Accélérateurs de calculs : exercices et TP

Sylvain HENRY (sylvain.henry@labri.fr)

Version 1.0 - 12 novembre 2012

1 Multiplication de matrices

1. Écrire le ou les noyaux de calculs OpenCL permettant d'effectuer une multiplication de deux matrices de Float. Pensez à indiquer pour chacune des invocations de kernels :
   - les dimensions de l'espace d'indices global
   - les dimensions des blocs
   - les contraintes sur les dimensions des matrices d'entrées
2. Utilisez la mémoire locale pour optimiser votre code, si ce n'est pas déjà fait.

2 Factorisation de Cholesky

La factorisation de Cholesky consiste, pour une matrice symétrique définie positive A de dimension $n \times n$, à déterminer une matrice triangulaire inférieure L telle que : $A = LL^T$.

Pour i allant de 0 à $n-1$, la $i^{ème}$ étape du calcul s'effectue ainsi :

- $a_{ii} = \sqrt{a_{ii}}$
- Pour $i < j < n$, $a_{ij} = a_{ij}/a_{ii}$
- Pour $i < j < n$ et $i < k \leq j$, $a_{kj} = a_{kj} - a_{ik} \times a_{ij}$

Dans cet algorithme, on modifie la matrice A sur place et le résultat L est stocké dans la partie inférieure gauche de la matrice A à la fin de l'algorithme.

1. Écrire un kernel OpenCL permettant d'effectuer cet algorithme.
2. Réfléchir à des optimisations au sein de ce kernel.
3. On constate que cet algorithme est assez séquentiel (boucle sur i). Comment faire pour améliorer le parallélisme en utilisant plusieurs kernels ? Indice : on cherche à réduire le temps de calcul du chemin critique (calcul des éléments de la diagonale).
4. Écrire les kernels correspondant à cette autre solution.
5. Décrire le code hôte (en pseudo code) permettant l'enchaînement de ces kernels.
6. On constate que les performances des kernels séquentiels sont très mauvaises et qu'il est plus efficace de les effectuer sur le CPU pendant que le GPU calcule autre chose.
   (a) Décrire en pseudo code le code hôte correspondant à cette solution (qui est celle utilisée actuellement dans les codes les plus performants).

3 Travaux pratiques

1. Implémenter la factorisation de Cholesky cette fois-ci avec le code hôte.
   (a) Commencez par les versions les plus simples avant de passer aux autres versions
   (b) Testez vos résultats. La matrice de Hilbert caractérisée par $h_{ij} = \frac{1}{i+j-1}$ est symétrique définie positive.
2. Réfléchir à l'utilisation simultanée de plusieurs accélérateurs, aux transferts de données induits et à la répartition de la charge.
   (a) Considérer ensuite le cas où les plateformes OpenCL utilisées sont différentes
3. Bonus : décrire puis implémenter une couche d'abstraction entre l'application et OpenCL qui effectue automatiquement les transferts de données et le placement des kernels sur les accélérateurs.

4 Évaluation

L'évaluation portera sur le TP. Il vous est demandé :

1. de rédiger un rapport synthétique
   (a) Pour chaque version implémentée
      i. principes de l'algorithme
      ii. validation des résultats (tests)
      iii. performances obtenues
   (b) Rédaction de la question 2
   (c) Rédaction de la question 3, le cas échéant
2. de rendre les différentes versions des codes