Inge5SE – Parallel Programming 10/2023 – Lab3 : OpenMP

Alexandre Berne, Etienne Hamelin

aberne@inseec-edu.com, ehamelin@inseec-edu.com

0. Know your hardware

During this course, you will measure the performances of many programs running under Linux. You'd better use a native Linux machine, or a Mac, or if not available, use Windows' Subsytem for Linux (WSL, https://doc.ubuntu-fr.org/wsl) or Oracle Virtualbox.

For your performance measurements to be actually useful, **always** check that:

- code are evaluated on the same machine for the whole lab,
- your machine runs at least 4 CPU cores (esp. in Virtualbox),
- your laptop is plugged in, and battery charged (otherwise, the OS might activate power-saving mode),
- minimize background load (quit all games, interaction-heavy web page, etc.),
- measure at least twice, to ensure that measurements are more or less constant.

The code is provided in the "source" directory; please modify only in the "work" directory; place your report (as a PDF file) in the "report" directory. When you're ready, modify the "submit.sh" script with your names, run it: it will put your work and report in a .tar.gz archive file. Submit the tar.gz file on the boostcamp interface.

Q1: *What is your OS/hypervisor system configuration?*
How many CPU physical and logical cores does your PC run? How much RAM memory?
Use the commands lscpu, lsmem, cat /proc/cpuinfo, and cat /proc/meminfo to answer.

1. Performance & caching

Look at program « 1-ln2 ». It computes an approximate to the series $\sum_{n=0}^{\infty} \frac{(-1)^n}{n} = \ln(2)$. Look at the role of variable k.

Q2 : Compile this program, draw a graph of its execution time as a function of k. Use command make ln2.csv *to automate the measurement part.*

What does `__attribute__ ((aligned (64)))` *mean (line 13) ?*

Explain how the value of k *affects execution time.*

Hint: it's related to multicore cache behavior.

2. Introduction to OpenMP

Discover OpenMP with this simple example.

Q3 : Using OpenMP, parallelize the program in work/2-ln2-omp

3. Basic matrix algebra

Figure 1 - take the red pill, and you stay in wonderland...

Linear algebra and matrix operations might remind you of a painful past in your curriculum… However it's a tool so powerful in computer science that it can not be ignored! Matrices are used everywhere, from 3D graphics and game engines, most forms of simulators, and even at the heart of neural network AI engines!

A matrix $\boldsymbol{A}$ is represented by a 2-dimension array. In linear algebra class, we usually note $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ a matrix of m rows and n columns, and we note a_{ij} its elements. To follow a computer science convention, indices are counted from 0. In memory, matrix elements are often stored in memory in "row-major" order. E.g. a 3×2 matrix $\boldsymbol{A} = \begin{pmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \\ a_{20} & a_{21} \end{pmatrix}$ will be stored at consecutive memory addresses in the order: $a_{00}, a_{01}, a_{10}, a_{11}, a_{20}, a_{21}$.

The Frobenius norm of a matrix $\boldsymbol{A}$ is defined as: $\|\boldsymbol{A}\| = \sqrt{\sum a_{ij}^2}$.

Q4 : Compare functions `mat_norm_ij` *and* `mat_norm_ji`*. One of them runs quicker than the other. Explain why. Hint: at this point, there's only 1 thread.*

Q5 : Implement function `mat_norm_omp` *using OpenMP. Verify that the computed value is correct. You may experiment with various OpenMP features, looking for the most efficient implementation.*

The cross-product of two matrices $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n \times p}$ is defined as matrix $\boldsymbol{C} \in \mathbb{R}^{m \times p}$ where:

$$\forall i \in [0 \dots m-1], j \in [0 \dots p-1],$$

$$c_{ij} = \sum_{k=0}^{n-1} a_{ik} \cdot b_{kj}$$

Q6 : in work/Implement the matrix multiplication algorithm using OpenMP. Look for the most efficient implementation.