Tutorial № 2

JI Zhuoran, The University of Hong Kong 28/07/2018

We have graduated from vector addition and will now take a look at vector dot products (sometimes called an inner product). We will quickly review what a dot product is. For two vector $\mathbf{a} = [a_1, a_2, ..., a_n]$ and $\mathbf{b} = [b_1, b_2, ..., b_n]$. The dot products is $a_1 \cdot b_1 + a_2 \cdot b_2 + ... + a_n \cdot b_n$ Noted that instead of then storing these values to a third, output vector, we sum them all to produce a single scalar output.

The main objectives in this tutorial are to learn about:

- how to do block level thread synchronization
- how to implement global reduction, a key requirement for many application codes
- understand how memory access pattern can influence the performance

Problem 1

In this problem, you will learn the importance of synchronization and how to do global reduction:

1. Read section B.6 in the **CUDA C programming guide** and understand what is synchronization
2. Read through **prac1a.cu** and add synchronization instruction(s) to make the code correct

Problem 2

However, the implementation is somehow stupid. In this problem, we will learn the difference between different memory access patterns.

1. Read the NVIDIA's tutorial, and understand the difference between kernel 1, kernel 2 and kernel 3
2. optimize the **prac2b.cu** with what you learn
3. compile and compare the execution time

