TDTP6 : OpenMP tasks & SIMD programming

December 5, 2022

Exercice 1 – Fibonacci

The Fibonacci sequence is defined by the recurrence

$$\begin{aligned} & \mathcal{F}_0 = 0, \\ & \mathcal{F}_1 = 1, \\ \forall n \geq 2, \qquad & \mathcal{F}_n = \mathcal{F}_{n-1} + \mathcal{F}_{n-2}. \end{aligned}$$

1. Retrieve from Moodle the (sequential) code which computes terms of this sequence. Parallelize it with OpenMP, and check that your code gives correct results.
2. Compare the performance of your parallel version against the performance of the provided sequential version, for relevant values of n. How would you improve the parallel performance?

Exercice 2 – *QuickSort*

If you don't remember it, read about the QuickSort algorithm on Wikipedia to understand its general principle.

1. Retrieve from Moodle the (sequential) code which implements the *QuickSort* algorithm. Parallelize it with OpenMP, and check that your code gives correct results.
2. Compare the performance of your parallel version against the performance of the provided sequential version, for relevant sizes of arrays. How would you improve the parallel performance?

Exercice 3 – Programmation SIMD

For each of the following examples, retrieve from Moodle the corresponding code and try to vectorize it using:

- the compiler's automatic vectorization options;
- the relevant directives from OpenMP;
- AVX intrinsics.

The examples:

- a term-by-term product (on single-precision floating point numbers);
- a scalar product (on single-precision floating point numbers);
- a product of two matrices of size 512×512 (whose entries are single-precision floating point numbers).