# Parallel Reaction Diffusion Proposal

Leo Lin (ruilin), Andres Mason (aamason)

## Abstract

Reaction diffusion is a mathematical model that captures the phenomena when two reagents diffuse in a space. In this project, we hope to explore the possibility of accelerating reaction diffusion simulation across multiple GPUs using CUDA.

## Background

When two substances that do not react with each other are mixed, they simply diffuse. However, if the mixing of the two substances results in a chemical reaction that consumes one of the substances and creates more of the other, we observe interesting patterns like the one you see happening in the background in real time.

This is mostly a pixel-wise iterative computation, where each pixel stores the amount of reagent A and reagent B. At each time step, every pixel computes the new amount of reagent A and B based on the amount of A and B in itself and its neighbors. This makes for a great parallel computation problem: its pixel-wise nature allows for parallel computation across the entire image while presenting an interesting communication challenge between neighboring pixels. In terms of parallelism, this problem is very similar to the ocean simulation problem we discussed in class.

## The Challenge

As mentioned before, the biggest challenge in this project is the communication between neighboring pixels in the convolution step. To obtain good speedup at a massive scale, we plan to use a hybrid approach to communication. On the GPU, we will use a shared memory model to make the best use of the fast cache on CUDA streaming multiprocessors and use the global memory to communicate between blocks. One of our goals is to further expand the scale of parallelism by using multiple GPUs (possibility across different nodes.) At this scale, we will need to use MPI to communicate border pixels across GPUs in between each iteration.

The other aspects of this workload should be in our favor. There should be minimal divergent execution as the calculation for each pixel is identical, which is great for making the best use of the CUDA cores. The memory access pattern is also very regular, which should help with memory coalescing. There is also no need for locks because the computation only reads value from the previous iteration. There is not a lot of inherent communication between GPUs. So we are not worried about message passing bandwidth. The biggest challenge will be reducing the time spent in between iterations. This time is spent on launching CUDA kernels, synchronizing

the kernels after each iteration, transferring data between global memory and shared memory, transferring data between host and device, synchronizing the computation across GPUS, and sending border pixels across GPUs.

Resources

We expect to develop on the GHC machines and run larger scale experiments on the PSC GPU nodes, which would allow us to test parallelism across multiple GPUs. If time permits, we will also try parallelizing across multiple nodes. There are some existing C++ implementations of reaction diffusion simulation that we can reference as we work on a CUDA implementation. Reaction diffusion is not a complicated algorithm so we do not expect to run into any major roadblocks there. There are a few websites and videos that provide a good introduction to reaction diffusion.

Goals and Deliverables

Our planned goal is to compute a 1000x1000x1000 3D reaction diffusion with video constructed from intermediate images of the state. We hope to minimize the I/O and get close to live video for the reaction. We would work towards this throughout the project by trying to achieve live video for smaller numbers of pixels, 250x250x250.

Deliverables are tied to our schedule. For each date below, the deliverable is next to it:

- April 9: Unoptimized working code w/ final image
- April 14: Slideshow made from the intermediate states
- April 16: Milestone Report, including current video capabilities
- April 23: Experiment data verifying optimization of I/O and compute times
- April 31: Demo video
- May 5th: Final Project Report, Poster

Platform Choices

We'll use C++ and CUDA as our main languages as they are the most suitable for the GPU programming needs of this project. We will use MPI to communicate between GPUs. These are the tools that we are most familiar with and are the most suitable for the computation needs of reaction diffusion.

Schedule

March 27: Proposal Due

April 9: Unoptimized working code, no I/O for images

April 14: Able to show intermediate states at a reasonable rate

April 16: Milestone Report

April 23: I/O optimization

April 31: Demo video ready, no more code
May 5th: Final Project Report, Poster