Advanced Topics in Numerical Analysis, High Performance Computing: Assignment 4

Mariya Savinov, mariyasavinov@nyu.edu

Repo location: `https://github.com/mariyasavinov/HPC_Spring_22.git`

1 Matrix-vector operations on a GPU

After successfully implementing a parallelized element-wise product between two-vectors (which could then be summed to achieve the inner product of the two vectors), I generalized this code to a matrix-vector multiplication (both on the CPU with OpenMP and on the GPU with CUDA). For the CPU part, the CPU parallel section is completed with the 32 threads on both of the following machines.

On `cuda2`, which has an `NVIDIA GeForce RTX 2080 Ti` GPU and `Intel Xeon E5-2660 v3` CPU (2.60 Hz):

```
CPU 0.066537 s
CPU Bandwidth: 6.051574 GB/s

GPU 0.029899 s, 0.003756 s
GPU Bandwidth: 13.467070 GB/s

Error = 0.000000
```

On `cuda3`, which has an `NVIDIA TITAN V` GPU and `Intel Zeon Gold 5118` CPU (2.30 Hz):

```
CPU 0.116728 s
CPU Bandwidth: 3.449496 GB/s

GPU 0.033467 s, 0.002017 s
GPU Bandwidth: 12.031516 GB/s

Error = 0.000000
```

2 2D Jacobi method on a GPU

3 Update on final project

Thus far, Paul Beckman and I have implemented and are in the process of debugging a serial version of the Barnes-Hut multipole algorithm built on a tree data structure. It has become throughout this process that it may be difficult to parallelize the tree-based implementation, due to its reliance on recursive function calls and pointer linkages. Our plans are to complete a basic OpenMP parallelization on the tree-based implementation, but then to experiment with developing an alternative array data structure implementation for more efficient parallelization.