerical computation has two very important aspects, accuracy and performance. My plan for the project will be to focus on the performance side of numerical computation while maintaining accuracy and stability for the algorithms that I implement. To improve performance, I will use parallel processing techniques. Parallel processing is a mode of computer operation in which a process is split into parts that execute simultaneously on different processors attached to the same computer. This enables multiple processes or threads to compute multiple operations at the same time. Many matrix, vector and linear system operations can be performed in parallel which can majorly improve the computation time of an algorithm. Although parallel processing is known to improve performance time, an incorrect one could massively decrease performance time. My plan is to correctly implement some algorithms that we have covered throughout the course by starting with algorithms as simple as matrix-matrix operations to more advanced linear system operations. I will also implement non-parallel versions of each algorithm and compare and contrast accuracy and performance. My goal is to demonstrate how effective parallel computing is when performing numerical computational tasks and to understand how scaling factors into performance of implementations.

Implementation

My implementation will be in the programming language C/C++. This low-level language will allow me to fully utilize parallel processing techniques alongside MPI. Message Passing Interface or MPI is a standardized and portable message-passing standard designed to function on parallel computing architectures. MPI has parallel processing functions that I will utilize for simply parallel message processing and no computation. All implementations will use no external libraries besides MPI functions in their respective implementations. I will gather results from various implementations of parallel and non-parallel implementations of numerical algorithms. The algorithms that I will initially cover will be the following:

- *Matrix-matrix multiplication*
- *Solving a linear system using Gaussian Elimination*
- *LU Decomposition with Doolittles Algorithm*
- *Calculating eigenvalue and eigenvector with using Jacobi Method*

The topics above are the initial algorithms that I assumed would be a wide range of topics covered in class and some that we did not cover but are useful. My input will be dense matrices that will be scaled from smaller 10x10 matrices to larger 100x100 matrices. Depending on the algorithm, I will change scaling factors. For each approach I will attempt to implement all the methods above and explain each of them. Some approaches I will explain at high-level if we covered in the course, and if we have not covered it, I will attempt to explain at a lower-level. Overall, I want my implementation to be fairly simple to understand and to function properly.

References

Hoefler, T., Gropp, W., Thakur, R., Träff, J.L. (2010). Toward Performance Models of MPI Implementations for Understanding Application Scaling Issues. In: Keller, R., Gabriel, E., Resch, M., Dongarra, J. (eds) Recent Advances in the Message Passing Interface. EuroMPI 2010. Lecture Notes in Computer Science, vol 6305. Springer, Berlin, Heidelberg. https://www.mcs.anl.gov/papers/P1758.pdf

Zhang, J. & Maple, Carsten. (2002). Parallel solutions of large dense linear systems using MPI. 312 - 317. 10.1109/PCEE.2002.1115280. https://www.researchgate.net/publication/3981881_Parallel_solutions_of_large_dense_linear_systems_using_MPI

MARTIN D. SCHATZ ROBERT A. VAN DE GEIJN,J. POULSON, Parallel Matrix Multiplication, A Systematic Journey, Department of Computer Science, Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX https://www.cs.utexas.edu/~flame/pubs/SUMMA2d3dTOMS.pdf

Buttari, A., Langou, J., Kurzak, J. and Dongarra, J. (2008), Parallel tiled QR factorization for multicore architectures. Concurrency Computat.: Pract. Exper., 20: 1573-1590. https://onlinelibrary.wiley.com/doi/10.1002/cpe.1301

Karniadakis, G., & Kirby II, R. (2003). Roots and Integrals. In Parallel Scientific Computing in C and MPI: A Seamless Approach to Parallel Algorithms and their Implementation (pp. 188-254). Cambridge: Cambridge University Press. doi:10.1017/CBO9780511812583.005 https://www.cambridge.org/core/books/abs/parallel-scientific-computing-in-c-and-mpi/roots-and-integrals/17DFA62F5FC1DCE2035A3C6853209A1D