Optim_Lab

December 6, 2023

Massive parallel programming on GPUs and applications, by Lokman ABBAS TURKI

1 10 Explicit and implicit simulation schemes for Black and Scholes Partial Differential Equation

1.1 10.1 Objective

This is the third lab of a series of four labs dedicated to the simulation of Parabolic Partial Differential equations using discretization schemes. The main purpose of the developed code is to improve further both the explicit and the implicit solutions using the shared memory. Indeed, although the arrays are sufficiently small and thus are already efficiently and automatically cached, using shared will increase further the performance of the overall algorithm. The NP function is used to check the simulation results and thus it should not be modified.

As usual, do not forget to use CUDA documentation, especially:

1) the specifications of CUDA API functions within the CUDA_Runtime_API.
2) the examples of how to use the CUDA API functions in CUDA_C_Programming_Guide

1.2 10.2 Content

Compile PDE.cu using

[ ]:
```
!nvcc PDE.cu -o PDE
```

Execute PDE using (on Microsoft Windows OS ./ is not needed)

[ ]:
```
!./PDE
```

As long as you did not include any additional instruction in the file PDE.cu, the execution above is supposed to return incorrect values on the left column. The right column is supposed to contain the true results that we should approximate with the discretization scheme.

1.2.1 10.2.1 PDE_diff_k5 and memory copy

The kernel PDE_diff_k5 is based on the syntax of the kernel PDE_diff_k3. In PDE_diff_k5 we rather use the shared memory to reduce the access time to the data.

a) Justify the allocation of sizeof(MyTab) for the array on the device and write the necessary code for CPU2GPU and GPU2CPU memory copy.

b) Replace as much as possible the use of global memory with shared memory allocated dynamically.

c) Complete the syntax of PDE_diff_k5 and compare it to PDE_diff_k3.

1.2.2 10.2.2 PDE_diff_k6 and memory copy

The kernel PDE_diff_k6 is based on the syntax of the kernel PDE_diff_k4. In PDE_diff_k6 we rather use the shared memory to reduce the access time to the data.

a) How many threads should be involved to copy the data from global to shared memory?

b) Replace as much as possible the use of global memory with shared memory allocated dynamically.

c) Complete the syntax of PDE_diff_k6 and compare it to PDE_diff_k4.

d) What other optimizations can be added to the code?

[ ]: