Universidad de Santiago de Chile
Departamento de Ingeniería Informática
High Performance Computing

Profesor: Fernando Rannou
Estudiante: Javier Arredondo

Laboratorio 3 - SIMT: CUDA

1. Experiencia

El presente documento presenta los resultados de la tercera experiencia de la asignatura de Computación de Alto Rendimiento (*High Performance Computing*); donde se ha implementado la simulación del comportamiento de una onda bajo la Ley de Schroedinger, véase Ecuación 1.

$$H_{i,j}^{t} = 2H_{i,j}^{t-1} - H_{i,j}^{t-2} + c^2(\frac{dt}{dd})^2(H_{i+1,j}^{t-1} + H_{i-1,j}^{t-1} + H_{i,j+1}^{t-1} + H_{i,j-1}^{t-1} - 4H_{i,j}^{t-1}) \quad \forall t \geq 2 \tag{1}$$

Es importante mencionar, que existen casos especiales para $t = 1$ y $t = 0$, véase Ecuación 2 y 3.

$$H_{i,j}^{t} = H_{i,j}^{t-1} - \frac{c^2}{2}(\frac{dt}{dd})^2(H_{i+1,j}^{t-1} + H_{i-1,j}^{t-1} + H_{i,j+1}^{t-1} + H_{i,j-1}^{t-1} - 4H_{i,j}^{t-1}) \quad \forall t = 2 \tag{2}$$

$$H_{i,j}^{0} = 20 \quad 0.4N < i < 0.6N, 0.4N < j < 0.6N \tag{3}$$

Y $H_{i,j}^{0} = 0$ para el resto de las celdas de Ecuación 3. La condición de borde está dado por la Ecuación 4.

$$H_{0,j}^{t} = H_{i,0}^{t} = H_{N-1,j}^{t} = H_{i,N-1}^{t} = 0 \quad \forall i, j, t \tag{4}$$

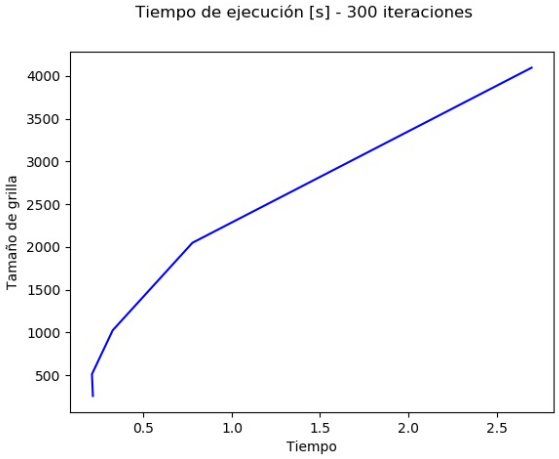
1.1. Objetivos

1. Calcular el tiempo de ejecución (*wall clock*) para diferentes tamaños de grilla.
2. Ocupancia de los *kernels* en distintos casos.
3. Tiempo de ejecución para diferentes tamaños de bloques.

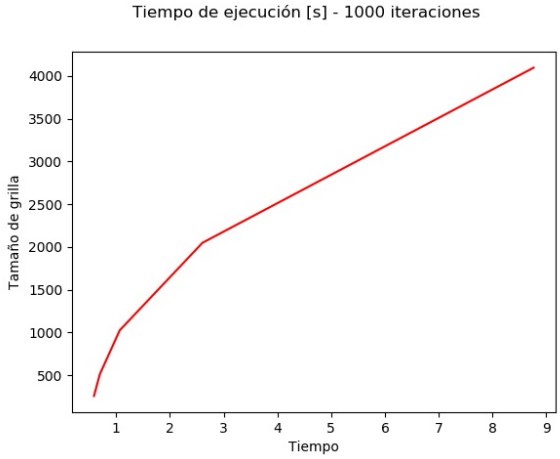
2. Resultados

2.1. Tiempos de ejecución

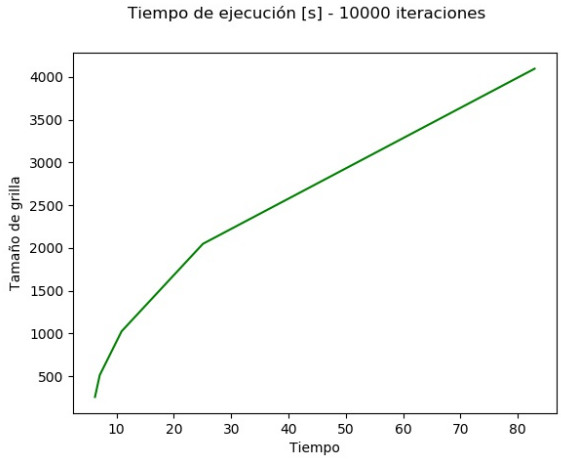
En primer lugar se ha procedido a calcular el tiempo de ejecución para diferentes tamaños de grilla: 512×512, 1024×1024, 2048×2048 y 4096×4096. Para esto se configuro el tamaño de bloque como $x = 16$ e $y = 16$, el número de iteraciones se fijó para $T = \{300, 1000, 10000\}$. Dicho esto se obtuvieron los resultados expuestos en la Figura 1.



(a) *Wall clock* para $T = 300$



(b) *Wall clock* para $T = 1000$



(c) *Wall clock* para $T = 10000$

Figura 1: Tiempos de ejecución para distintos tamaños de grilla e iteraciones

Analizando los resultados del tiempo de ejecución, se puede desprender un comportamiento similar para una distinta cantidad de iteraciones y tamaño de grilla. El rendimiento no se

ve afectado, demostrando enb gran forma la capacidad de la unidad de GPU para procesar ciertas tareas.

2.2. Ocupancia *kernel*

Para calcular la ocupancia del dispositivo, se realiza a través de la expresión mostrada en la Ecuación 5. Para esto se ha utilizado la herramienta de *CUDA Occupancy calculator*, en donde ajustando algunos parámetros es posible conocer la ocupancia. En primer lugar, se debe conocer las características de la GPU utilizada:

$$ocupancia = \frac{warps \quad activos}{maximo \quad numero \quad warps} \tag{5}$$

- *Compute capability*: 5.0 (para GPU GeForce 940MX)
- Configuración de memoria compartida: 65536 bytes
- Tamaño máximo de bloque de hilo: 1024
- *Threads per block*: 128
- *Registers Per Thread*: 40
- *Shared Memory Per Block (bytes)*: 372

Al establecer los parámetros en *CUDA Occupancy calculator*, se obtiene la información mostrada en el Cuadro 1. En donde es posible interpretar que la implementación de la solución del problema, alcanza una ocupancia del 75 %. Esto indica que la solución propuesta casi alcanza a tener los *warps* activos a el número máximo de *warps*. Como señala NVIDIA en CUDA warps and occupancy, existen algunas limitantes para la ocupancia:

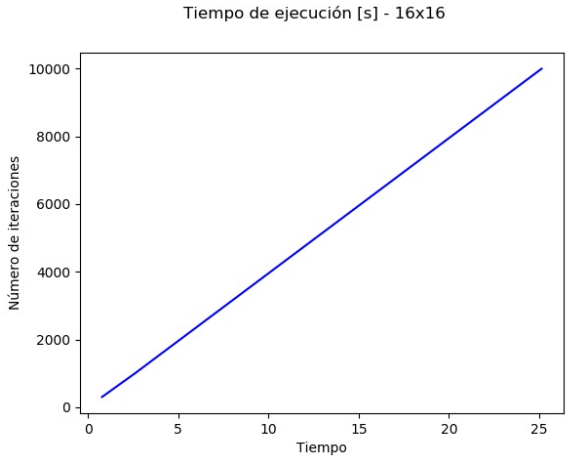
- Uso de registros
- Uso de memoria compartida
- Tamaño del bloque

Active Threads per Multiprocessor	1536
Active Warps per Multiprocessor	48
Active Thread Blocks per Multiprocessor	12
Occupancy of each Multiprocessor	75 %

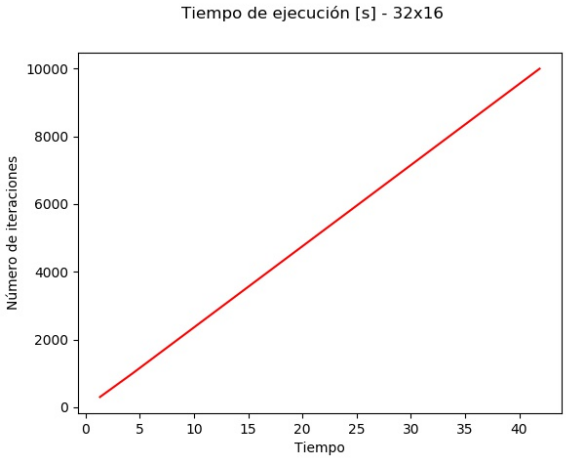
Cuadro 1: Ocupancia del *kernel*

2.3. Tiempos de ejecución para distintos tamaños de bloques

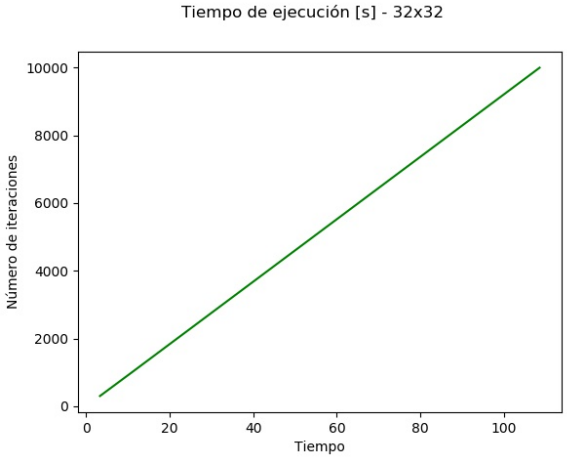
Finalmente se ha variado el tamaño de bloque en $16 \times 16, 32 \times 16$ y 32×32, con un tamaño fijo de la grilla en 2048. Para realizar una comparación con la cantidad de iteraciones, esta ha variado en $300, 1000$ y 10000. Los resultados obtenidos está en Figura 2.



(a) *Wall clock* para bloques 16×16



(b) *Wall clock* para bloques 32×16



(c) *Wall clock* para bloques 32×32

Figura 2: Tiempos de ejecución para distintos tamaños de bloque e iteraciones

Como se logra apreciar, existe un comportamiento lineal al mantener el tamaño del bloque y variar el número de iteraciones. Considerando que el tamaño de la grilla es relativamente grande 2048×2048, el menor tiempo de ejecución ocurre cuando el bloque tiene menor tamaño, seguramente esto ocurre dado que permite balancear más la carga de trabajo en GPU.

Aspectos importantes a considerar

El *benchmark* se ha realizado intentando no realizar ninguna otra tarea la máquina, la cual posee las siguientes características:

- Notebook Lenovo Ideapad 720s.
- 8 GB de RAM.
- Intel® Core™ i7-7500U CPU @ 2.70GHz × 4.
- Sistema operativo Ubuntu 19.04 con Kernel Linux 5.0.0-31-generic.
- GeForce 940MX. NVIDIA-SMI 440.33.01
- CUDA Version: 10.2