INFORME COMPARATIVO - CUDA

Hardware Utilizado:

CPU	: AMD Athlon(tm) II X3 455 Processor × 3 3,300 Mhz
Memoria	: 8 GB
SO	: Ubuntu 16.04 x64
Dispositivo CUDA	: GeForce GTX 1050 – 640 CudaCores
Ancho de Banda Memoria (bits)	: 128
Peak Memory Bandwidth (GB/s)	: 112.128000
Memoria global	: 1997mb
Memoria compartida	: 48kb
Memoria constante	: 64kb
Registros por bloque	: 65536
Multi Processor Count	: 5
Tamaño Warp	: 32
Threads por block	: 1024
Dimension Max block	: [1024, 1024, 64]
Dimension Max grid	: [2147483647, 65535, 65535]

Pruebas:

Para las pruebas respectivas , se utilizaron matrices cuadradas, en bloques respectivos de 16 y 32 . Se tomo como maximo 10240 como tamaño de la matriz, debido al tamaño de memoria de la GPU.
Ademas se debe tener en cuenta que la multiplicacion de matrices con memoria compartida necesita que el tamaño de la matriz y tamaño del bloque sea un multiplo de este.

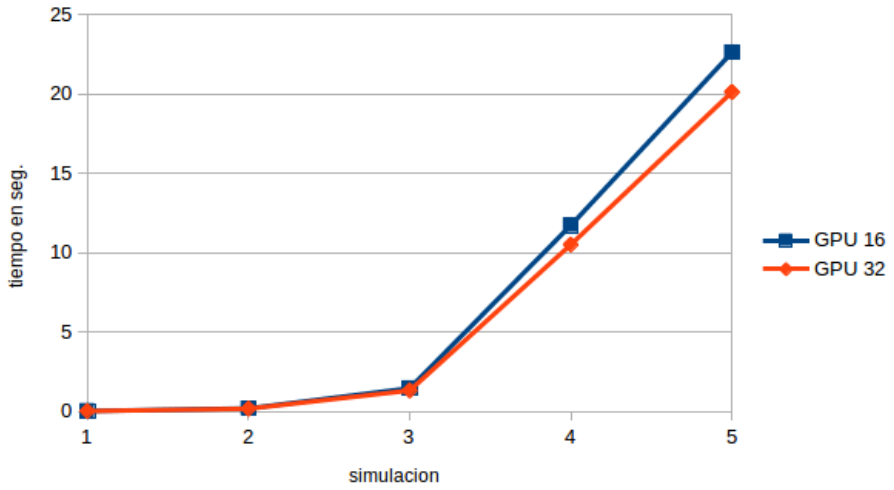
- Multiplicacion de matrices tradicional en cuda

En la tabla podemos apreciar que entre mas vaya creciendo el tamaño de la matriz, el tiempo de ejecuion va en aumento

		bloques de 16	bloques de 32
Simulacion	Matriz cuadrada	GPU (seg.)	GPU (seg.)
1	1024	0.029394	0.029315
2	2048	0.196258	0.176635
3	4096	1.46673	1.32266
4	8192	11.733	10.5179
5	10240	22.6319	20.1311

* Podemos observar que existe una pequeña diferencia de alguno segundos, entre bloques de 16 y bloques de 32.

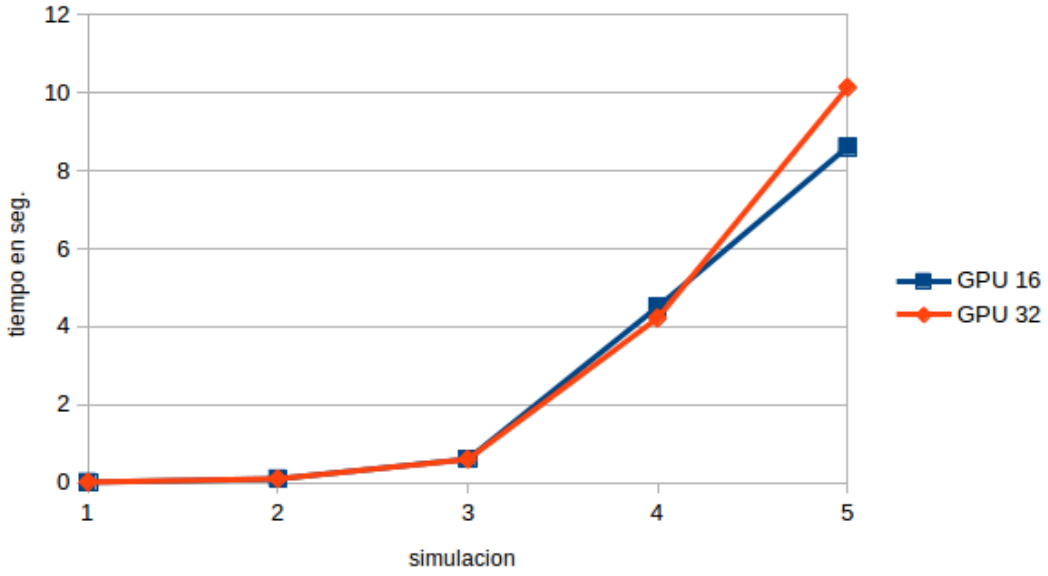
-



Multiplicacion de matrices utilizando memoria compartida en cuda

		bloques de 16	bloques de 32
Simulacion	Matriz cuadrada	GPU (seg.)	GPU (seg.)
1	1024	0.014546	0.01655
2	2048	0.100888	0.101018
3	4096	0.605461	0.599393
4	8192	4.50833	4.22293
5	10240	8.61183	10.1405

En general podemos apreciar que la multiplicacion en matrices con el uso de memoria compartida es mas rapido debido a que la memoria local tiene un acceso mas rapido y los datos no se estan leyendo constantemente de la memoria global



En las graficas siguientes podemos observar la diferencia entre la multiplicacion por bloques de tamaño 32. Fig1. Y la diferencia entre multiplicacion con tiles de tamaño 32. Fig 2

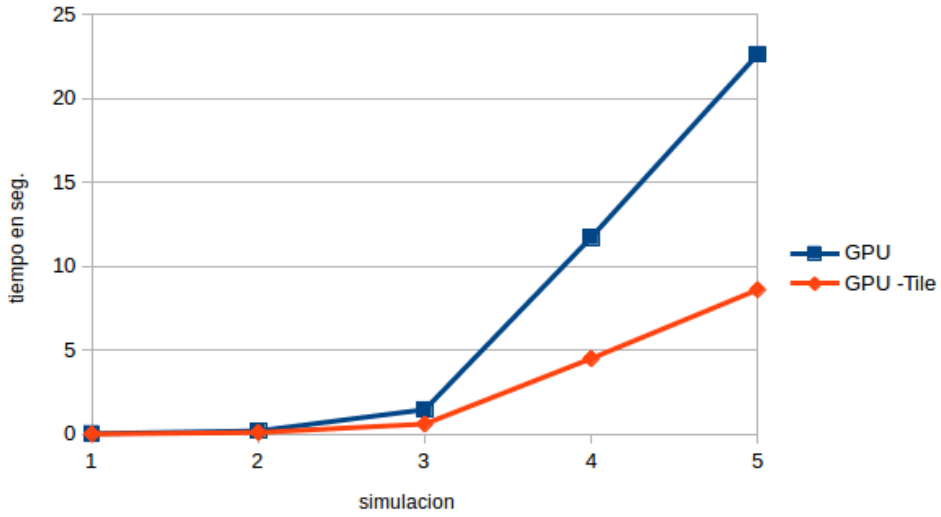


Fig. 1 Grafica de tiempo entre bloques de tamaño 16, usando la multiplicacion por bloque y multiplicacion con tiles.

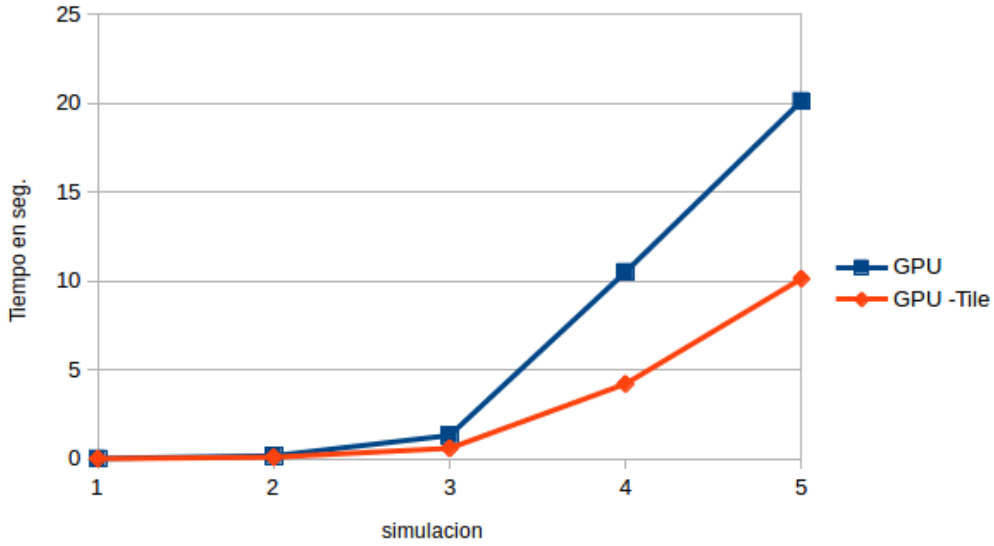


Fig. 2 Grafica de tiempo entre bloques de tamaño 32, usando la multiplicacion por bloque y multiplicacion con tiles.

Conclusiones

- El uso de meoria global tiene un coste mayor de lectura y escritura de datos
- El uso de memoria local tiene un coste menor de lectura y escritura de datos, pero contiene un limite de memoria mucho menor a la memoria global, y esta depende de la arquitectura de la GPU utilizada.