Weekly Report - Learning CUDA to Understand the Code

MSc Project
Runchao Han

December 26, 2017

1 Progress of the Last Week

1. Successfully imported the Ethminer project to Nvidia Visual Profiler
2. Learned about CUDA basic principles and programming
3. Compared two Ethash implementations in Ethminer
4. Got more accurate data

2 Imporing Ethminer to Nvidia Visual Profiler

The problem I met last two weeks is that I installed two CUDA versions, and the version in `/bin` is the false one.

After I substituted related binary files with correct ones, the importing succeeded, as shown in Fig. 1.

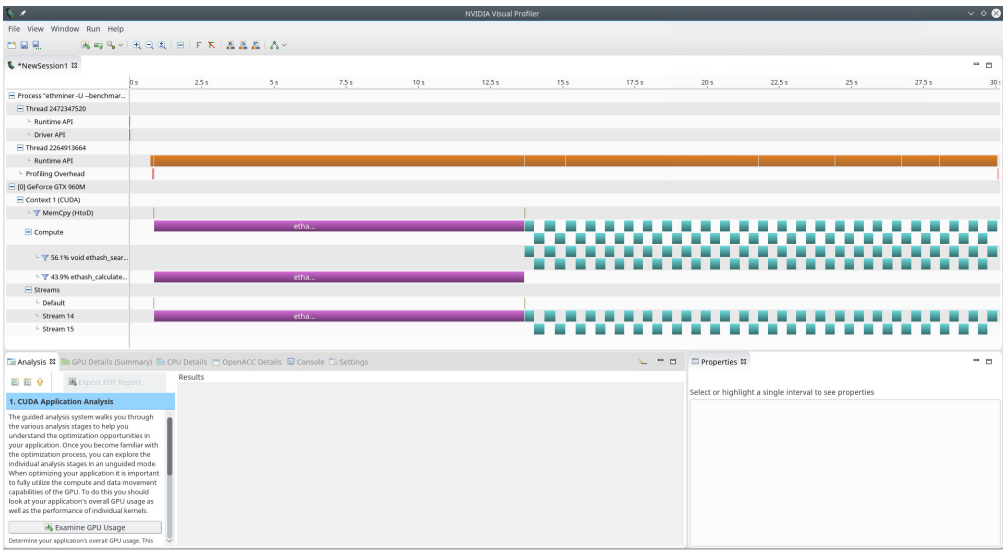

Figure 1: Importing the Ethminer to Nvidia Visual Profiler

However, this tool is unsuitable for profiling a small part of the code in a big project as far as I know. Before I isolate the critical code from the project, I will still use the timestamp approach.

3 CUDA Basic Principles

This two week I spent most time on learning GPU and CUDA, because my hardware knowledge is still rudimentary.

3.1 An Introduction to GPU

Graphical Processing Unit (GPU) was designed for computer graphic applications at first, focusing on parallel scientific computing. The comparison between CPU and GPU is shown in Fig. 2.

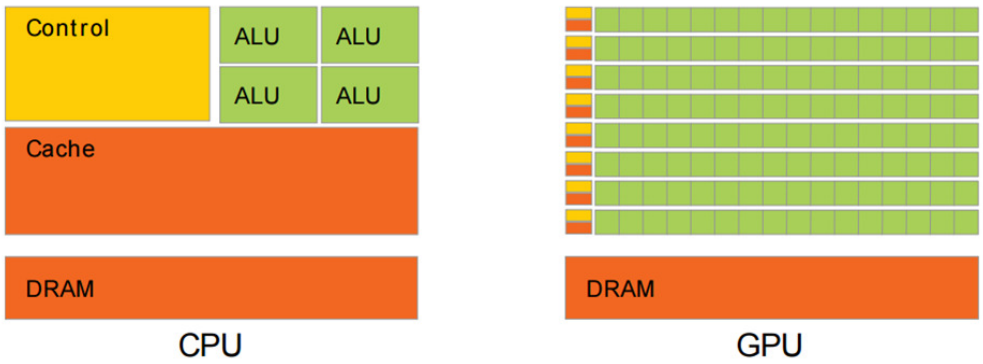

Figure 2: The architecture comparison between CPU and GPU

It is shown that GPU contains much more ALUs which execute computations. Moreover, state-of-the-art GPUs contain a large number of cores so suitable for parallel computing.

3.2 The Lifecycle of a CUDA Program

CUDA was introduced in previous reports, so this section focuses on the execution of a CUDA program.

Fig. 3 shows the logical view of CPU, GPU, main memory and GPU memory.

A CUDA function is called a "Kernel function".

Typically, a CUDA program is executed by following steps:

1. Launch the binary file
2. Load data to the main memory
3. Allocate space to data structures in the Graphics Card memory
4. Copy data from the main memory to the Graphics Card memory
5. Invoke the kernel function executed by GPU
6. Copy the result from the Graphics Card memory to the main memory

3.3 Basic APIs

3.3.1 Function Type Specifiers

Function type specifiers specify the execution place of a function, which are:

- *__device__*
- *__glocal__*

3.3.2 Variable Type Specifiers

3.3.3 Existing Variables

3.4 Progammers' Perspective: Grid, Block and Thread

3.5 Hardware's Perspective: SP, SM and Warp

3.6 The New Shuffle APIs

4 A More Detailed Profiling on Ethash (Shuffled)

5 Miscellaneous

-

6 Next Week's Plan

1.

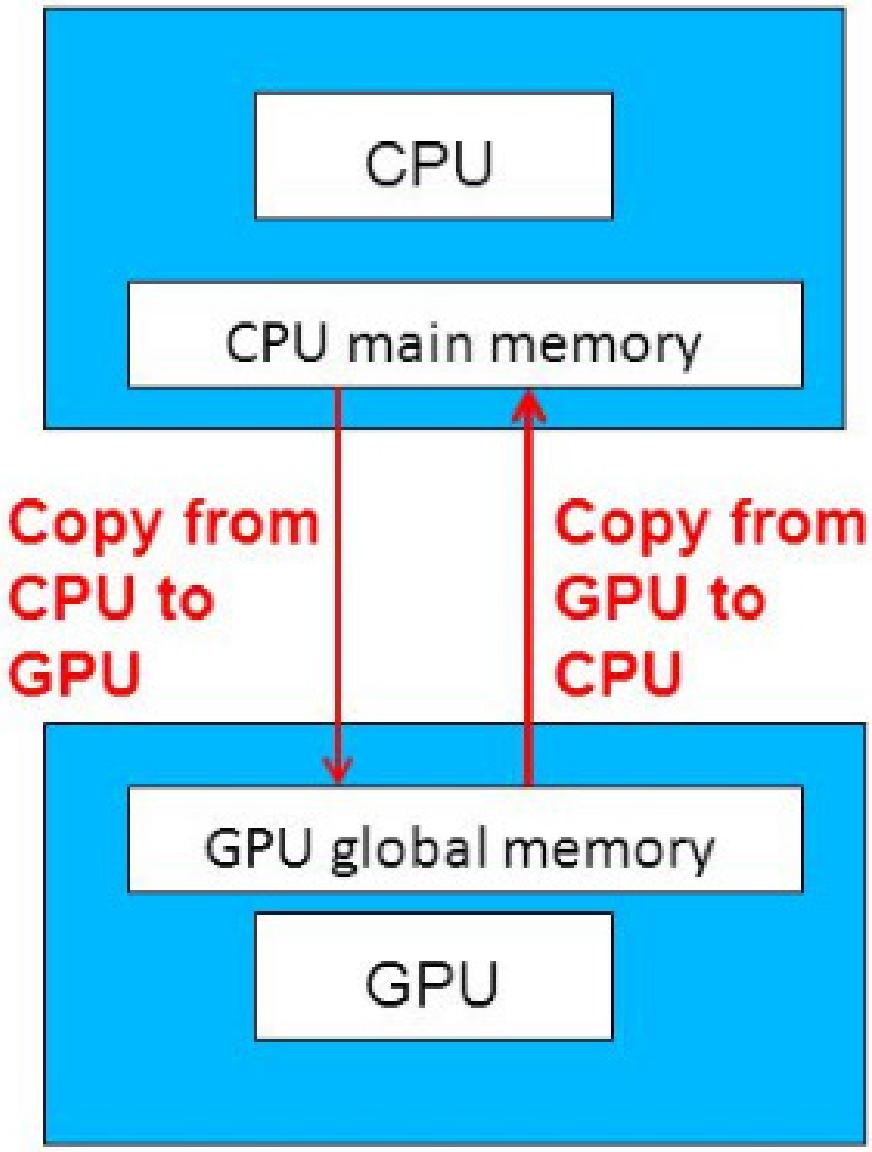

Figure 3: The architecture comparison between CPU and GPU