High Performance Computing: Homework 4

Dmitriy (Tim) Kunisky [dk3105]

Problem 1: Dot Product and Matrix-Vector Multiplication

I tested the matrix-vector multiplication with a square matrix and dimension $N = 4096$, and the inner product calculation with $N = 2^{25} \approx 3 \times 10^7$. The CPU implementation measurements are those run in the same executable on one of the CUDA machines, but not using the GPU in the code. This may explain why the memory bandwidth is so high for the CPU implementation of inner product—the test instance is very large, the testing machine has 40 CPUs supporting 2 threads per core via hyperthreading, and the implementation uses OpenMP parallelization.

Testing Machine	Mem. Band. (GB/s), Inner Product	Mem. Band. (GB/s), Matrix-Vector
CPU (on `cuda2`)	16.09	4.41
`cuda1`	18.24	23.90
`cuda2`	95.33	50.52
`cuda3`	170.62	56.78
`cuda4`	17.73	48.31
`cuda5`	11.52	16.94

Problem 2: Jacobi Iteration

I tested on an $N \times N$ grid with $N = 100$ and performing 10,000 Jacobi iterations to solve the Laplace equation over the unit square in two dimensions. The procedure appears to be slower on `cuda4` and `cuda5`, which is compatible with the results from the previous problem suggesting that these machines also had lower memory bandwidth.

Testing Machine	Time (s)
CPU (on `cuda2`)	9.45
`cuda1`	0.27
`cuda2`	0.16
`cuda3`	0.15
`cuda4`	0.72
`cuda5`	0.44

Problem 3: Project Proposal

I will be working by myself on implementing branch-and-bound methods for optimization using either linear or semidefinite programming as a primitive. If there is time, after getting a reasonable initial implementation, I will look to focus on questions related to load balancing of the branching sequences of problem instances that these techniques generate, and how this aspect can be optimized in an MPI implementation where communication among processes solving different instances is limited.