HW 4
Numerical Integration with CUDA

Carlos Vazquez Gomez

April 24, 2019

1 Timing Analysis

Within my elapsed time I only include the execution of the kernel (which begins by calculating the function values before integrating them). I exclude allocation and memory transfers by only timing the kernel:

```
cudaEventRecord(start);
kernel<<<1,threads>>>(d_F,d_value,num_threads);
cudaEventRecord(stop);
```

But from the output of the profiler, we can see that the memory transfer, which copies only 8 bytes, took about the same time (4 ms) as the kernel execution itself when run with 512 threads and 1,000,000 points. The following table demonstrates the execution times for kernels using 1,000,000 points with 256, 512, and 1024 threads. I conclude that the kernel with 1024 threads may have

Threads	kernel time (ms)
256	6.17
512	4.67
1024	4.77

Table 1: Execution times for different thread numbers

performed the same as the one with only 512 threads because of contention for the memory bus due to the atomicAdd() instruction. There must be a balance between the number of threads independently summing chunks of the array, and the number of competitors to critical sections of the code.