TP 2 : Programmation parallèle - SIMD et OpenMP

Vous pouvez trouver les prototypes de toutes les fonction SIMD ainsi que leur coût en cycles sur le site d'Intel : `http://software.intel.com/sites/landingpage/IntrinsicsGuide`.
Pour compiler, utilisez la commande suivante :

g++ -std=c++11 -O3 -fno-tree-vectorize

Pour compiler avec openmp, utilisez la commande suivante :

g++ -std=c++11 -O3 -fno-tree-vectorize -fopenmp

Le but de ce TP est de mettre en avant l'impact de la parallélisation de code avec OpenMP et SIMD. Pour ce faire, nous allons d'abord travailler sur la fractale de Mandelbrot en scalaire puis en SIMD pour enfin finir sur des optimisations OpenMP. Plusieurs versions du code seront donc faites pour pouvoir aussi comparer les résultats d'un code scalaire contre un code SIMD optimisé.

Exercice 1 Ensemble fractal de MandelBrot

Dans cet exercice, nous allons étudier l'ensemble fractal de Mandelbrot qui est défini comme l'ensemble c d'un plan complexe tel que la suite :

$$z_0 = 0 \tag{1}$$

$$z_{n+1} = z_n^2 + c \tag{2}$$

ne tende pas vers l'infini. Le module doit rester inférieur ou égal à deux.
En remplaçant z_n par le couple de réels (x_n, y_n) et c par (a,b) on obtient :

$$(x_0, y_0) = (0, 0) \tag{3}$$

$$x_{n+1} = x_n^2 - y_n^2 + a \tag{4}$$

$$y_{n+1} = 2 * x_n y_n + b \tag{5}$$

$$|z_n| = \sqrt{x_n^2 + y_n^2} \tag{6}$$

1. Complétez le code de la fonction mandelbrot_scalar qui prend en entrée a, b et max_iter et qui calcule l'ensemble de mandelbrot tel que décrit précédemment (3-6) et cela tant que le module reste inférieur à 2 (6) et que le numéro de l'itération est inférieur à max_iter.

2. Faites en de même pour la fonction mandelbrot_simd et mesurez les temps donné par benchmandelbrot pour des tailles de 200 et 1200 ainsi que l'accélération obtenue en combinant les différents niveaux d'optimisations. Optimisez le plus possible vos fonctions scalaire et simd et expliquez vos

optimisations.

3. Pour améliorer la vitesse de calcul, ajoutez un pragma OpenMP dans la fonction calc_mandelbrot_scalar et calc_mandelbrot_simd. Optimisez l'utilisation de OpenMP en détaillant l'ordonnancement comme vu en cours. Faites un tableau comparatif entre les différentes versions simd, scalaire et openmp. Quelle est l'accélération maximale théorique que vous pouvez obtenir sur votre machine en utilisant OpenMP et simd. Vous pouvez obtenir toutes les informations sur votre processeur en utilisant la commande suivante :
cat /proc/cpuinfo

4. Il est possible de pipeliner les calculs flottants en simd avec des calculs sur des entiers. Comment pouvez-vous utiliser cette information pour optimiser encore plus votre code simd ? Expliquez juste comment faire.

Exercice 2 Produit scalaire et calcul de filtres en OpenMP

1. Trouver un moyen d'appliquer OpenMP au produit scalaire en simd et notez l'accélération que vous obtenez. Pouvez-vous expliquer la différence entre le speedup théorique maximal et celui que vous obtenez?

2. Pouvez-vous utiliser OpenMP pour le calcul de filtre ? Comment feriez-vous pour gérer les bords dans le calcul de filtre ? Il n'est pas nécessaire de le coder expliquez juste votre raisonnement.