MULTIPLICACIÓN DE MATRICES EN CPU, GPU Y GPU CON MEMORIA COMPARTIDA

HIGH PERFORMANCE COMPUTING

DANIEL ESTEBAN ARIAS ACOSTA
1088279598

UNIVERSIDAD TECNOLOGICA DE PEREIRA
INGENIERIA DE SISTEMAS
Y COMPUTACIÓN
PEREIRA 2015

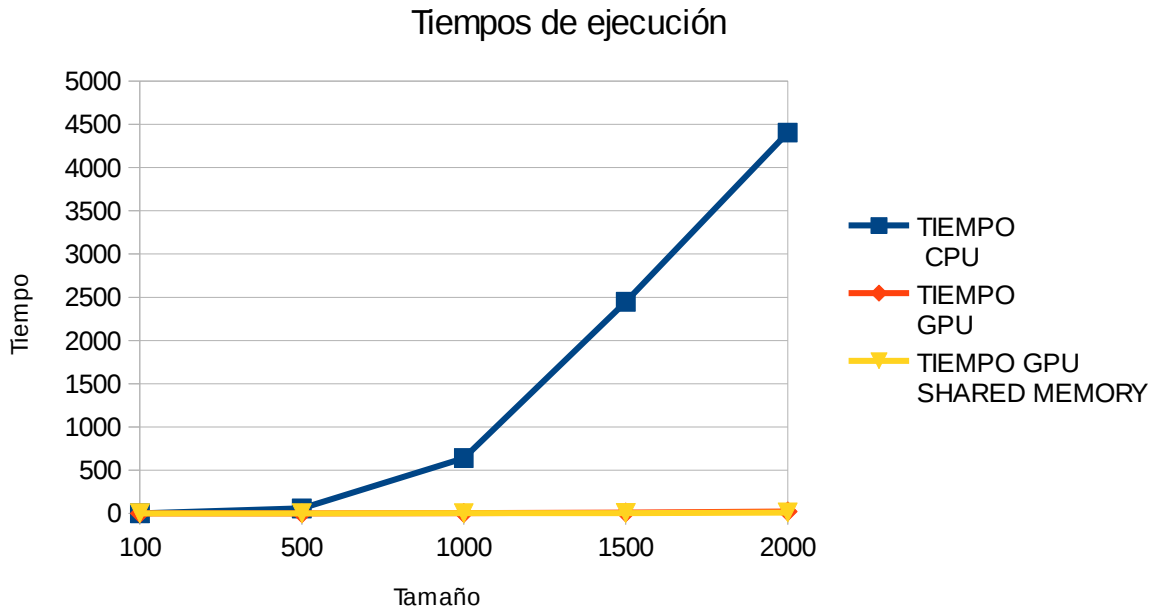
RECOLECCIÓN DE DATOS

1. En la tabla 1,1 se pueden observar los tiempos de ejecución para matrices de diferentes tamaños en las diferentes versiones del algoritmo: Secuencial, cuya ejecución se lleva a cabo en la CPU, el paralelo sin tiling en la GPU y en la última columna, el paralelo con tiling (también ejecutado en la GPU pero usando memoria compartida).

TAMAÑO MATRIZ	TIEMPO CPU	TIEMPO GPU	TIEMPO GPU SHARED MEMORY
100	0,395	0,0148	0,0105
500	57,0969	0,4328	0,2176
1000	639,625	2,9088	1,2591
1500	2446,1794	9,647	3,5283
2000	4404,5723	21,8998	8,8858

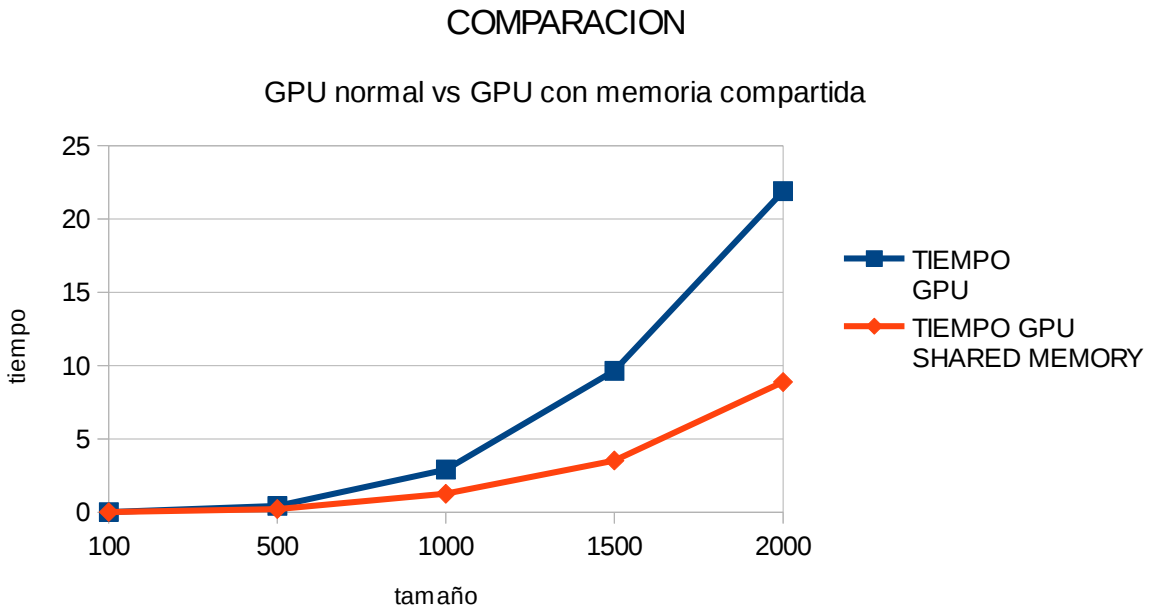
Tabla 1. Tiempos de ejecución en segundos x10^(-2).

En la gráfica 1,1, se puede apreciar la gran diferencia en los tiempos de ejecución en las diferentes versiones del algoritmo, con distintos tamaños. Es significativo el tiempo que toma la CPU al ejecutar la versión del algoritmo secuencial a partir de los 500 datos en adelante.



Gráfica 1,1. Aceleraciones con las tres versiones del algoritmo.

Para apreciar de mejor forma la diferencia, en la gráfica 1,2, se logra divisar con mas detalle los tiempos de ejecución de los algoritmos en la GPU, con y sin memoria compartida.



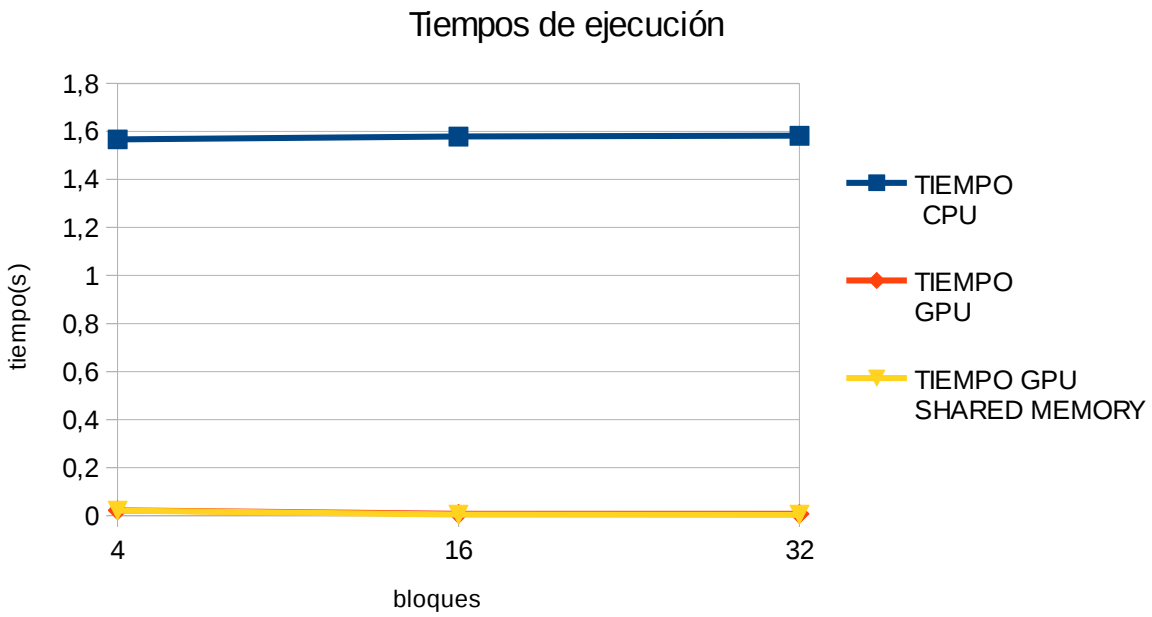
Gráfica 1,2. Gráfico detallado de tiempos de ejecución en GPU.

En los puntos **2**,**3** y **4,** se encuentran condensados los tiempos de ejecución tomados de la multiplicación de matrices en las tres versiones del algoritmo con sus respectivas gráficas.
A diferencia del punto anterior, en este caso se mantienen las dimensiones de la matriz y se cambian la cantidad de bloques en cada toma de datos, para ver que tanto afecta esto en los tiempos de ejecución.

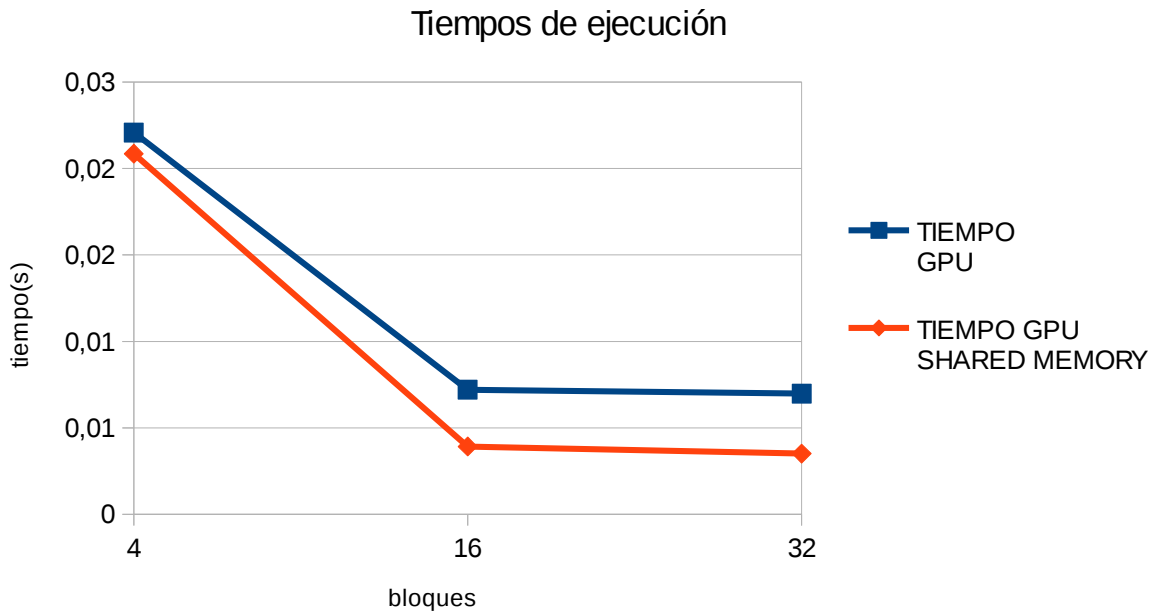
2.

Matriz A		Matriz B		TILE_WIDTH	TIEMPO CPU	TIEMPO GPU	TIEMPO GPU HARED MEMOI	CPU/GPU	CPU/SHM
Filas	Columnas	Filas	Columnas						
1024	512	512	512	4	1,566935	0,02208	0,020855	70	75
1024	512	512	512	16	1,578815	0,007206	0,00392	219	402
1024	512	512	512	32	1,581926	0,006977	0,003508	226	450

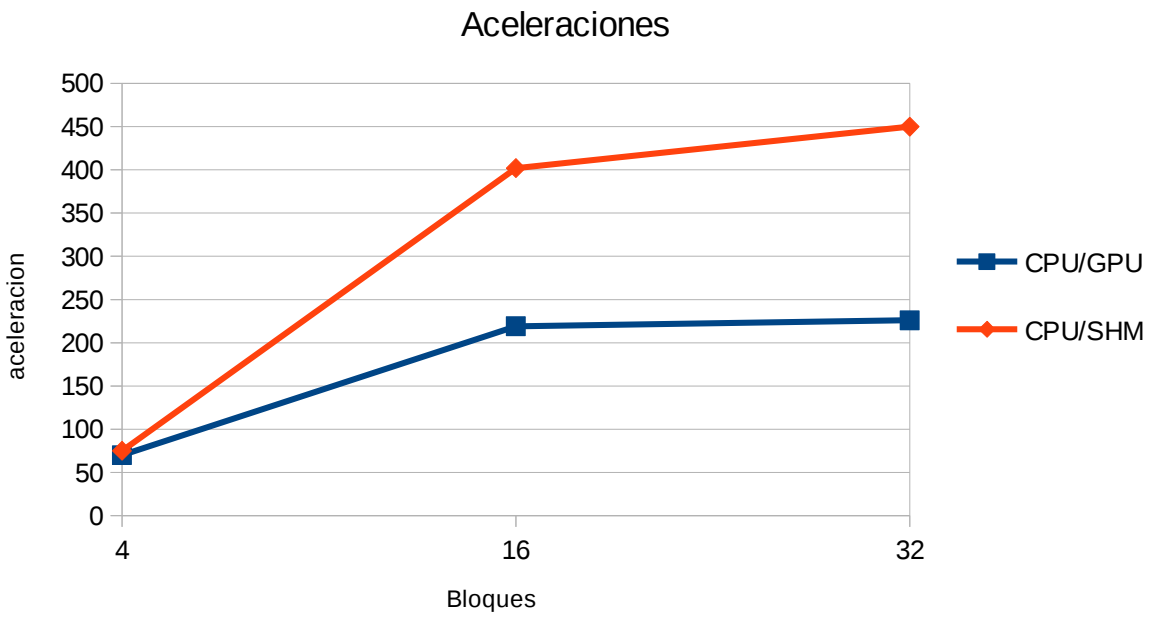
Tabla.2. Datos tomados a partir de la multiplicación de matrices.



Gráfica 2,1. tiempos de ejecución en los tres algoritmos



Grafica2,2. Gráfico detallado de tiempos de ejecución en GPU.

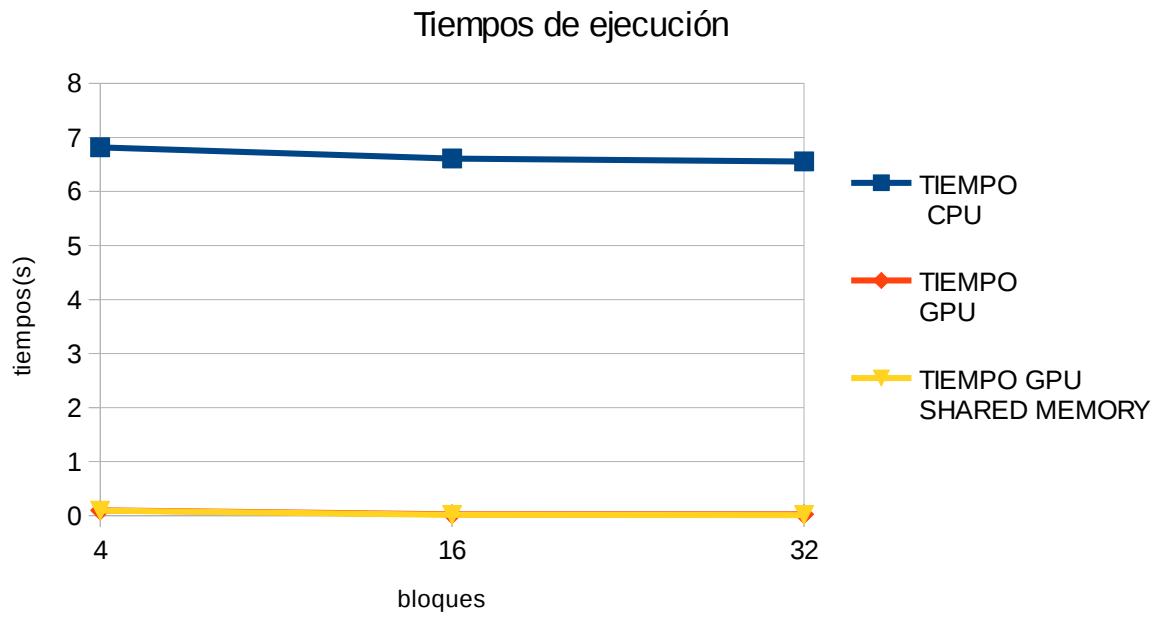


Gráfica 2,3. Aceleración de los algoritmos por bloque

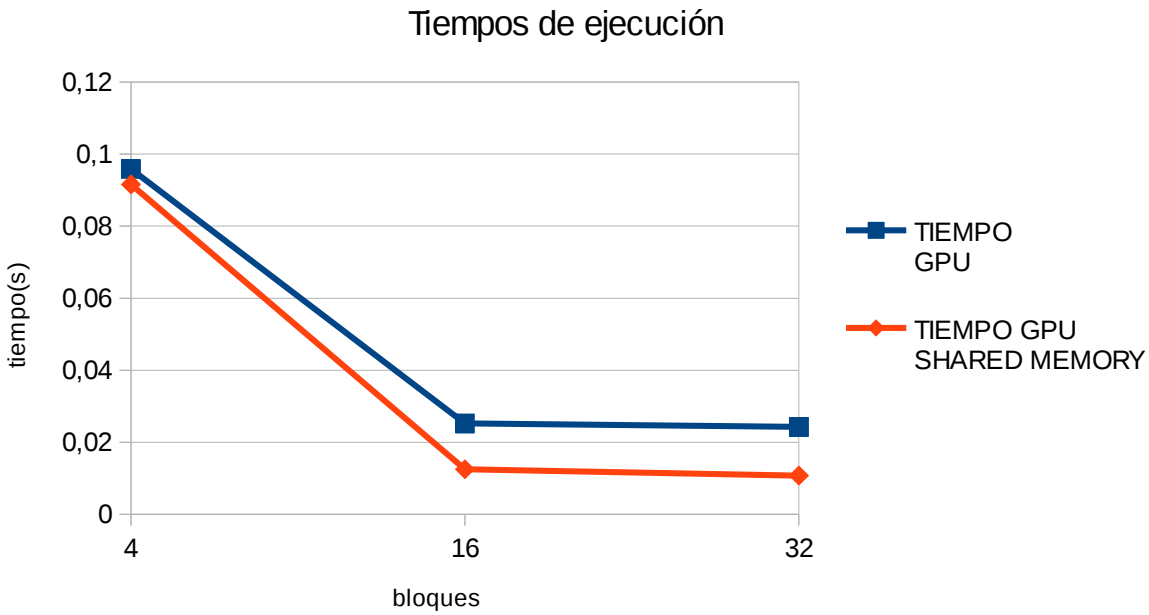
3.

Matriz A		Matriz B		TILE_WIDTH	TIEMPO CPU	TIEMPO GPU	TIEMPO GPU SHARED MEMORY	CPU/GPU	CPU/SHM
Filas	Columnas	Filas	Columnas						
1024	1024	1024	1024	4	6,817961	0,095859	0,091581	71	74
1024	1024	1024	1024	16	6,608912	0,025207	0,012489	262	529
1024	1024	1024	1024	32	6,556473	0,02424	0,010722	270	611

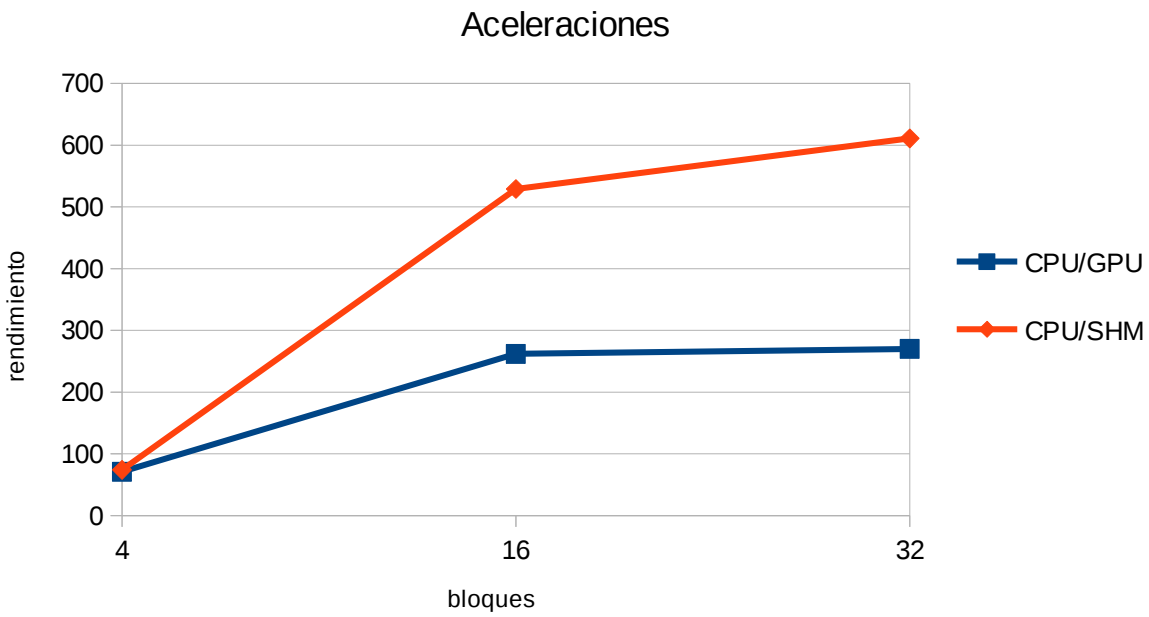
Tabla.3. Datos tomados a partir de la multiplicación de matrices.



Gráfica 3,1. tiempos de ejecución en los tres algoritmos



Grafica3,2. Gráfico detallado de tiempos de ejecución en GPU.

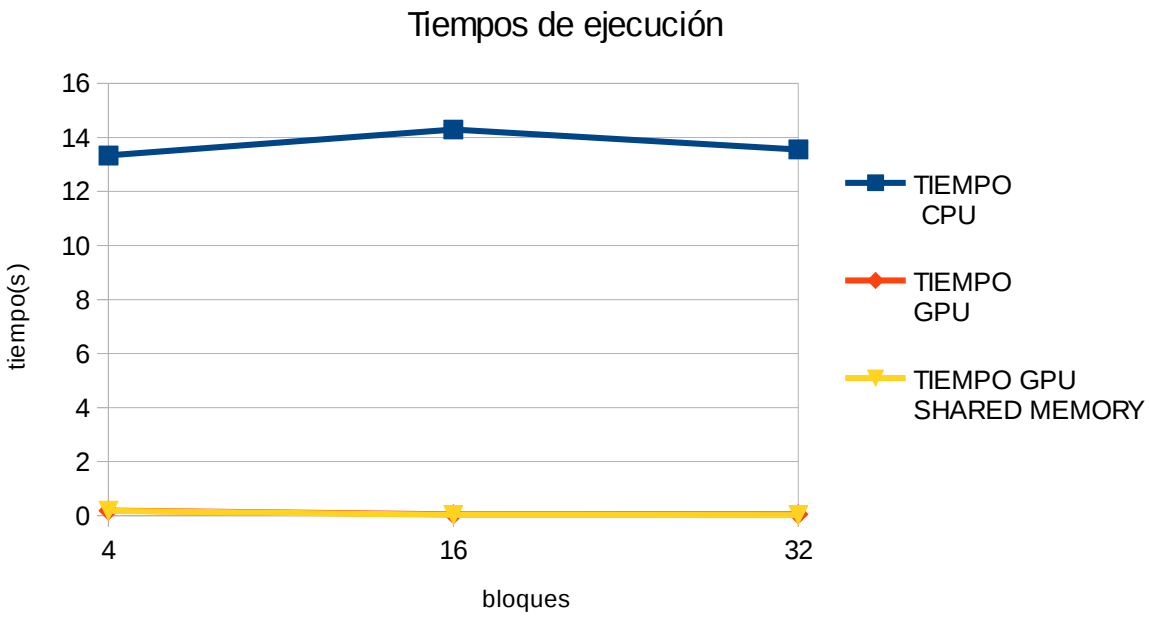


Gráfica 3,3. Aceleración de los algoritmos por bloque

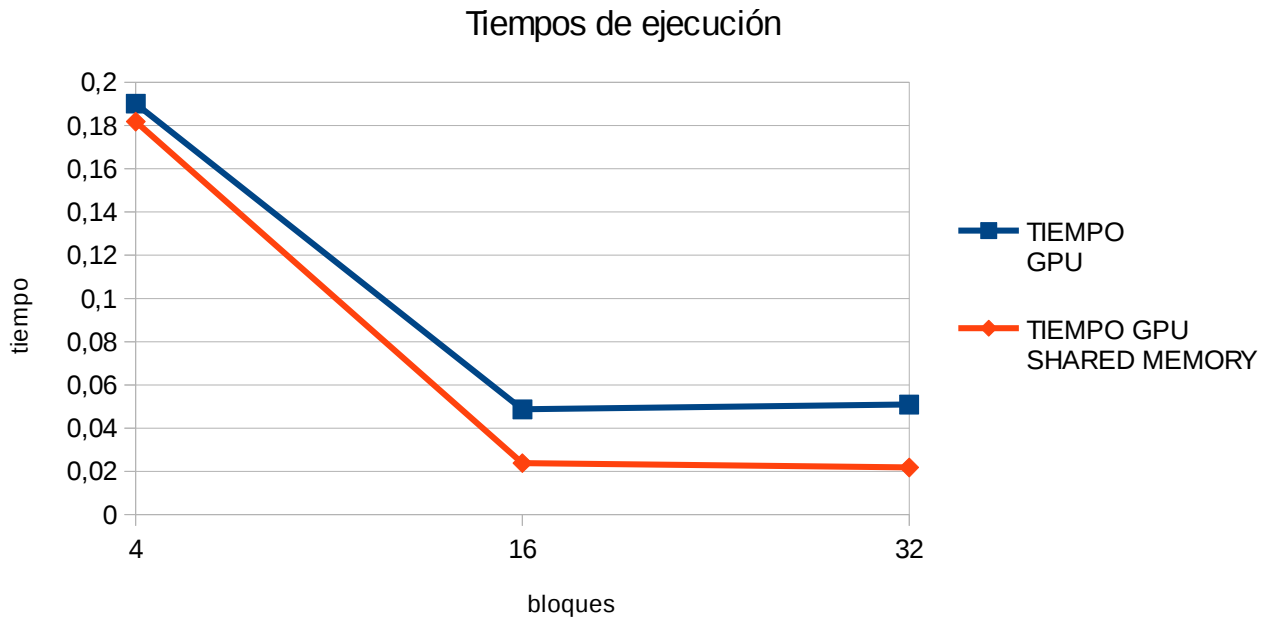
4.

Matriz A		Matriz B		TILE_WIDTH	TIEMPO CPU	TIEMPO GPU	TIEMPO GPU SHARED MEMORY	CPU/GPU	CPU/SHM
Filas	Columnas	Filas	Columnas						
2048	1024	1024	1024	4	13,329757	0,190123	0,181807	70	73
2048	1024	1024	1024	16	14,293106	0,048688	0,023789	293	600
2048	1024	1024	1024	32	13,553782	0,050877	0,021752	266	623

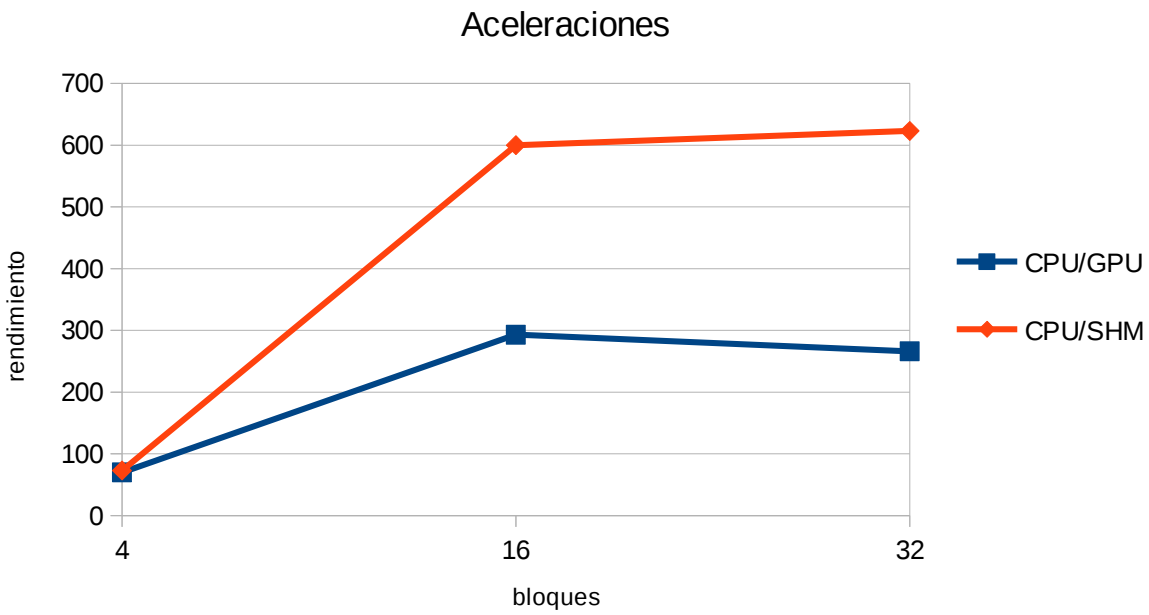
Tabla.4. Datos tomados a partir de la multiplicación de matrices.



Gráfica 4,1.



Gráfica 4,2.



Gráfica 4,3.

CONCLUSIONES

En la gráfica 1 de cada punto, se puede apreciar una gran diferencia entre los tiempos de ejecución de la CPU y la GPU, tanto asi que no se logran diferenciar los tiempos entre el algoritmo que utiliza memoria compartida y el que no. En primera instancia se concluye que los tiempos de ejecución en la GPU son muy inferiores.

En la gráfica 2 de cada punto, se puede observar una leve diferencia en los tiempos de ejecución al incrementar el número de bloques y su TILE_WIDTH, sin embargo, de aquí se puede concluir que al tener un numero de bloques y TILE_WIDTH mayor, se pueden disminuir los tiempos de ejecución.

En la gráfica 3 de cada punto, se complementa lo visto en la dos respectivamente, sin embargo en esta podemos apreciar un mejor comportamiento del algoritmo de memoria compartida en la GPU con respecto a su par sin memoria compartida.