UNIVERSIDAD DE ALICANTE

CUDA WORKSHOP 2018

PRÁCTICAS

Sergio Orts Escolano (sorts@ua.es)

Albert García García (agarcia@dtic.ua.es)

José García Rodríguez (jgarcia@dtic.ua.es)

PRÁCTICA 4: REDUCCIÓN PARALELA

La operación reducción es muy común y muy importante para el tratamiento paralelo de datos. Además es muy fácil de implementar en CUDA, aunque como veremos después no es tan fácil obtener grandes rendimientos.

Una de las formas de resolver el problema de la reducción en paralelo es la siguiente:

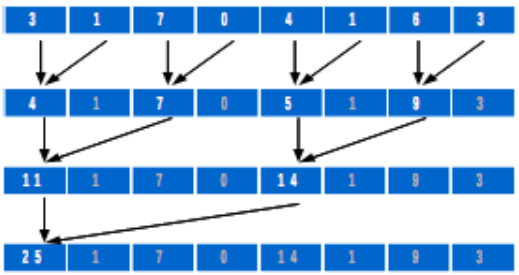

FIGURA 1. OPERACIÓN REDUCCIÓN (SUMA)

En este laboratorio aprenderemos:

- Como reservar y liberar memoria en la GPU.
- Como copiar datos desde CPU hacia la GPU y al contrario.
- Como escribir un programa GPU que haga la operación reducción suma
- Como medir los tiempos de ejecución y los conflictos de memoria compartida usando el CUDA profiler.

- Como diferentes esquemas de reducción pueden afectar a los conflictos del banco de memoria.

El código fuente se encuentra en la carpeta *cuReduction*. Inicialmente este código no compila, tendrás que completarlo para ello.
Paso 1: Completa la función *global_reduce_kernel_v0(…)* en el fichero *"vector_reduction.cu"*. Se tiene que implementar el esquema de reducción mostrado en la Figura 1. Anota el tiempo de ejecución obtenido utilizando este esquema.

Paso 2: Completa la función *global_reduce_kernel_v1(…)* en el fichero *"vector_reduction.cu"*. Se tiene que implementar el esquema de reducción mostrado en la Figura 2. Anota el tiempo de ejecución obtenido utilizando este esquema. ¿Por qué se obtiene un mayor rendimiento con este esquema de reducción? Puedes utilizar el profiler de NVIDIA para analizar ciertas métricas de rendimiento.

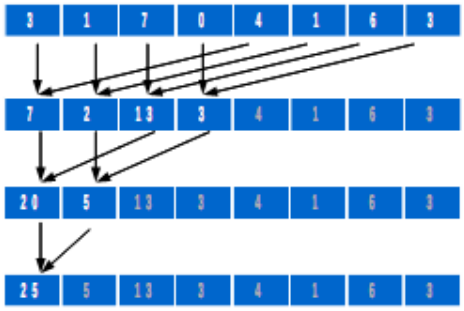

FIGURA 2. ESQUEMA DE REDUCCIÓN 2 (SUMA)

Paso 3: Una vez completado el esquema 2 prueba a mejorar su rendimiento haciendo uso de la memoria compartida disponible en la GPU. Para ello tendrás que declarar una variable de tipo *__shared__* dentro del Kernel. Implementa esta versión optimizada y anota los tiempos de ejecución. Para observar de forma más estable el rendimiento puedes ejecutar iterativamente el proceso de reducción y finalmente calcular una media, una forma de hacer esto sería introduciendo la invocación del Kernel en un bucle *for* y finalmente promediar todos los tiempos obtenidos.

Paso 4 (Opcional): Modifica el código anterior para calcular la operación búsqueda de valor máximo o mínimo en paralelo en lugar de la operación suma.