CUDA Reduction

Class: CS315 – Distributed Scalable Computing
Whitworth University, Instructor: Scott Griffith

Last Modified: 11/15/2018

Part 1: Reduction Algorithm and Implementation (100 pts)

Big data almost always requires a reduction. Big data input with big data output does not help scale the problem that much. Big data with a reasonable output is the dream of computer science application. This learning module is going to push you towards dealing with this problem.

At first glance, reduction (going from a large data set to a smaller data set) seems straightforward. But you will find that the process is inherently non-optimal on a GPU. If you set your threads to map to input cells, you are going to waste occupancy as the data set gets smaller. If you set your threads to map to output cells, for most data sizes you are not going to be leveraging the massively parallel hardware.

For this LM you are going to need to implement a generalized reduction kernel. It will take in a single dimensional array of size N and reduce it by a factor of R. The reduction will be an average of the reduced variables.

As an example:

Input array: N=9

9	4	2	6	11	8	9	3	4

Output Array: reduction factor R: 3, Output size: N/R = 3

5	8.3	5.3

Be aware of N and R sizes that result in uneven outputs. You cannot ignore that, you have to do something. I might suggest a wrap: if the last reduction needs to be 5 elements, but the remaining array only has 3 elements (N-3, N-2, N-1), take elements 0 and 1 into the average. Or you can take the remainder of the remaining values.

Your code should also work on N=R. This would average the whole array. It should also work on R =2 which is just going to pair-wise average the array. N can be anywhere between 2^10 up to 2^25

There are many ways to do this. Many of them are non-optimal. Some are optimized. **You will be graded on how well the kernel utilizes the GPU.**

(Up to 20 pts Bonus) Document your code iterations. You should have a timing infrastructure in place that you should be able to get performance metrics. As you attempt iterations, you should see time changes.