Práctico 3 del curso GPGPU: Grillas 2D, 3D y acceso coalesced

El objetivo de este práctico es dominar el direccionamiento de datos en grillas 2D y 3D y evaluar el desempeño de distintos patrones de acceso a la memoria global de la GPU. Los datos 2D y 3D se almacenarán en arreglos lineales (double *). Si se cuenta con un arreglo que representa un cubo de lado 2, la asignación de hilos (*x,y,z*) al arreglo será siempre de la siguiente forma:

0	1	2	3	4	5	6	7
(0,0,0)	(1,0,0)	(0,1,0)	(1,1,0)	(0,0,1)	(1,0,1)	(0,1,1)	(1,1,1)

Es decir, se almacenan los distintos planos en orden creciente de *z* y, dentro de cada plano, se almacenan las filas en orden creciente de coordenada *y.* En los siguientes ejercicios, cada hilo deberá realizar los cálculos correspondientes para determinar la posición del arreglo sobre la cual debe trabajar.

Ejercicio1

Considere un par de ejes cartesianos *x* e *y*, y un cuadrado de lado $2\pi=6.28318530718$ centrado en el origen. Divida el cuadrado en una grilla de puntos equiespaciados (x_i,y_i) tal que $h=x_i-x_{i+1}=y_i-y_{i+1}=10^{-3}$. Luego construya una función que genere una matriz cuyas entradas sean el valor de la función *f(x,y)=sen(x+y)* evaluada en cada punto del dominio definido anteriormente.

Ejercicio2

Ahora considere tres ejes cartesianos *x*, *y*, z, y un cubo de lado $2\pi=6.28318530718$ centrado en el origen. Divida el cubo en una grilla de puntos equiespaciados (x_i,y_i,z_i) pero con $h=10^{-2}$. Luego construya una función que genere un cubo (expresado como un arreglo lineal) cuyas entradas sean el valor de la función *f(x,y,z)=tan(x+y+z)* evaluada en cada punto del dominio definido anteriormente.

Ejercicio3

a) Desarrolle un kernel que dada una grilla 3D como la del ejercicio 2 devuelva una matriz *S* tal que

$$S(y,x)=\sum_{z=-\pi}^{\pi} f(x,y,z) \quad .$$

b) Desarrolle un kernel que dada una grilla 3D como la del ejercicio 2 devuelva una matriz *S* tal que

$$S(z,x)=\sum_{y=-\pi}^{\pi} f(x,y,z) \quad .$$

c) Desarrolle un kernel que dada una grilla 3D como la del ejercicio 2 devuelva una matriz *S* tal que

$$S(z,y)=\sum_{x=-\pi}^{\pi} f(x,y,z) \quad .$$

d) Registre el tiempo de ejecución de los tres kernels y explique las diferencias. Utilizando la herramienta nvprof registre también el valor de las métricas gld_efficiency y gst_efficiency. Investigue sobre su significado y utilícelo para justificar los resultados obtenidos.

Ejercicio4

En este ejercicio se evalúa un patrón de cómputo muy utilizado en la resolución numérica de ecuaciones diferenciales. En este patrón, cada elemento de la entrada se procesa de forma independiente, pero el elemento de salida se computa en función de un conjunto fijo de elementos vecinos llamado *stencil*.

a) Considere la matriz generada en el ejercicio 1. Desarrolle un kernel que genere una matriz de salida donde cada elemento siga la siguiente fórmula:

$$d_{\mathrm{out}}(x,y)=(-4\,d_{\mathrm{in}}(x,y)+d_{\mathrm{in}}(x+1,y)+d_{\mathrm{in}}(x-1,y)+d_{\mathrm{in}}(x,y+1)+d_{\mathrm{in}}(x,y-1))/h$$

Los valores que corresponden a posiciones fuera de la matriz valen 0.

b) Utilizando nvprof obtenga el valor de la métrica gld_transactions. Teniendo en cuenta los tamaños de la grilla y los accesos a memoria realizados por cada thread ¿Qué valor sería esperable para esta métrica? ¿Qué estrategia podría aplicarse para intentar disminuir esta cantidad?

Entrega

La entrega consiste en todo el código correspondiente a la solución de los ejercicios y un informe que contenga el análisis de los resultados experimentales y la respuesta a las preguntas planteadas.