CUDA Homework Assignment 1

Hao-Tse , Hsiao

B12902100

Abstract. In Homework Assignment 1, we are required to define a matrix C such that $c_{ij} = 1/a_{ij} + 1/b_{ij}$. We will compute this matrix using two methods: first, by performing the calculation on a CPU; and second, by utilizing a GPU.
Finally, we aim to determine the optimal GPU block size that maximizes computational efficiency.

1 Task Description

We begin by initializing two NN matrices, A and B , and then define C such that $c_{ij} = 1/a_{ij} + 1/b_{ij}$.The input matrices A and B (where $N = 6400$) are initialized with random values between 0.0 and 1.0. The optimal block size is determined through experimental evaluation.

To avoid division by zero errors, we ensure that no element in A or B is zero. Initializing the elements with values in the range [0.1, 1.0] is an effective approach to achieve this.

2 Task Result

After performing the matrix computation on both the CPU and GPU, we compiled the results in the following table. The number in brackets represents the block size N , where the actual block area on the GPU is $N \times N$.Each value in the table represents the average running time calculated from three executions of the matrix computation. And every calculation result on GPU is the same as the calculation result on CPU (which norm is equal zero).

Table 1. Execution Time Comparison

Device	Total Running Time (ms)	speed up (compare with CPU)
CPU	314.979	1
GPU(2)	243.382	1.294
GPU(4)	199.027	1.583
GPU(8)	193.785	1.625
GPU(16)	189.850	1.659
GPU(32)	193.145	1.631

As shown in the results, the optimal performance was achieved with a block size of 16, followed by block sizes of 8 and 32.

3 Disscussion

As we can see, the block size = 16 has accelerate the most, and then is block size = 8 and 32. This result indicates that a block size of 16 × 16 achieves the optimal balance for GPU computation.

A block size that is too large may reduce parallelization efficiency, while a block size that is too small could create excessive thread overhead, reducing computational performance.

4 Reference

1. Introduction to CUDA Parallel Programming — Homework Assignment 1