CS 267 Applications of Parallel Computers
Spring 2022 Aydın Buluç and Jim Demmel

HW 0 Pre-proposal

My name is Brian Park and I am an undergraduate Computer Science major who will be graduating this semester. I also do research in the RISELab on the NumS project, an open source library that extends NumPy to distributed systems using Ray. It also provides distributed model training. My interests lie deeply in computer architecture, systems, and machine learning. I'm passionate about finding ways we can accelerate machine learning as well as make it efficient. What I want to get out of this class is learn and expand my knowledge of other parallel programming tools, such as CUDA, MPI, and UPC++. My upcoming internship will also be involved with accelerating machine learning primitives, so CS 267 will definitely help me prepare for that. If possible, I could also try to work on a project that would be relevant to my research in NumS, as we are trying to push a MPI backend to support scientific computing workloads on supercomputers like Cori.

For topics that I'm personally interested in and are related to research is parallelization of sparse array computations. As I've been familiar with dense array computations and performance, I would be interested in seeing how to parallelize and optimize sparse arrays, as many machine learning applications have led me to realize that sparse arrays are common, such as missing data, one hot encoding a dataset, and Laplacian matrices. Other common uses that I'm not yet familiar with involve neural networks, such as uses in graph neural networks and deep neural networks, commonly using sparse-dense matrix-matrix multiplication kernels. [1] Working with sparse arrays will be a new challenge for me, but I hope that it will be an interesting project for anyone interested in deep learning applications. We could analyze the `SpMM` or `SpMV` as a starting point. Of course, you could naively use dense matrix operations, but if most of your data is 0, you will waste computation power. I would be interested how they are sped up and how they perform with different kernels, analyzing how computations are load balanced, handled in memory, and what types of parallelization are involved. Analyzing how half precision could aid in performance would also be interesting, as a lot of machine learning applications and hardware use lower precision to achieve higher performance in FLOPS.

According to a research paper from our own faculty, one of the main bottlenecks seems to be communication, for sparse-dense matrix-matrix multiplication at least. [1] This is why GPUs are often the best use in application, as it has much more SIMD units than a CPU can offer in a shared memory setting. They used MPI as their communication backend and used IBM Summit system at Oak Ridge National Laboratory, which is in fact number 2 in the Top500 today. They only used CPU to aid in communication, and used nodes that were equipped with six NVIDIA Volta V100 GPUs. Because of the communication bottlenecks, it doesn't seem to scale well, hence why they used nodes with 6 GPUs.

Another issue with dealing with sparse array computation is managing load balancing. [2] We would also have to consider different ways of formatting a sparse matrix to make it appealing for SIMD units to efficiently operate on data. You don't want SIMD vector that operates over zero values, so finding a

format that can utilize both SIMD and MIMD, as well as overcoming the communication bottleneck is a plus. Another research paper performed their experiments on GPUs as well to analyze these affects on the Swiss National Supercomputing Centre, which is also a supercomputer 20th on the Top500 today. In general, sparse computations are notorious for achieving fractional peak performance, which the analysis shows. Interestingly enough, the number of non-zero elements vs. GFLOPS resemble a roofline model.

References

[1] O. Selvitopi, B. Brock, I. Nisa, A. Tripathy, K. Yelick, and A. Buluç, "Distributed-memory parallel algorithms for sparse times tall-skinny-dense matrix multiplication," in *Proceedings of the ACM International Conference on Supercomputing*, ICS '21, (New York, NY, USA), p. 431–442, Association for Computing Machinery, 2021.

[2] G. Flegar and H. Anzt, "Overcoming load imbalance for irregular sparse matrices," in *Proceedings of the Seventh Workshop on Irregular Applications: Architectures and Algorithms*, IA3'17, (New York, NY, USA), Association for Computing Machinery, 2017.