Faculté des Sciences & Techniques — Université de Limoges

Master 1ère année

Dév. GPGPU

Examen — décembre 2023

Durée : 1h30 — Documents autorisés

1 – Parallélisation de la somme des préfixes ou opération « *SCAN* »

10pts

Soit un tableau de valeurs $T = [a_0, a_a, ..., a_{n-1}]$ et une opération associative $\oplus$ avec une identité I alors $\text{scan}(T) = [I, a_0, (a_0 \oplus a_1), ..., (a_0 \oplus a_1 \oplus ... \oplus a_{n-2})]$

Exemple :

- ▷ $\oplus$est l'addition ;
- ▷ I est la valeur zéro ;
- ▷ l'ensemble est $T = [3, 1, 7, 0, 4, 1, 6, 3]$;

$\Longrightarrow \text{scan}(T) = [0, 3, 4, 11, 11, 15, 16, 22]$

Questions :

a. Qu'est-ce que permet de faire une opération « *associative* » dans le cadre de la parallélisation ? Quelle type de complexité algorithmique peut-on envisager atteindre pour la parallélisation ? *(1pt)*

b. Proposez une méthode de parallélisation générique n'utilisant pas la mémoire partagée. *(2pts)*

c. Est-ce que les accès mémoire seront optimaux ? Pourquoi ? *(1pt)*

d. Écrivez un programme utilisant votre méthode sans utiliser la mémoire partagée. *(3pts)*
Vous donnerez uniquement le code du kernel et l'appel de ce kernel.

e. Écrivez un programme utilisant la mémoire partagée. *(3pts)*
Vous donnerez uniquement le code du kernel et l'appel de ce kernel.

2 – 10pts

On voudrait optimiser le **chargement de camion**, on essayant **toutes les combinaisons possibles**, d'un ensemble de cartons numérotés :

carton	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...	N
Poids	80	39	61	111	65	104	81	104	95	96	54	50	104	81	...	92

Il est possible d'exprimer le **chargement d'un camion** en indiquant si un carton et ou non chargé, à l'aide d'une valeur binaire associée :

- ▷ $1 \Longrightarrow$ chargé ;
- ▷ $0 \Longrightarrow$ non chargé ;

carton	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...	N
chargé	0	1	1	1	1	0	0	0	0	1	1	0	0	0	...	1

Une fois un chargement défini, il est possible :

- ▷ de calculer le **poids total** de ce chargement ;
- ▷ de **valider** ce chargement s'il est **inférieur à un seuil** : $\text{seuil} = \frac{\text{chargement max}}{2}$

Questions :

a. Combien de **combinaisons possibles** de chargement peut-on faire s'il y a 10 cartons ? *(1pt)*

b. Comment **créer** toutes les combinaisons possibles de chargement ? *(1pt)*

c. Proposez une **méthode de parallélisation** permettant d'obtenir tous les chargements acceptables :
- ◇ Comment allez vous définir vos structures de données et votre traitement dans le cas où l'on veut traiter $N = 1000$ cartons ? *(1pt)*
- ◇ Combien de mémoire allez vous consommer ? *(1pt)*
- ◇ Est-ce possible de tirer partie de la **mémoire partagée** ? Est-ce intéressant ? *(1pt)*

d. Dans votre solution parallèle, est-il possible **d'arrêter le traitement** d'un chargement dès qu'il dépasse le seuil (éviter le travail inutile) ? *(1pt)*

e. Donnez le code du ou des kernels, ainsi que leur appels pour obtenir l'ensemble des **chargements acceptables**. *(4pts)*