Comparative study of methods of Principal Component Analysis of automatic segmentation of functional magnetic resonance imaging (fMRI).

1. Motivation - how to proces huge volume of data to obtain results in a real-time application
2. In the beginning of the last decade an increase of CPU (Central Processing Unit) clock speed was generally stopped. There are some reasons for that, the essential one is because of the thermal losses.
3. Nowadays in order to maintain continuous increase of the performance processors comprise many cores (multicores processor).
4. This implies that a paradigm of sequentially written programs has become unable to fully utilize this architecture. To achieve that it is necesarry to develop parallel applications i.e. applications which exploit all available cores so that the total execution time of the program is lower than the one implemented sequentially using one CPU.
5. In practice there are two main approaches to develop parallel applications. The first one is about processors containing several cores (2,4,6,8,…), each one (processor) processing several „heavy” threads. This approach is common for clusters of CPU processors.

Another type of processors are those which contain many cores (hundreds, thousands) being able to proces many „light” threads. This is how GPU (Graphic Processor Unit) works.

1. Nowadays numerical applications with big computational complexity are implemented mainly on GPU. This is because of some of the features they have. Graphic procesor unit (GPU) is specialized for compute-intensive, highly parallel computation - exactly what graphics rendering is about - and therefore designed such that more transistors are devoted to data processing rather than data caching and flow control [1 cuda programming guide].
2. Tutaj może o tym co na slajdzie 11 pierwszego wykładu RIM – dlaczego cuda – cena rozpowszechnienie, itd.

Other reasons for that is they are cheap and very common (many GPUs are able to do computational work).

1. Objectives

Celem tej pracy było zaimplementowanie równoległej wersji algorytmu PCA dla przetwarzania fMRI na platformie CUDA i porównanie wydajności / czasu z odpowiednią implementacją algorytmu PCA w Matlabie. W dalszej części pracy została opisana implementacja w CUDA, porównanie wyników czasowych uzyskanych na testach przeprowadzonych na danych z badań fmri.

The aim of this work was to implement PCA algorithm on CUDA (Compute Unified Device Architecture) platform in order to obtain better performance (speed-up) regarding to Matlab version of the algorithm. The study includes the comparison of the execution time of the methods implemented in Matlab and CUDA. These methods were tested for real problems (on data sets obtaining from fMRI scanning).

1. Explanation of CUDA and Matlab computational platform

CUDA language – as its core are three key abstractions – a hierarchy of thread groups, shared memories and barrier synchronization

1. Różnice między CPU a GPU – slajd 6, wykład 1
2. Platforma CUDA – wykład 2 RIM
3. Dlaczego GPU – wydajność, przyspieszenie, przepustowość, wykorzystanie krzemu, osiągnięcia technologii CUDA – wykład 1 RIM (buforowanie i pamięć, skalowalność, itd.)
4. CPU cores are designed to execute instructions sequentially, so they are specialized in flow control. They have much bigger cache than GPUs to minimize memory access latency, because memory bandwitdh in CPUs are generally low.

On the other hand GPU’s architecture was optimized for games, so they contain many simple floating-point ALU executing in groups millions of instructions. The control flow is simplified. Many „light” threads executing simultaneously, hiding memory access.

1. CUDA platform consists of a host (CPU) and one or more devices (NVIDIA GPU) under host’s control. The NVIDIA GPU architecture is built around a scalable array of multithreaded Streaming Processors (SMs) each one containing its own memory and registers.

When a kernel is executed cuda spans threads grouped in blocks over multiprocessors.

Thread hierarchy

NVIDIA developers named CUDA parallelism model by SIMT (Single Instruction, Multiple Threads) which is very similar to SIMD model (Single Instruction, Multiple Data).

SIMT model

1. Two essential advantages of CUDA technology over CPU processors are memory bandwidth (B/s) and computional throughput (FLOPS/s) – performance.