Homework 2, ME498/599, WI 2017

Hongxi JIN (1628292)

Set-Up Tasks:

1. CUDA Installation

Completed

2. "deviceQuery"

Device 0: "GeForce 940MX"

Total amount of global memory: 2048 MBytes (2147483648 bytes)

(3) Multiprocessors, (128) CUDA Cores/MP: 384 CUDA Cores

Total amount of constant memory: 65536 bytes

Total amount of shared memory per block: 49152 bytes

Maximum number of threads per multiprocessor: 2048

Maximum number of threads per block: 1024

Max dimension size of a thread block (x,y,z): (1024, 1024, 64)

Max dimension size of a grid size (x,y,z): (2147483647, 65535, 65535)

3. "nbody"

3072 bodies, total time for 10 iterations: 737.939 ms

= 0.128 billion interactions per second

= 2.558 single-precision GFLOP/s at 20 flops per interaction

Programming Exercise:

1. Vector Operation

See "vec_fun" notes in "kernel.cu" file.

2. Testing

a. N = 32;

b. u2 = { 0.25; }

v2 = { 0.75; }

c. u2 + v2 = { 1.00; }

d. -3u2 + v2 = { 0.00; }

e. u2 * v2 = { 0.19; }

3. Library Construction

Refer to "hw2.h" and "kernel.cu" files.

4. Scale Operation

a. When coded with CUDA kernel function, the scale operations can not be operated together with computing for individual entries of the output vector. If not operate with the "addWithCuda" function, then the sum must be operated after computation for the output vector have been completed.

b. Not like what's done in C programming. This is because the calculation in each thread is operated together instead of one by one. Thus, accumulation cannot be accompanied with entry computation.

c. Inner product of u2 and v2 = 6.00

Revised inner product of u2 and v2 = 6.00

Euclidean norm of u2+v2 = 5.66

d. No. This is because the computation of each entry is done simultaneously.