IT341
Accélérateurs de calcul
19 janvier 2016

Partie 1 – 40 points

1) Définitions (30 points)

Donnez une description brève et rapide des termes suivants : (6 points / réponse)

- Réutilisation de données (Data reuse)
- Modèle de programmation par tâche
- SIMT
- Barrière de synchronisation
- Condition de concurrence (race condition)

2) Exécution d'un WARP (5 points)

Décrivez brièvement la notion de WARP dans le cas « d'accélérateurs graphiques »

3) Hiérarchie mémoire (5 points)

Décrivez brièvement l'architecture mémoire dans une carte accélératrice en prenant soin de préciser (dans le modèle de programmation) qui peut accéder à cette mémoire.

Partie 2 – 60 points

1) cudaMallocAndMemcpy (10 points)

5 étapes à remplir :

1. allouer la mémoire pour les pointeurs du device ciblé
2. copier de la mémoire du host vers le device (h_a vers d_a)
3. faire une copie mémoire device vers device (d_a vers d_b)
4. faire une recopie vers le host depuis le device (d_b vers h_a)
5. libérer la mémoire

2) KernelBase (20 points)

5 étapes à remplir :

1. allouer la mémoire pour d_a
2. définir une grille monodimensionnelle , des blocs de threads monodimensionnels et lancer le noyau de calcul
3. effectuer l'opération suivante pour chaque thread :
   - d_a[idx] = 200 * blocIdx.x + threadIdx.x
   - avec idx à défnir (sachant que idx est l'indice global)
4. recopier d_a vers h_a
5. verifier les résultats

3) Inversion de structure (30 points)

L'objectif de chacun des programmes de cette partie est d'inverser une liste d'entrée $\{a_0,a_1,....,a_{n-1}\}$ (pointeur d_a) en $\{ a_{n-1},...a_1, a_0 \}$ (pointeur d_b)

3.1) inversion simple block

Dans ce cas, l'inversion n'est à effectuer que sur un seul bloc de threads pour inverser un bloc de données de taille égale au nombre de threads, égale à 256

soit donc : N = numThreads = 256

3.2) inversion multi blocks

Le cas multiblock s'appuie sur l'implémentation précédente.
L'idée est d'excécuter un noyau de calcul à travers plusieurs blocks de threads. Ce nombre de blocks de threads doit simplement être un multiple de 256 afin de pouvoir inverser un tableau de taille N à travers N / 256 blocks

soit donc : N est un multiple de 256

3.3) inversion multi blocks rapide

Le cas multiblock_rapide demande maintenant de reprendre l'implémentation précédente est d'utiliser la mémoire partagée afin de tirer partie des accès rapides sur la mémoire.