

Durée : 1h30 — Documents autorisés

1– Soit le traitement suivant :

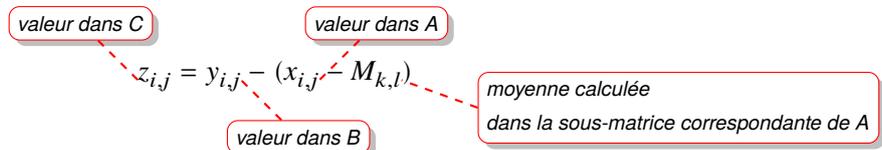
- 10pts** ❶ on dispose de deux tableaux A et B de dimensions 1024x2048 ;
- ❷ chaque tableau contient des mesures de températures obtenues dans le cadre d'une expérimentation dans la physique des plasmas ;
- ❸ on découpe le tableau A en sous-matrices de 8x8, de rang k, l dans A :

$x_{0,0}$	$x_{0,1}$	$x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$	$x_{0,6}$	$x_{0,7}$
$x_{1,0}$	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$	$x_{1,6}$	$x_{1,7}$
$x_{2,0}$	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	$x_{2,5}$	$x_{2,6}$	$x_{2,7}$
$x_{3,0}$	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$	$x_{3,5}$	$x_{3,6}$	$x_{3,7}$
$x_{4,0}$	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$	$x_{4,5}$	$x_{4,6}$	$x_{4,7}$
$x_{5,0}$	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$	$x_{5,5}$	$x_{5,6}$	$x_{5,7}$
$x_{6,0}$	$x_{6,1}$	$x_{6,2}$	$x_{6,3}$	$x_{6,4}$	$x_{6,5}$	$x_{6,6}$	$x_{6,7}$
$x_{7,0}$	$x_{7,1}$	$x_{7,2}$	$x_{7,3}$	$x_{7,4}$	$x_{7,5}$	$x_{7,6}$	$x_{7,7}$

où :

- ◊ k varie de 0 à 127 ;
- ◊ l varie de 0 à 255 ;

- ❹ pour chaque sous-matrice de A, on calcule la moyenne $M_{k,l} = \frac{\sum_{a=0}^7 \sum_{b=0}^7 x_{a,b}}{64}$;
- ❺ on remplit le tableau **résultat** C de dimensions 1024x2048 de la manière suivante :
- ◊ chaque valeur $z_{i,j}$ de C est égale à :



Où i varie de 0 à 1023 et j varie de 0 à 2047.

Questions :

- a. Combien de **moyennes de sous-matrice** de A va-t-on calculer ? (1pt)
- b. Donner une **association grille, bloc, thread** pour le traitement de ce problème. (1pt)
- c. Pour le calcul de $M_{k,l}$, peut-on utiliser de la **mémoire partagée** ? (2pts)
Est-ce **intéressant** ?
Comment faire ?
- d. Donner le **code CUDA** réalisant le traitement demandé. (4pts)
- e. Si on veut traiter un nouveau tableau B sans changer le tableau A, comment peut-on le faire de **manière optimale** ? (2pts)
Vous donnerez les modifications à apporter à votre code.



2– Soit le code suivant :

10pts

```
1 #include <stdio.h>
2 #include <cuda.h>
3
4 /* code supprimé */
5 __global__ void mon_noyau(float *t, int r)
6 {
7     int position = threadIdx.x + r;
8
9     if ((position%2) == 0)
10    {
11        t[position/2] = t[position] * t[position+1];
12    }
13 }
14 int main(void)
15 {
16     float *gpu_data;
17     int t = TAILLE;
18     /* code supprimé */
19     cudaMemcpy(gpu_data, data, 2*TAILLE*sizeof(float), cudaMemcpyHostToDevice);
20     while (t>1)
21     {
22         mon_noyau<<<1, TAILLE>>>(gpu_data, t);
23         t=t/2;
24     }
25     /* code supprimé */
26 }
```

En lignes 4, 18 et 25, du code a été supprimé

Questions :

- a. Détaillez et expliquez **ce que fait le code** de ce programme. (2pts)
- b. Est-ce que les **accès à la mémoire** sont optimaux ? (1pt)
Expliquez pourquoi.
- c. Est-ce que le programme est **facilement extensible** : peut-on augmenter facilement la constante TAILLE ? (1pt)
- d. Est-il possible de mettre le travail des lignes 20 à 24 **directement dans le noyau** ? (2pts)
Donnez le code correspondant.
- e. Donnez un programme CUDA **plus efficace** réalisant le même travail. (4pts)