**Just doing parts I can since im still working on getting it to run

Can the GPU give a boost to the parallelism in the Jacobi Iteration?

By: Michael Plekan

Abstract: The GPU is made up of a lot of simple cores, Jacobi is a problem with alot of simple math problems. We can get a decent boost in performance just by parallelizing on the CPU but we might be able to get an even better one on the GPU. Inorder to parallelize, OpenMP is being used, and with the right setup in the code and everything needed for compilation, it gives a big boost in performance.

intro

Throughout this class we have done the Jacobi Iteration problem in many different ways. This project is taking that one step further and parallelizing it on the GPU. There is a big performance gain to be had with doing Jacobi with the GPU. There are a lot of issues that can pop up while trying to get this done. There is also a lot to be learned from this troubleshooting and from these unexpected problems.

skipping results section

issues I ran into

The issues that came during this project were sometimes unexpected and difficult to get around. After a bit of research, I made a Jacobi version that should run on the gpu. I went to compile it, it works fine but it isn't running on the GPU. In openMP the is a target directive, it is how a programmer can use other hardware components while using OpenMP, in this case the GPU. My target wasn’t set when it compiled, so it went to default, the CPU. At this point I was running mingw on windows. I downloaded the nvidia cuda toolkit to see if that had the libraries I needed. The only issue was gcc ran in ming and nvcc ran in windows command prompt, but neither in both.

OpenMP directives/ GPU techniques learned

Conclusion