Ways to use GPU

Sivasankar Arul, IT4Innovations

June/2021

Univerza v Ljubljani

TECHNISCHE UNIVERSITÄT WIEN

CINECA consorzio interuniversitario

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

Objectives

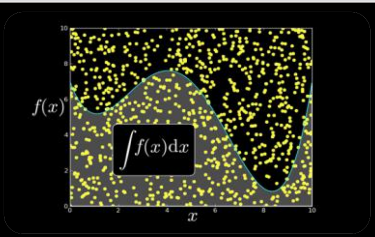
Objectives

- Ways to utilize GPU

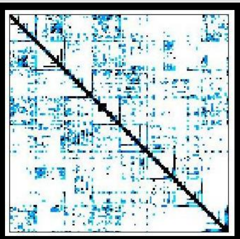
Libraries

Several libraries has GPU acceleration

NVIDIA cuBLAS



NVIDIA cuRAND


NVIDIA cuSPARSE

NVIDIA NPP

GPU VSIPL

Vector Signal Image Processing

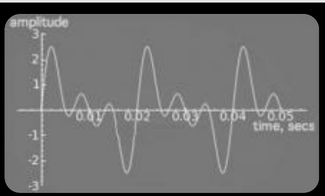
CULA | tools

GPU Accelerated Linear Algebra

MAGMA

Matrix Algebra on GPU and Multicore

open source initiative



NVIDIA cuFFT

ROGUE WAVE SOFTWARE

IMSL Library

Building-block Algorithms for CUDA

CUSP

Sparse Linear Algebra

open source initiative

Thrust

C++ STL Features for CUDA

open source initiative

Libraries

Several libraries has GPU acceleration

Features:

- In depth knowledge of GPU programming is not needed.
- The libraries follow standard APIs therefore can used in existing code with minor modifications.
- High quality and suitable for variety of application.

Libraries

Several libraries has GPU acceleration

Features:

- In depth knowledge of GPU programming is not needed.
- The libraries follow standard APIs therefore can used in existing code with minor modifications.
- High quality and suitable for variety of application.

Compiler Directives

Compiler Directives

For C, C++, Fortran

- Statements in the source code.
- Instructs the compiler to recognize those parts of the code that should be run in GPU.

For example: OpenACC

Compiler Directives

Serial Code

```
for (i = 0; i < Nrow; i++ ){
sum = 0.0;
        for (j = 1; j < Nrow; j++ ){
             sum+ = A[i*Nrow + j]*x[j];}
         }
         b[i] = sum;
}
```

Parallel Code for GPU

```
#pragma acc parallel loop
for (i = 0; i < Nrow; i++ ){
sum = 0.0;
        for (j = 1; j < Nrow; j++ ){
             sum+ = A[i*Nrow + j]*x[j];}
         }
         b[i] = sum;
}
```

Compiler Directives

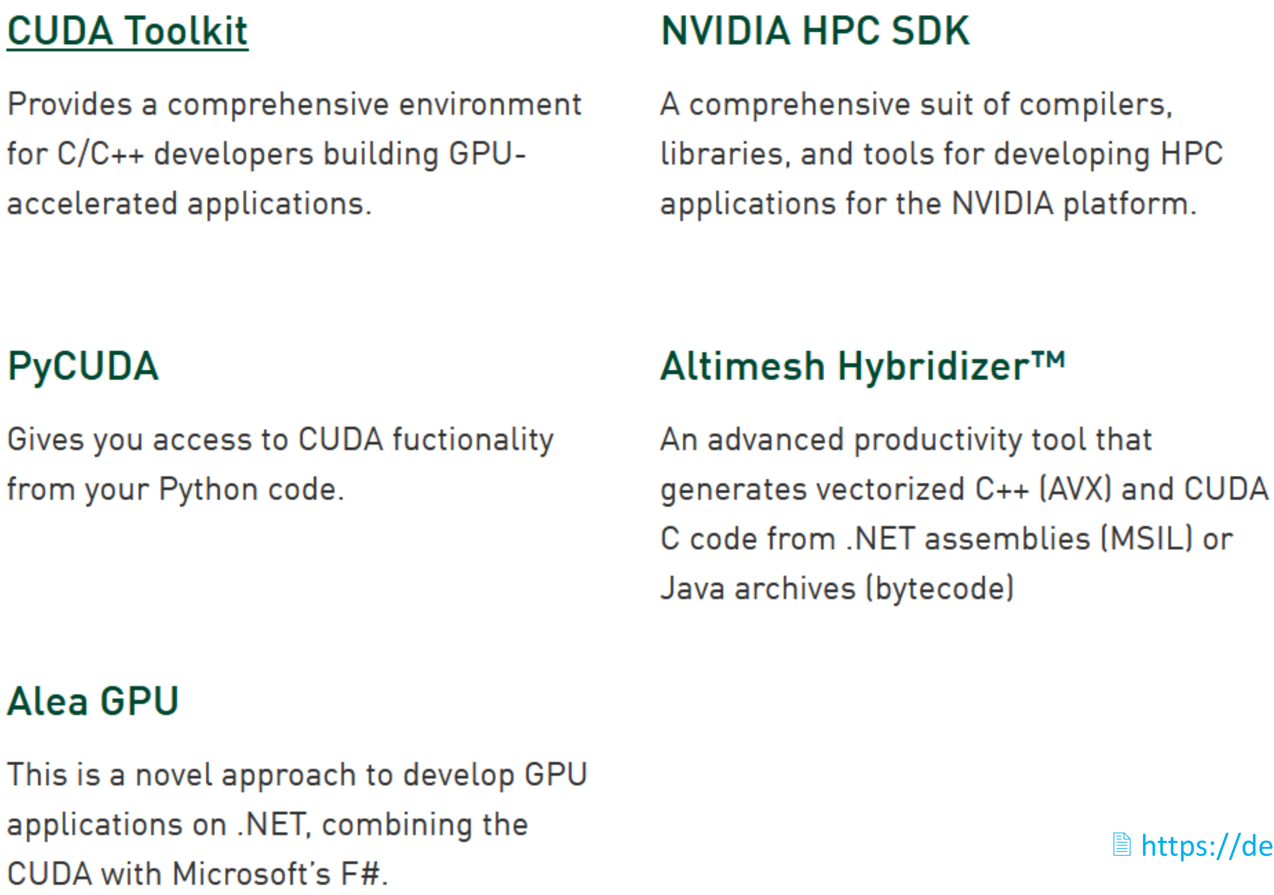
Compiler Directives

Features:

- It is simple, powerful and portable.
- Compiler does parallelism management and data movement.
- Different compiler versions give different performance.

Programming Languages

Programming Languages



CUDA Toolkit

Provides a comprehensive environment for C/C++ developers building GPU-accelerated applications.

NVIDIA HPC SDK

A comprehensive suit of compilers, libraries, and tools for developing HPC applications for the NVIDIA platform.

OpenACC

Directives for parallel computing, the most popular GPU parallel programming model for researchers and technical programmers.

PyCUDA

Gives you access to CUDA fuctionality from your Python code.

Altimesh Hybridizer™

An advanced productivity tool that generates vectorized C++ (AVX) and CUDA C code from .NET assemblies (MSIL) or Java archives (bytecode)

OpenCL™

OpenCL is a low-level API for GPU computing that can run on CUDA-powered GPUs.

Alea GPU

This is a novel approach to develop GPU applications on .NET, combining the CUDA with Microsoft's F#.

🗎 https://developer.nvidia.com/language-solutions

Programming Languages

Programming Languages

For example: CUDA C

```
__global__ void vector_add (float *out, float *a, float *b, int n){
    int index = blockIdx.x *blockDim.x + threadIdx.x;
    if (index < n){
        out[index] = a[index] + b[index];}
}
```

```
int block_size = 256;
int grid_size  = (N+block_size)/block_size;
vector_add<<<grid_size,block_size>>>(d_out, d_a, d_b, N);
```

Programming Languages

Programming Languages

Features:

- Good control of parallelism and data movement
- Can be used for any type of computation
- Good performance

Thank you for your attention!

http://sctrain.eu/

Univerza v Ljubljani

TECHNISCHE UNIVERSITÄT WIEN

CINECA consorzio interuniversitario

VSB TECHNICAL UNIVERSITY OF OSTRAVA | IT4INNOVATIONS NATIONAL SUPERCOMPUTING CENTER

Co-funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission.
This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.