CS194-15: Engineering Parallel Software Assignment 4

October 6, 2014

1 Overview

The goal of this assignment is to introduce GPU programming with OpenCL. As GPU programming includes both a new programming model and OpenCL includes a large (and verbose) API, we are providing a small OpenCL framework to simplify some of the ugly details of OpenCL programming. Your task for this assignment is to first understand the simple vector increment program (incr) we have provided. After understanding this simple OpenCL program, you'll implement both vector-vector addition and matrix multiply in OpenCL.

As mentioned in class, there are 5 steps in a basic OpenCL program:

1. Initialize the GPU (we do this for you)
2. Compile the OpenCL kernels (we do this for you)
3. Allocate memory on the GPU (you do this)
4. Set kernel arguments (you do this)
5. Read results back from the GPU (you do this)

For this assignment, you have to write both the *kernel code* and interface with the *OpenCL runtime*. We have also provide a code skeleton with comments to direct you where your code is required.

2 Tasks

2.1 Understanding vector increment

We've provided a working implementation of vector increment in the two files, incr.cpp and incr.cl. We want you to understand the code in these two files as this example provides a demonstration of the core ideas of OpenCL. To demonstrate your understanding, we want you to provide comments for the code in both files. Make sure to clearly demonstrate your understanding of what each call to the OpenCL runtime does. We don't want you to write a novel in your comments, just a sentence or two will be sufficient. To make this task easier, we've placed the OpenCL reference card on bSpace under the references section.

Please include your fully commented versions of incr.cl and incr.cpp in your homework submission.

2.2 Implement vector vector addition

```
void vvadd(float *Y, float *A, float *B, int n)
{
  for(int i = 0; i < n; i++)
    Y[i] = A[i] + B[i];
}
```

Listing 1: Vector vector addition

Now that you've mastered OpenCL, we want you to code up vector-vector addition in OpenCL. If you've forgotten what vector vector addition looks like, we've provided a serial C reference in Listing 1. This is the grand-daddy of data-parallel benchmarks and no parallel computing class would be complete without forcing the students to implement it. The code provided for vector increment should provide a good reference of the required OpenCL functions and a good starting point for your implementations.

Please implement your OpenCL kernel code in vvadd.cl and your runtime code in vvadd.cpp. Provide comments in both files.

2.3 Implement matrix multiply

```
void matmuld(float *Y, double *A, double *B, int n)
{
  for(int i=0;i<n;i++)
    for(int j=0;j<n;j++)
      for(int k=0;k<n;k++)
        Y[i*n+j] += A[i*n+k]*B[k*n+j];
}
```

Listing 2: Three-nested loop matrix multiply

Please implement matrix multiply (Listing 2 in OpenCL. We've provided a starting point in the files matmul.cpp and matmul.cl. We'd like you to use 2D work groups (in comparison to the 1D work groups you've used in the previous tasks).

Please implement your OpenCL kernel code in matmul.cl and your runtime code in matmul.cpp. Provide comments in both files.