Université de Bordeaux

Programmation Multicœur et GPU

Simulation de particules sur architectures parallèles

Auteurs :
Salmane Bah
Guillaume Thibault

Référents :
Raymond Namyst
Pierre-André Wacrenier

9 mai 2014

Résumé

Ce rapport présente des mesures de performance portant sur une simulation d'atomes réalisée dans le cadre du projet de l'unité d'enseignement Programmation Multicœur et GPU. Dans cette partie, nous étudions la parallélisation de la simulation à l'aide de noyaux OpenCL (Open Computing Language) qui permettent de calculer les interactions entre les atomes sur des accélérateurs.

1 Mise en place de la simulation

1.1 Les noyaux *border_collision* et *gravity*

Avant de pouvoir tester la simulation il a fallu écrire les noyaux *border_collision*, qui gère le rebond des atomes contre les parois du domaine, et *gravity*, qui permet de gérer l'action de la gravité sur les atomes. Nous avons pu le faire en reprennant la structure des fonctions similaires que nous avions définies dans le fichier *openmp.c*.

1.2 Le noyau *lennard_jones*

Dans un premier temps, nous avons implementé une version sans memoire locale partagée pour les workgroups. Après avoir verifié son bon fonctionnement, nous avons ensuite implementé une seconde version avec une stratégie de cache s'appuyant sur les workgroups.

D'après les tests effectués, le calcul des forces semble correct, cependant, on observe la disparition de certains atomes à interval régulier sans avoir pu en cibler l'origine.

Nous avons choisi de laisser les deux implémentations du noyau dans le code afin de pouvoir passer de l'une à l'autre (en commentant l'une des deux) pour vérifier les résultats suivant le noyau choisi.

2 Méthodes d'expérimentation

Pour mettre en avant l'évolution des performances en fonction du nombre de workgroup utilisé, nous avont fait varier la simulation sur ce dernier, il est donc difficile de les comparer rigoureusement avec les simulations précédentes qui elles variaient sur le nombre de threads utilisés.

Afin d'automatiser la simulation, nous avons choisi de rajouter une option de lancement du programme (*-w <nb_workgroup>*).

Nous avons constaté qu'il n'était pas possible de créer plus de 128 workgroups sous peine de lever l'erreur Opencl suivante : «**(-54) : CL_INVALID_WORK_GROUP_SIZE**».

2.1 Variation du nombre d'atomes et du type de machine

Les courbes

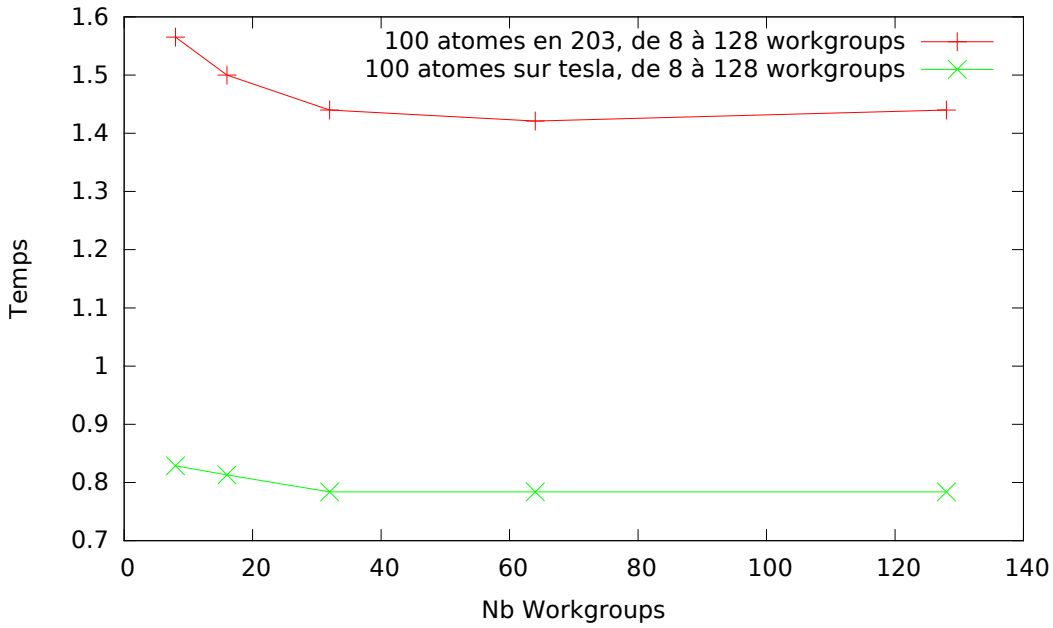


FIGURE 1 – Comparaison de la simulation avec 100 atomes en 203 et sur tesla en variant le nombre de workgroups

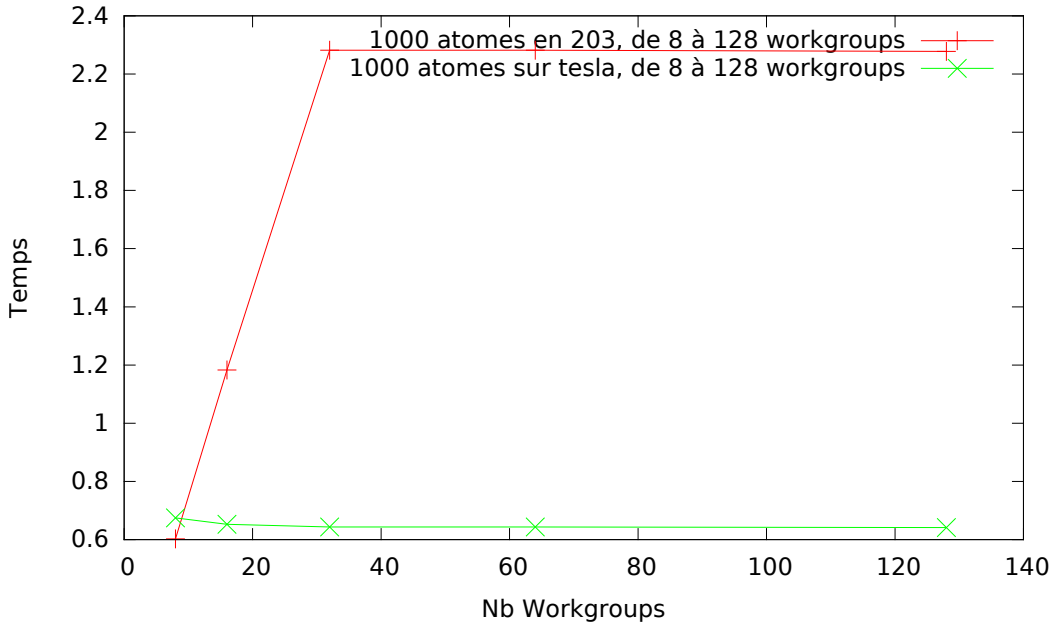


FIGURE 2 – Comparaison de la simulation avec 1000 atomes en 203 et sur tesla en variant le nombre de workgroups

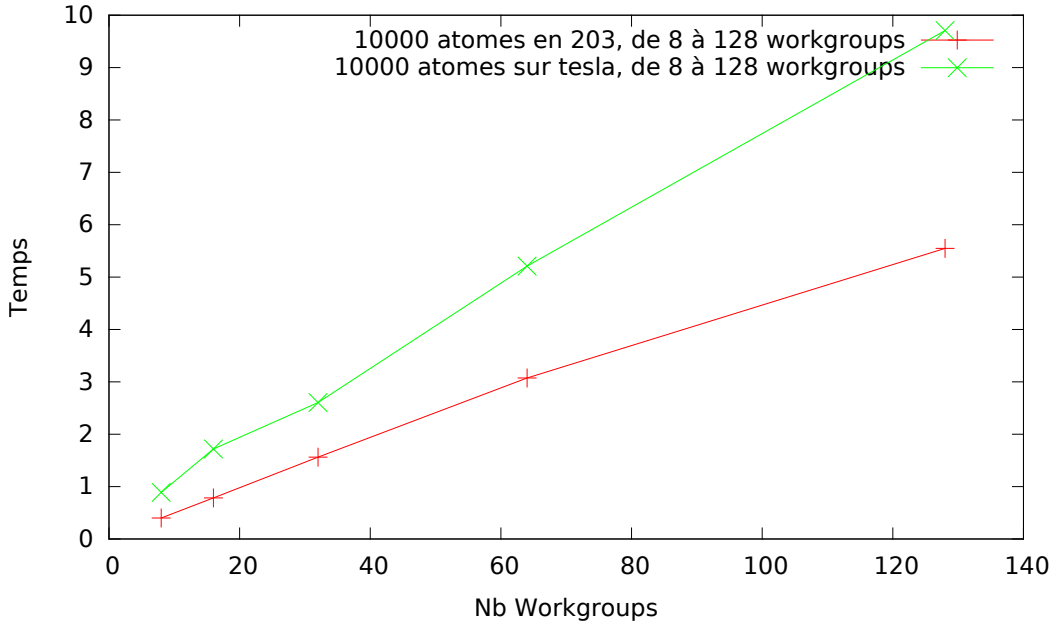


FIGURE 3 – Comparaison de la simulation avec 10000 atomes en 203 et sur tesla en variant le nombre de workgroups

Commentaire

Avec 100 et 1000 atomes pour 100 itérations, la différence entre le temps d'exécution de chaque simulation est trop faible, les courbes sont donc inexploitables (la limite d'accélération est immédiatement atteint, c.f loi d'Amdahl).

Pour 10000 atomes, on observe que l'accélération de la simulation est proportionnelle au nombre de workgroup utilisé, cette observation est encore plus marquée sur le serveur Tesla.

2.2 Variation sur le domaine initial

Les courbes

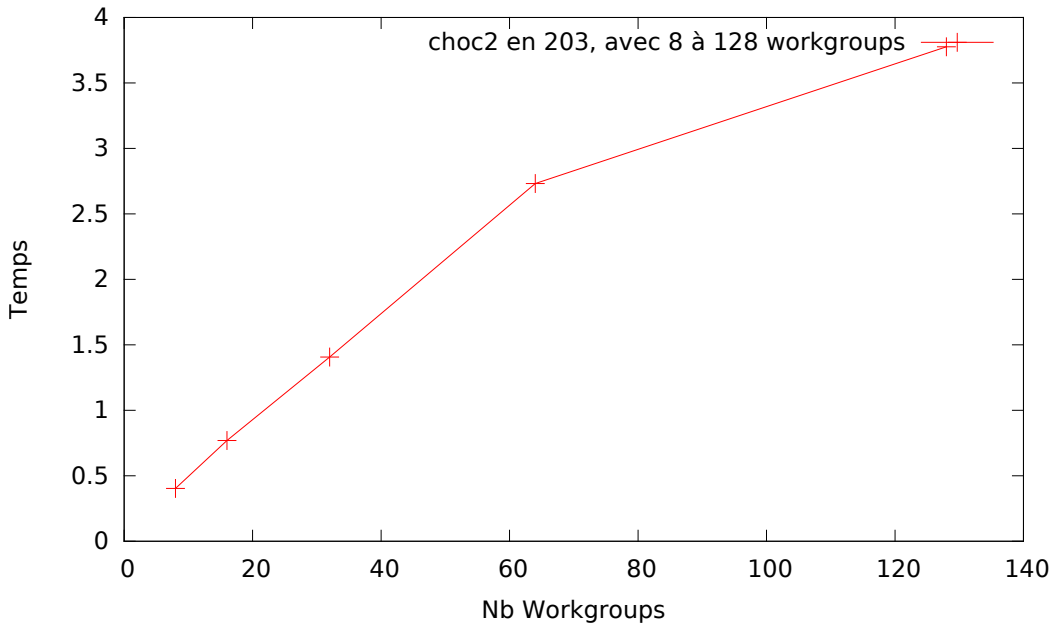


FIGURE 4 – Simulation avec choc2 en OpenCL

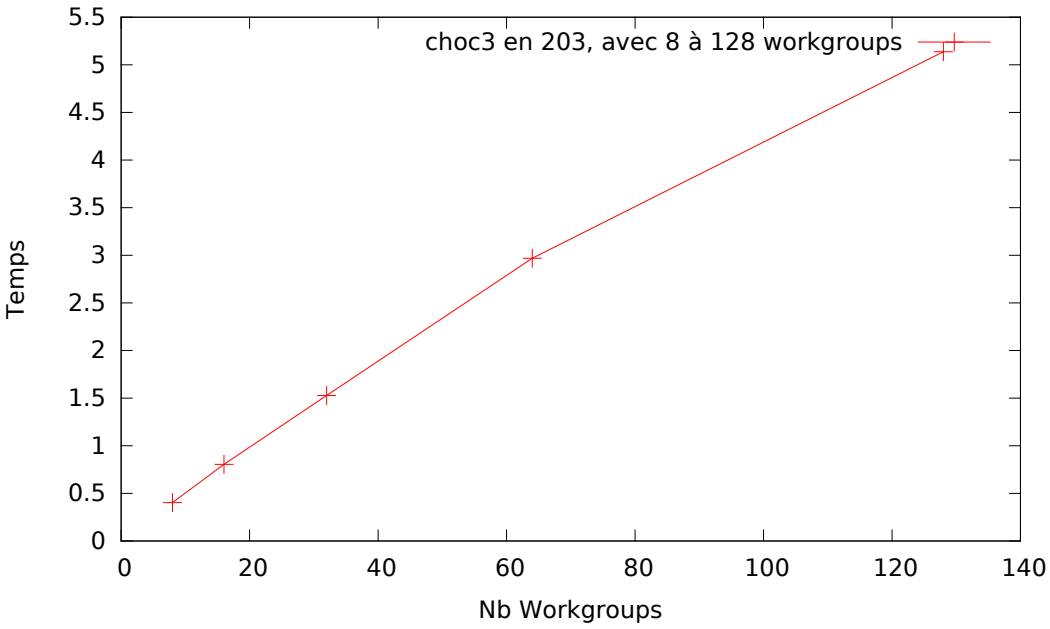


FIGURE 5 – Simulation avec choc3 en OpenCL

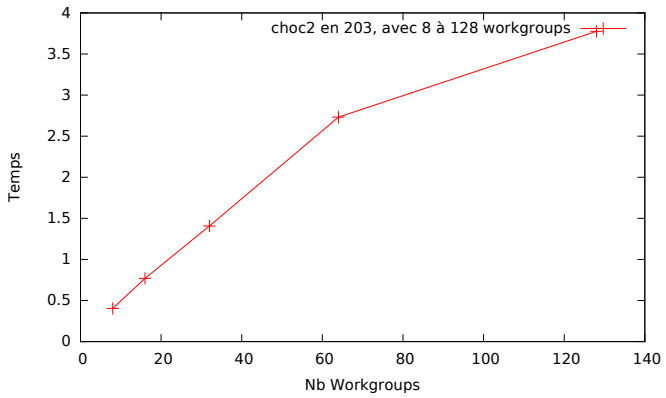
Commentaire

Comme pour la simulation lancée avec 10000 atomes et 100 iterations, on constate que les simulations lancées avec choc2 et choc3 comme fichier de configuration ont une accélération presque proportionnelle au nombre de workgroups utilisés.

2.2.1 Comparaison avec le tri par boite

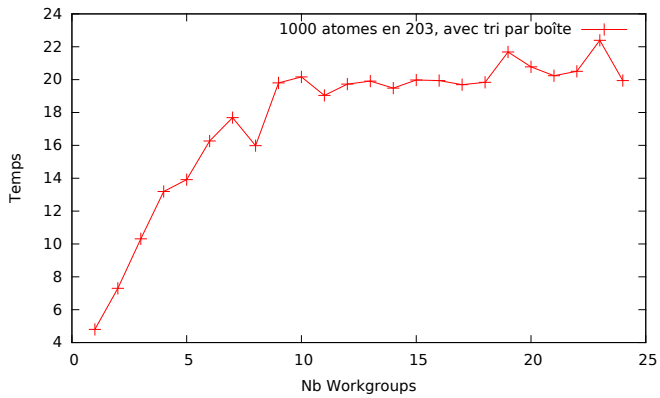
Les courbes

(a) Choc2 avec tri par boite

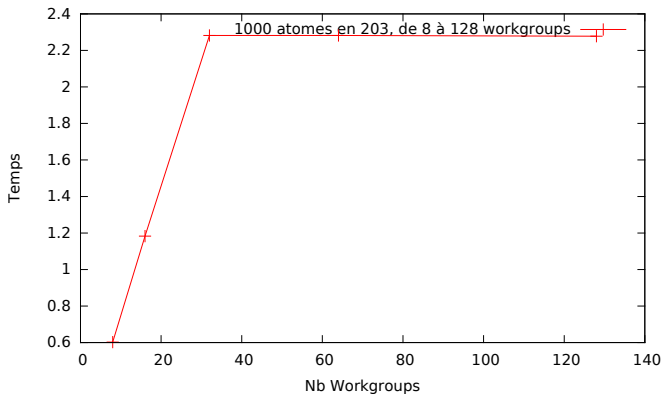


(b) Choc2 en OpenCL

FIGURE 6 – Comparaison de la simulation avec choc2



(a) 1000 atomes avec tri par boite



(b) 1000 atomes en OpenCL

FIGURE 7 – Comparaison de la simulation avec 1000 atomes

Commentaire

Les figures ci-dessus juxtaposent la version de la simulation realisée en OpenMP (tri par boite) et en OpenCL.

Il est difficile d'en tirer des conclusions étant donné que les paramètres d'expérimentation utilisés sont différents. L'axe des abscisses représentant *OMP_NUM_THREADS* pour l'un et la taille des workgroups pour l'autre.

2.3 Comparaison des noyaux lennard_jones

Courbe

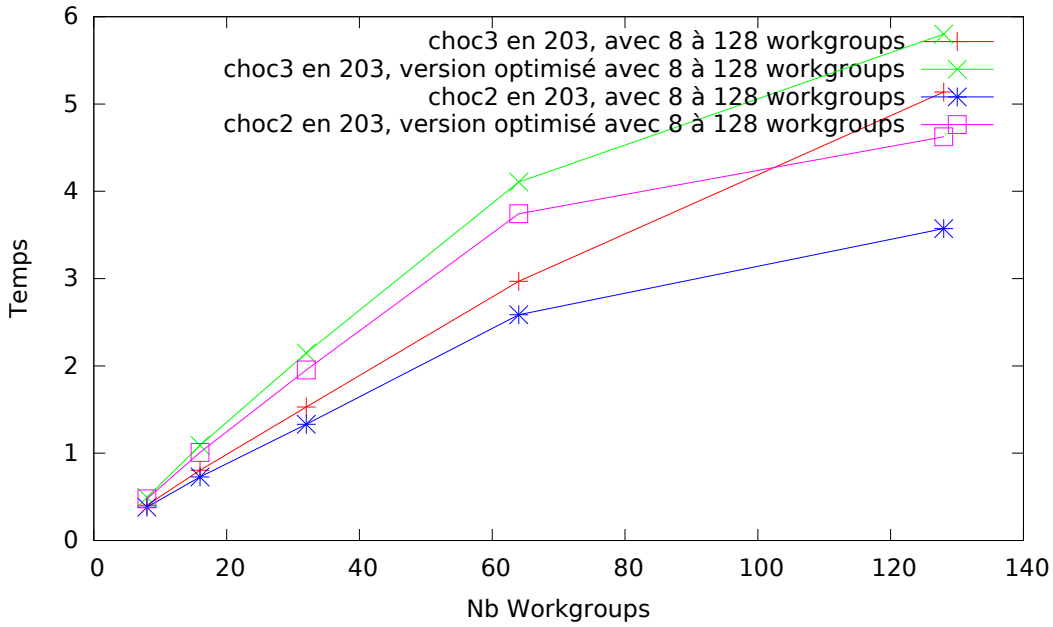


FIGURE 8 – Simulation avec choc3 en OpenCL

Commentaire

Sur cette figure, on observe clairement que les simulations réalisées avec la version du noyau lennard_jones optimisée présente de meilleurs accélérations, en effet les accès non coalescés en mémoire globale sont réduit par la gestion du cache.

3 Conclusion

La version OpenCL, même pour une complexité d'exécution $\mathcal{O}(n^2)$ du calcul des force, rivalise en performance avec les versions OpenMP triées.

Ce projet nous permis de découvrir différentes manières de paralléliser un programme, de manière simple et efficace en utilisant OpenMP ou alors de manière plus complexe et plus performante a l'aide d'OpenCL.