Optimizing the code that analyzes data from gravitational waves will make it possible to learn about binary black hole properties in a realistic time frame, and I will learn about whether or not using the GPU (Graphics Processing Unit) is more efficient than the CPU (Central Processing Unit) with our particular project. Gravitational waves are essentially the curvature of space time in the form of waves that travel outwards from its source. The GPU has been proven to be vastly more efficient than the CPU in solving many problems, though not all. For example, it is used by video game engines to run expansive worlds that need to be available to the player with the fastest loading times possible.[[1]](#footnote-1) In addition, the GPU is proven to be more useful than the CPU to study locust swarms, collective behavior of animals, and improving airspace traffic control because it splits up the task multiple computations by using multiple smaller CPUs.[[2]](#footnote-2) This project is important because gravitational wave analysis from binary black hole mergers allow us to learn about how these systems are formed and test if general relativity holds. We need to specifically focus on gravitational waves because we are studying systems that emit little to no light, black hole and neutron star systems. My part in the project consists of improving and testing the processing time of waveform generation against the current times by switching the code to be processed by the GPU rather than the CPU.

This lab has done work to speed up the process of computing by optimizing the formulas of which are calculating most of their data.[[3]](#footnote-3) For example, they found that the most significant optimization was when they changed their Runga-Kutta integrators to use adaptive size step, which is used for equations of random variations, and interpolation on those steps.[[4]](#footnote-4),[[5]](#footnote-5) However, the time spent on generating the gravitational waveforms is still significantly long. The lab schedules a run of 8 CPU cores at Northwestern during different times of the year to collect data about binary black hole mergers. On average, at total CPU capacity, it takes about 1000 CPU hours, or 120 hours in real-time to analyze completely, the difference coming from using 8 cores during each trial. In addition, recent updates to LIGO (Laser Interferometer Gravitational wave Observatory), which collects gravitational wave data from space, is increasing its sensitivity and therefore will make the waveforms even longer, which is more cumbersome to process. I hypothesize that the GPU will significantly reduce overall processing time by splitting up computations to be run on multiple CPU threads as opposed to using only the main CPU.

My process before the summer consists of learning how to integrate Ordinary Differential Equations (ODEs) – related to orbits of various masses – into programs, learning about GPU structure using OpenCL, and learning about the UNIX operating system environment. However, most of my time will be used learning about OpenCL, which is an open sourced GPU programming language. Therefore, learning about the syntax of GPU programming in OpenCL and applying the knowledge to split up processing tasks will significantly help with the main project during the summer.

During the summer I will be using the actual formulas and modifying the actual code, and I will make the big switch from the CPU to the GPU to begin testing. By the end of the summer I plan to have had implemented my knowledge to the current computer programs that are used to produce waveforms and process the data. The improvement of the data output would drastically reduce the time spent on analyzing the data, which in turn ultimately helps the astrophysicists make conclusions in a realistic time frame and be less dependent on the processing times of the current computer systems.

**Methods**:

Currently I am learning how to integrate ODEs regarding planetary orbits in the C# language, of which I am most familiar. In addition, I am learning how to port programs run on the main CPU to be run by the GPU. It is a long process where I take pieces of code and determine whether or not the task could be split up by the GPU. With OpenCL I will be writing programs for the GPU so that I can learn how to effectively split up the work properly. Since the computer systems I will work with in the summer will be running on UNIX, I will learn UNIX syntax so I can implement my own code and later the lab’s code. Then I will take advantage of and test OpenCL’s parallel computing component against previous processing times; I will collaborate with Pascal Paschos who already has experience with parallel computing. During the summer I will do the switch, with the lab’s code, from the CPU to the GPU and start testing processing times. I will be taking each section of the code and determine which parts can be split up to be processed by separate smaller CPUs (allocated by the GPU). Then I will use OpenCL to allocate these sections of code to be processed simultaneously. Because of the sheer size of the code being transported to the GPU, the summer is a reasonable amount of time to work on this project. By the end of the summer I will be able to come to a final conclusion about the processing times of the GPU versus the main CPU with generating gravitational waveforms.

**Preparations**:

I have working knowledge of various types of coding languages that include C++, C#, Java, Visual Basic (all versions), and META (Scheme/Lisp). I have taken EECS 111, EECS 211, and EECS 310. During the spring quarter I will be taking EECS 311 which is about algorithm analysis and optimization, extremely useful knowledge to have for this type of research. In addition, my mathematical experience continues to the present from taking the MENU MATH 290 series in addition to the mathematical foundations of computer science that comes from EECS 310. I have programmed in Microsoft’s XNA Framework for video game creation; my video games have used a variety of physics applications and screen state management. However, learning about multiple threading coming from OpenCL my video games will be vastly improved. I also intend to learn and research applicable physics concepts with guided help from Will Farr and Ben Farr.

**Conclusion**:

My interest in using computer science for natural world applications led me to help with this gravitational wave project. The project brings together my valued interest in physics, math, and computer science. I hope that the experience I gain from learning about, OpenCL/GPU computing, UNIX, and solving natural science problems with computer science will give me a leading edge on being able to get more hands on work with computer science. A long term goal of mine is to develop and program video game engines and software. I am aware that the best engine programmer is an expert in physics, math, algorithms, optimization, problem solving, and researching skills. Therefore, this project is essential to me in order to get a real feel on what I would be dealing with in the future if I pursue this option. I hope this project leads to many other projects at Northwestern that will benefit me in similar ways.

1. http://developer.nvidia.com/ [↑](#footnote-ref-1)
2. http://developer.nvidia.com/ [↑](#footnote-ref-2)
3. http://arxiv.org/abs/0805.1689 [↑](#footnote-ref-3)
4. https://www.lsc-group.phys.uwm.edu/daswg/projects/lal/nightly/docs/html/LALAdaptiveRungeKutta4\_8c\_source.html [↑](#footnote-ref-4)
5. https://www.lsc-group.phys.uwm.edu/daswg/projects/lal/nightly/docs/html/index.html [↑](#footnote-ref-5)