CSCE 435 Summer 2025

HW 6: GPU Programming

Given a set of n points in a plane, you need to determine the distance between the closest pair of points. Distance between points $p_i=(x_i, y_i)$ and $p_j=(x_j, y_j)$ is computed as

$$d_{ij} = \sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2}$$

You are provided with a program `nbody.cu` that has the following capability:

- The code generates coordinates for n points on the host;
- The coordinates are copied to the device memory;
- An incomplete kernel function is provided that is intended to compute pairwise distances between all pairs of points, determine the minimum distance, and save the value in a device variable;
- The device variable is copied to the host;
- The value is compared with the minimum distance calculated on the host;
- The code reports the time spent in the kernel function, data transfer between host and device, and host computation.

You need to modify the kernel function to compute the minimum distance. You are allowed to make other changes to the code, which facilitate parallelization on the GPU.

1. (70 points) You need to develop CUDA-based parallel code to compute the distance between the closest pair of points on a GPU. 50 points will be awarded if the code compiles and executes the following commands successfully.
```
./nbody_sp25.exe 16
./nbody_sp25.exe 1024
./nbody_sp25.exe 9999
```
   10 points are reserved for a brief write-up describing the changes you made to the code. Additional 10 points are reserved for performance of the code: speed improvement obtained by the code over the host code.
2. (20 points) Execute the code for $n = 2^k$ for k = 4,…,10. Plot GPU and CPU execution time versus k on the same plot to demonstrate how execution time varies with the problem size on these platforms. Use logarithmic scale for the x-axis. Next, plot the GPU and CPU execution time for $n = 2^k$ for k = 11,…,16. For what value of n does the GPU code become faster that the CPU code?
3. (10 points) Plot the data transfer time from host to device and from device to host on the same plot for $n = 2^k$ for k = 4,…,16.

Submission: Upload two files to Canvas:

1. Submit the file `nbody.cu`.
2. Problem 1 writeup, 2 & 3: Submit a single PDF or MSWord document with your responses.

Helpful Information:

1. You may use Grace or Faster for this assignment.
2. Information on compiling and running CUDA programs on a GPU for Grace is available at https://hprc.tamu.edu/kb/User-Guides/Grace/Compile/#cuda-programming.
3. To develop code interactively, log on to one of the GPU-equipped login nodes on Grace; these nodes are named graceX.hprc.tamu.edu, where X is to be replaced by the numbers 1, 2, or 3. See https://hprc.tamu.edu/kb/User-Guides/Grace/Hardware/#login-nodes for additional information. If you use portal.hprc.tamu.edu to access Grace, you are randomly assigned a node where X can be 1-5. You will then need to use ssh to log into a GPU-equipped login node from the initial shell.
4. Load the modules for compiler and CUDA prior to compiling your program. At the time of preparing this assignment, the following module combination worked on Grace (later modules did not work):
   ```
   module load intel/2023a CUDA/12.2
   ```
5. Compile C/C++ programs using `nvcc`. For example, to compile `code.cu` to create the executable `code.exe`, use
   ```
   nvcc -o code.exe code.cu
   ```
6. The run time of a code should be measured when it is executed in dedicated mode. Check out https://hprc.tamu.edu/kb/User-Guides/Grace/Batch/#job-examples for sample batch files. To execute the code on a GPU-equipped node, you must use the submission options for GPUs.