Tutorial № 4

JI Zhuoran, The University of Hong Kong 29/07/2018

Most of the GPUs are Single Instruction Multiple Data(SIMD) execution model. By the very nature all thread performs same instruction in parallel, at least conceptually. If we want to perform different operations on the different thread in same block, say, if A, then op1, else op2, then we essentially force these thread to do op1 first and then op2. This "unneccessary" serialization carries performance penalty. The aim of this tutorial are:

- test those cost
- try to avoid this penalty

Problem 1

In this problem, you will implemented a benchmark application:

1. Read the topic in StackOverflow and understand the concept of branch divergence.
2. read through **prac4a.cu** and draw a diagram of which thread performs which operation.

Problem 2

In this problem, we are going to rearrange the "thread positation" (actually we rearrange the input data) to reduce the penalty introduced by branch divergence.

1. Read through the **prac4b.cu** and try to rearrange the input data before launching the kernel function to avoid branch divergence
2. compile and compare the execution time