UPMC

LIP6

HPC

Optimisation SIMD pour le TSI

Lionel Lacassagne

Introduction

Le but de ce TP est de manipuler les instructions SIMD flottantes pour implémenter les algorithmes de base en TSI. Pour chaque codage, donnez, dans un tableau récapitulatif, les spécifications de votre code, Ã savoir la complexité arithmétique (`MUL`+`ADD`), le nombre d'accès mémoire (`LOAD`+`STORE`) ainsi que le nombre de copies registre à registre (`MOVE`) et l'intensité arithmétique (le ratio entre la complexité arithmétique et le nombre d'accès mémoire).
Conseil: vous êtes *vivement* encouragés à écrire des macros pour les calculs et manipulations répétitifs comme par exemple:

- `vec_add3` et `vec_add5` qui serviront en 1D et 2D,
- `vec_left1` et `vec_right1` pour les opérateurs de taille 3×1 et 3×3,
- `vec_left2` et `vec_right2` pour les opérateurs de taille 5×1 et 5×5.

1 Opérateurs 1D

1.1 Addition de deux vecteurs (tableaux 1D) de registres SIMD

Soient les vecteurs $\overrightarrow{X_1}$ et $\overrightarrow{X_2}$ de taille n et$\overrightarrow{Y}$ leur somme:

$$\overrightarrow{Y} = \overrightarrow{X_1} + \overrightarrow{X_2} = \sum_{i=0}^{n-1} X_1(i) + X_2(i) \tag{1}$$

1. Codez cette addition SIMD dans la fonction `add_vf32vector()`.
   Afin de séparer les instructions de calcul des instructions d'accès mémoire, utilisez les registres SIMD x_1, x_2 et y tels que $x_1 = X_1(i)$, $x_2 = X_2(i)$ et $y = Y(i)$.
2. Cet exercice est déjà fait et sert d'exemple complet pour les suivants en explicitant le principe de lectures-calculs-écritures.

1.2 Produit scalaire de deux vecteurs (tableaux 1D) de registres SIMD: réduction totale

Soient les vecteurs $\overrightarrow{X}$ et $\overrightarrow{Y}$ de taille n et d leur produit:

$$d = \overrightarrow{X_1} \cdot \overrightarrow{X_2} = \sum_{i=0}^{n-1} X_1(i) \times X_2(i) \tag{2}$$

1. Codez ce produit scalaire SIMD dans la fonction `dot_vf32vector`. Que faut il faire en fin de traitement pour que le produit scalaire complet (la somme de **tous** les produits) soit dans chacun des quatre blocs du registre SIMD ? Proposez une solution pour les architectures SSSE3 puis pour SSE4.1.

1.3 Moyenne sur un voisinage 1D de registres SIMD: réduction partielle

Soient les opÃ©rateurs de moyennage 1D A_3 et A_5:

$$A_3 = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{3}$$

$$A_5 = \frac{1}{5}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{4}$$

1. Codez les opÃ©rateurs de moyennage A_3 et A_5 en SIMD dans les fonctions `avg3_vf32vector` et `avg5_vf32vector`.
2. Codez les versions avec rotation de registres dans les fonctions `avg3_vf32vector` et `avg5_vf32vector`.
3. Faire un tableau rÃ©capitulatif et commentez.

2 Opérateurs 2D

2.1 Moyenne sur un voisinage 2D de registres SIMD: réduction partielle

Soient les opérateurs de moyennage 2D, M_3 et M_5

$$M_3 = \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} * \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{5}$$

$$M_5 = \frac{1}{5}\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} * \frac{1}{5}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix} = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{6}$$

1. Codez les opérateurs de moyennage M_3 et M_5 en SIMD dans les fonctions `avg3_reg_vf32matrix` et `avg5_reg_vf32matrix`.
2. Codez les versions avec rotation de registres dans les fonctions `avg3_rot_vf32matrix` et `avg5_rot_vf32matrix`.
3. Codez les versions avec réduction par colonne dans les fonctions `avg3_red_vf32matrix` et `avg5_red_vf32matrix`.
4. Faire un tableau récapitulatif des performances en *cpp* pour différentes valeurs de n et commentez.