TD #1: OpenMP – Manipulation de Matrices et de Vecteurs

Jonathan Rouzaud-Cornabas (jonathan.rouzaud-cornabas@insa-lyon.fr)

Pour chaque sous partie (vecteur-vecteur, matrice-matrice, compter les mots), il est nécessaire en plus d'avoir un code fonctionnel de fournir les performances en passage à l'échelle fort et faible. Vous pouvez utiliser les timer haute précision (sous partie chrono de la STL) pour calculer les performances.

Passage à l'échelle fort En passage à l'échelle fort, on garde la taille du problème (taille des vecteurs et des matrices) et on augmente le nombre de processeurs. Le minimum est de tester sur 1, 2 et 4 coeurs dans votre cas. Si possible, aller plus loin.

Passage à l'échelle faible En passage à l'échelle faible, on augmente la taille du problème avec l'augmentation du nombre de coeurs. En pratique, vous multipliez la taille du problème par 2 pour 2 coeurs et 4 pour 4 coeurs.

1 Exercice

1.1 Vecteur

Question 1.1 Ecrivez le code séquentiel pour remplir les vecteurs (des nombres double généré en aléatoire)

Question 1.2 Ecrivez le code permettant d'afficher les résultats

Question 1.3 Ecrivez le code séquentiel pour additioner 2 vecteurs

Question 1.4 Ecrivez le code permettant de sommer toutes les cases d'un vecteur

Question 1.5 Modifier votre code pour pouvoir spécifier le nombre de coeurs à utiliser et la taille des vecteurs

Question 1.6 Paralléliser votre code avec OpenMP

Question 1.7 Utiliser le pragma reduction

Question 1.8 Ecrivez la multiplication d'un vecteur par un double

Question 1.9 Faite une évaluation de performance (passage à l'échelle fort et faible) de:

- Addition de 2 vecteurs
- Somme d'un vecteur
- Multiplication d'un vecteur par un double

1.2 Matrice

Question 1.10 Rejouer les questions précédentes mais pour des manipulations de matrices.

Question 1.11 (Optionnel) Ecrivez la multiplication d'un vecteur par une matrice

Question 1.12 (Optionnel) Ecrivez la multiplication de deux matrices

2 Compter des lettres

Pour information, le fonctionnement de l'algorithme sous jacent est assez proche de MapReduce World-Count. Le but est de compter le nombre d'occurences de chaque lettre présent dans une matrice.

Question 2.1 Ecrivez le code séquentiel permettant d'initialiser une matrice avec un ensemble aléatoire de lettre (minuscule a-z).

Question 2.2 Ecrivez le code permettant d'afficher le résultat.

Question 2.3 Ecrivez le code séquentielle permettant de faire le calcul.

2.1 Parallel for

Question 2.4 Ecrivez le code paralléle permettant de faire le calcul sur le problème complet.

Question 2.5 Proposer un code de découpage du problème en sous problème.

Question 2.6 Proposer une adaptation de votre code parallèle pour calculer un sous problème.

Question 2.7 Proposer un code permettant de sommer le résultat des sous-problème.

2.2 Task (Optionnel)

Question 2.8 Récrivez votre code de calcul du problème complet avec tâches et plus des boucles for.

Question 2.9 Réécrivez votre code de découpage, calcul de sous problèmes et de leurs sommes avec des tâches (pensez aux dépendances).