INF-351: Computación de Alto Desempeño
Laboratorio 4
Streams

Prof. Álvaro Salinas

20 de Julio de 2020

1. Descripción y Marco Teórico

En el siguiente laboratorio serán evaluados sus conocimientos sobre CUDA streams, sincronización y manejo de memoria global.

Problema

Consideremos la siguiente malla regular bidimensional:

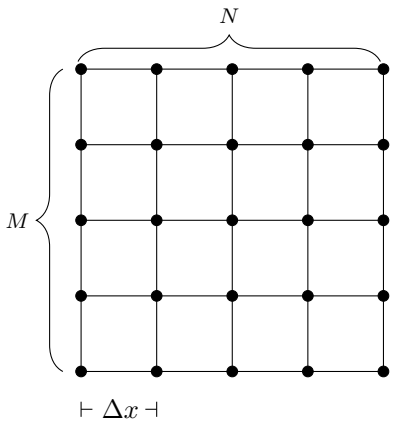

Figura 1: Malla.

Ésta es una malla de $M \times N$ nodos equidistantes a una distancia Δx respecto a cada uno de sus vecinos. Cada nodo tiene un valor $x_{j,i}(t)$ asociado, donde $0 \leq j \leq M-1$ corresponde a la fila del nodo, $0 \leq i \leq N-1$ corresponde a la columna del nodo, y $t \geq 0$ corresponde a un tiempo determinado. Consideraremos únicamente tiempos discretos con valores enteros, es decir, $t \in \{0, 1, 2, 3, 4, \dots\}$. teniendo en cuenta que valor inicial de cada nodo ($x_{j,i}(0)$) es conocido, implementaremos un método iterativo que actualice dichos valores según:

$$x_{j,i}(t+1) = \frac{x_{j,i+1}(t) - x_{j,i-1}(t)}{2\Delta x}$$

En otras palabras, cada nodo requiere únicamente del valor de sus vecinos horizontales (a la izquierda y a la derecha) para actualizar su valor. Considere condiciones periódicas, es decir, los nodos con $i = 0$ son los vecinos "a la derecha" de los nodos con $i = N-1$, y viceversa. De esta forma, solo necesitamos conocer los valores en el tiempo $t = 0$ para calcular todos los valores en el tiempo $t = 1$. Estos últimos serán utilizados para obtener los valores en $t = 2$, y así sucesivamente.

2. Desarrollo

Para la ejecución de su código, utilice un arreglo de flotantes de tamaño $M \times N = 10000 \times 10000$, inicializado con valores aleatorios entre 0 y 1. Su arreglo debe ser unidimensional, representando la malla bidimensional con sus filas concatenadas (la dimensión es realmente 100000000). Utilice $\Delta x = 0{,}001$.

1. Cree una función de CPU que actualice una vez los valores de todos los nodos de la malla. Utilice esta función para obtener el valor de todos los nodos en el tiempo $t = 10$.

2. De forma similar, implemente un CUDA kernel que actualice una vez los valores de todos los nodos de la malla. Utilice este kernel para obtener el valor de todos los nodos en el tiempo $t = 10$.

3. Utilice 4 CUDA streams para dividir la malla en 4 secciones de tamaño $M/4 \times N$, es decir, dividirla en 4 "franjas horizontales" con la misma altura. Implemente un CUDA kernel que actualice los valores de todos los nodos en una determinada submalla especificada por parámetro, y llame a este kernel desde cada uno de los 4 streams para procesar la totalidad de la malla. Nuevamente, obtenga los valores para el tiempo $t = 10$.

4. Finalmente, y de forma similar al paso anterior, utilice 4 CUDA streams para dividir la malla. Pero esta vez, divídala en "franjas verticales" de tamaño $M \times N/4$. Cree un kernel que actualice los valores de una submalla especificada por parámetro, y llame a dicho kernel desde los diferentes streams para procesar toda la malla. Al igual que en los puntos anteriores, obtenga los valores para el tiempo $t = 10$.

   [Informe] Presente una tabla o gráfico con los tiempos obtenidos para las cuatro implementaciones.

   [Pregunta] Comente sobre los tiempos observados y argumente sobre por qué cada implementación es más o menos eficiente que las demás.

   [Pregunta] ¿Qué desafíos enfrentó al realizar la última implementación? ¿Explique detalladamente cómo afrontó estos desafíos y por qué eligió dicha solución?

3. Reglas y Consideraciones

Entrega

- La entrega debe realizarse en un archivo de nombre Lab4-X.tar.gz (formatos rar y zip también son aceptados), donde X debe ser reemplazado por el número de su grupo. Diríjase a la inscripción de grupos para consultar su número.
- El archivo de entrega debe contener un informe en formato pdf junto con el código implementado para resolver el laboratorio. Se le ruega entregar un código ordenado.
- El informe debe contener:
  - Título y número del laboratorio.
  - Nombre y rol de todos los integrantes del grupo.
  - Modelo y compute capability de la tarjeta gráfica que fue utilizada para ejecutar el código. Si se probó con más de una tarjeta gráfica, incluya los datos de todas y específique qué tarjeta se utilizó en cada pregunta y resultado reportado.
  - Desarrollo. Preocúpese de incluir cada ítem señalado con los tag **[Pregunta]** e **[Informe]** en el enunciado.
  - Conclusiones. Incluya aprendizajes, comentarios, observaciones o supuestos que surgieron durante el desarrollo del laboratorio.
- El descuento por día de retraso es de 30 puntos, con un máximo de 1 día de retraso. No se aceptarán entregas posteriores.
- En caso de copia, los grupos involucrados serán evaluados con nota 0. No hay problema en generar discusión y compartir ideas de implementación con sus compañeros, pero códigos copiados y pegados no serán aceptados.
- El incumplimiento de estas reglas implica descuentos en su evaluación.
- La fecha de entrega es el día Miércoles 05 de Agosto. Se habilitará la opción de entrega en Aula.

Consideraciones

- Para mediciones de tiempo, utilice `cudaEvent()` para códigos de GPU (puede también utilizar la herramienta NVIDIA Visual Profiler si así lo desea). Trabaje con milisegundos $[ms]$.
- En caso de optar por la realización de gráficos, puede optar por la herramienta que más le acomode. La librería matplotlib de Python siempre es una buena opción.