Primer parcial

Santiago Olarte Zapata

Cod: 1.088.317.736

High Performance Computing

Universidad Tecnológica de Pereira

Marzo de 2015

Introducción

En los últimos años se ha logrado aumentar progresivamente la velocidad de procesamiento de los equipos de cómputo. Sin embargo no fue sino hasta el año 2000 cuando procesadores especializados llamados GPUs empezaron a ser fabricados por NVIDIA y ATI/AMD, que eran capaces de soportar las cargas de trabajo de la computación de alto desempeño. Luego, CUDA proporcionó acceso directo al conjunto de instrucciones virtual y la memoria de los elementos de computación paralela de las GPUs, permitiéndoles tareas de procesamiento de propósito general. Con la implementación de algoritmos paralelos, no sólo se logró una aceleración de los algoritmos diseñados de la forma tradicional, sino que también permitió reducir la carga computacional de los procesadores de las máquinas. De la misma forma, se han desarrollado formas de acelerar aún más los algoritmos paralelos, como es el caso de una técnica llamada tiling que consiste en "partir" los datos de tal forma que puedan ser almacenados en memoria compartida, que trae como resultado una reducción dramática de latencia causada por acceso a memoria, en la medida que esta es más rápida que la memoria global del dispositivo.

El objetivo del presente documento es demostrar, en primer lugar la efectividad de los algoritmos paralelos a la hora de realizar tareas repetitivas aprovechando su capacidad de procesamiento, permitiendo resolución de problemas de manera más rápida. De la misma forma, se espera comprobar la forma en la que la aplicación de la técnica de tiling para el algoritmo paralelo afecta el tiempo que este tarda para resolver una multiplicación de matrices de diferentes tamaños.

Procedimiento

Se implementaron 3 versiones de un algoritmo para realizar la multiplicación de dos matrices dadas las dimensiones de las mismas:

-Algoritmo secuencial

-Algoritmo paralelo

-Algoritmo paralelo aplicando técnica de tiling

Luego se ejecutaron los algoritmos en la plataforma CUDA Online Compiler del grupo de investigación SIRIUS que consta de un editor de texto y una consola de salida.

El desempeño de los algoritmos se evaluó ejecutándolos con tamaños de matrices diferentes. Los valores de los números de datos que se muestran en las gráficas corresponden a la multiplicación de las dos dimensiones de una matriz sumada con la multiplicación de las dos dimensiones de la otra.

Se midió el tiempo que tardó cada uno de los algoritmos en realizar la multiplicación con el mismo set de datos, se registraron en tablas y posteriormente se realizaron gráficas basadas en dichos resultados. También se calculó la aceleración efectuada por los algoritmos así:

a. Tiempo algoritmo secuencial/ tiempo algoritmo paralelo
b. Tiempo algoritmo paralelo /Algoritmo paralelo optimizado
c. Tiempo Algoritmo secuencial/ Tiempo algoritmo optimizado

Los datos anteriores fueron igualmente registrados en tablas.

Se ejecutaron los algoritmos con diferentes tamaños de bloque (32X32, 16X16, 4X4) y se realizaron las pruebas anteriormente descritas.

Finalmente se realizaron todas las pruebas tanto para matrices de tipo punto flotante como para entero y se compararon los resultados obtenidos en cada una de ellas.

PUNTO FLOTANTE

a. Tamaño de bloque 32X32

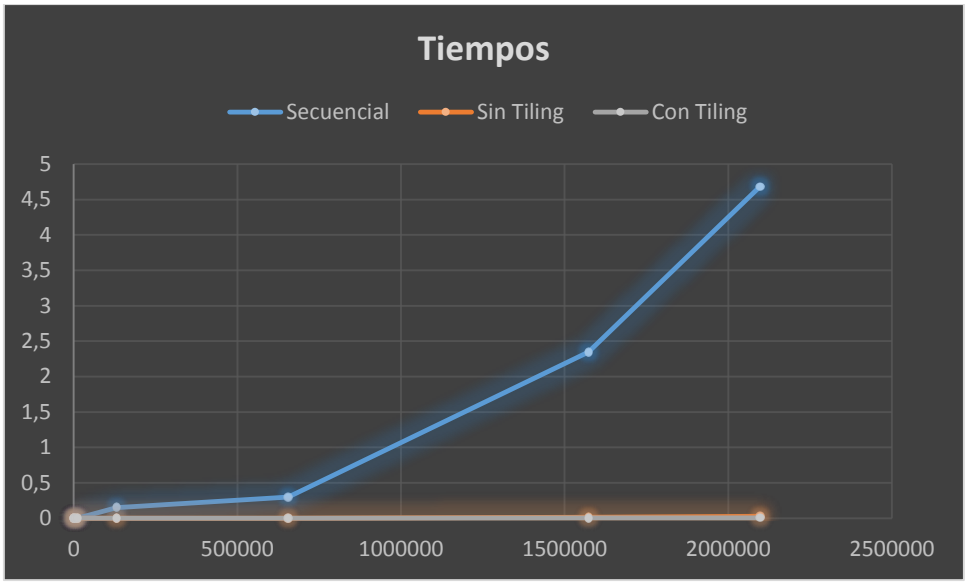


Figura 1: Gráfica de tiempos de algoritmos Secuencial, paralelo y paralelo optimizado (# Datos Vs. Tiempo)

En la figura 1 es muy notable las diferencias en el tiempo de ejecución del algoritmo secuencial con los algoritmos paralelos. Sin embargo, no es posible comparar el rendimiento de los algoritmos paralelos

Figura 2: Gráfica de tiempos algoritmos paralelo y paralelo optimizado (# datos Vs. Tiempo)

En la figura 2 se puede apreciar una disminución notable en el tiempo de ejecución del algoritmo paralelo utilizando la técnica de tiling.

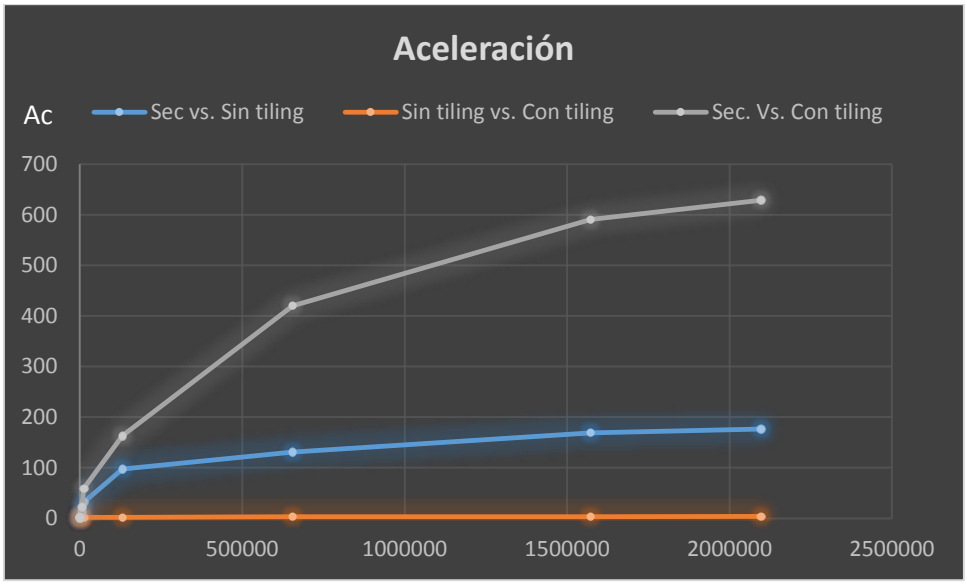


Figura 3: Gráfica de aceleración de los 3 algoritmos (# Datos Vs. Aceleración)

La gráfica 3 muestra el grado de aceleración de los tres algoritmos comparados así:

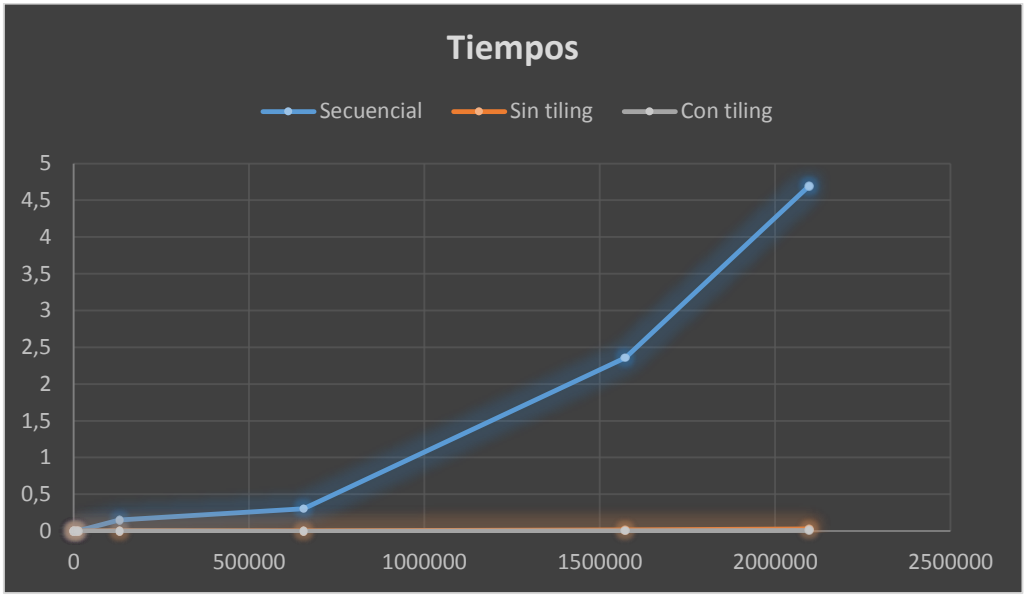
-Algoritmo secuencial Vs. Algoritmo paralelo sin optimizar

-Algoritmo secuencial Vs. Algoritmo paralelo optimizado

-Algoritmo paralelo sin optimizar Vs. Algoritmo paralelo optimizado.

Se puede observar que las aceleraciones que dan como resultado los algoritmos es de alrededor de 600X para un total de datos mayor a 1’600.000, con el algoritmo paralelo sin optimizar, y de alrededor de 150X para el mismo número de datos, con el algoritmo optimizado, en contraste con el algoritmo secuencial. Aunque no se logre apreciar, la aceleración lograda con el algoritmo paralelo optimizado en comparación con el no optimizado es de alrededor de 3X, para el mismo número de datos.

b. Tamaño de bloque 16X16

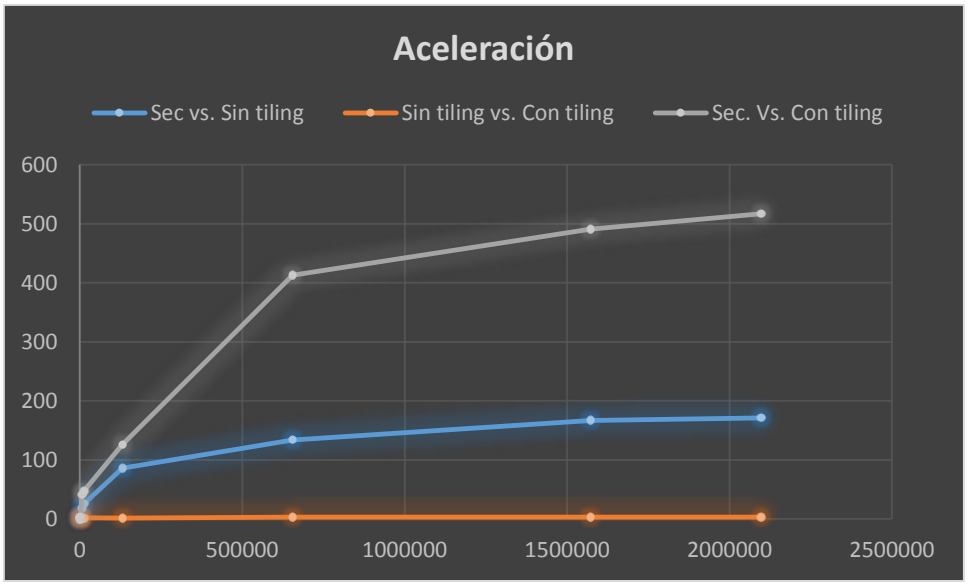


Grafica 4: Gráfica de tiempos de los tres algoritmos Secuencial, paralelo y paralelo optimizado.

Los resultados obtenidos en la gráfica 4 son similares a los obtenidos en las pruebas con tamaño de bloque de 32X32; se identifica un mejor rendimiento de los algoritmos paralelos.

Gráfica 5: Grafica de tiempos de ejecución los algoritmos paralelo y paralelo optimizado.

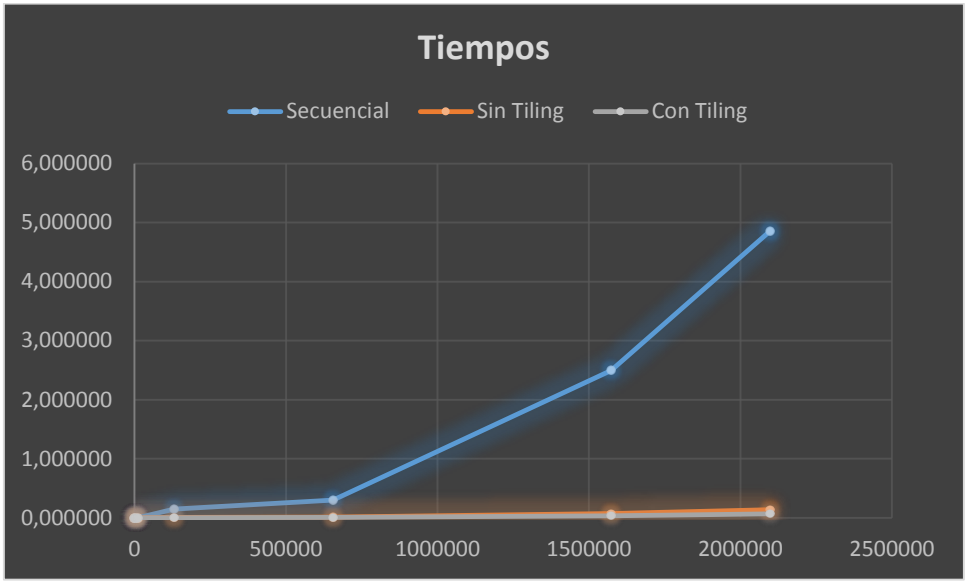
En la gráfica 5 se puede observar que al aplicar la técnica de tiles al algoritmo paralelo el rendimiento del algoritmo mejora.



Grafica 6: Gráfica de aceleración de los 3 algoritmos (# Datos Vs. Aceleración)

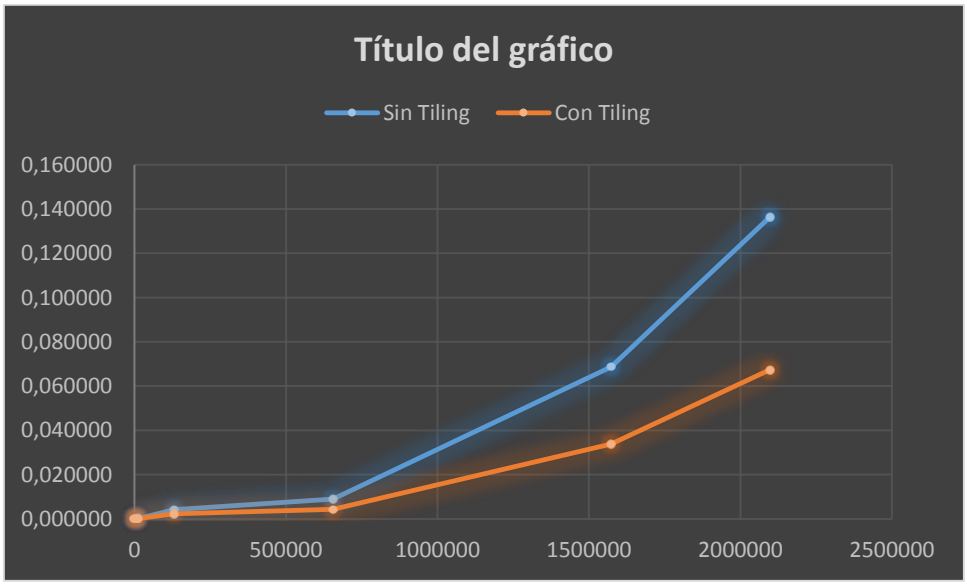
En la gráfica 6, se puede observar que las aceleraciones que se obtiene con el algoritmo optimizado con respecto al secuencial es de alrededor de 500X para un total de datos mayor a 1’500.000, con el paralelo sin optimizar respecto al secuencial alrededor de 200X para el mismo número de datos. Aunque no se logre apreciar, la aceleración lograda con el algoritmo paralelo optimizado en comparación con el no optimizado es de alrededor de 3X, para el mismo número de datos.

c. Tamaño de bloque 4X4



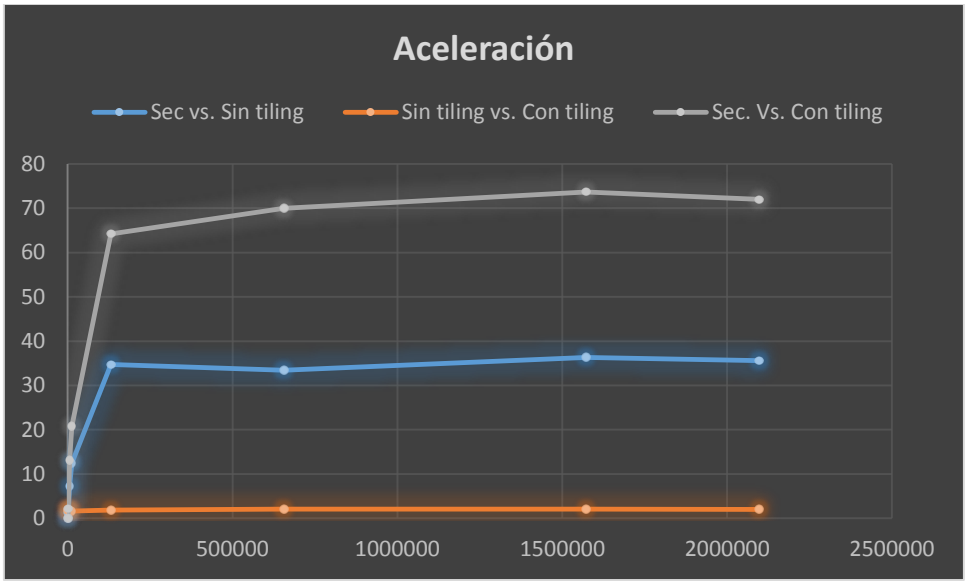
Grafica 7: Gráfica de tiempos de algoritmos Secuencial, paralelo y paralelo optimizado (# Datos Vs. Tiempo)

En la gráfica 7 se observa un tiempo de ejecución menor que en el algoritmo secuencial. Sin embargo, no es posible apreciar diferencias en el tiempo de ejecución entre las dos formas del algoritmo paralelo



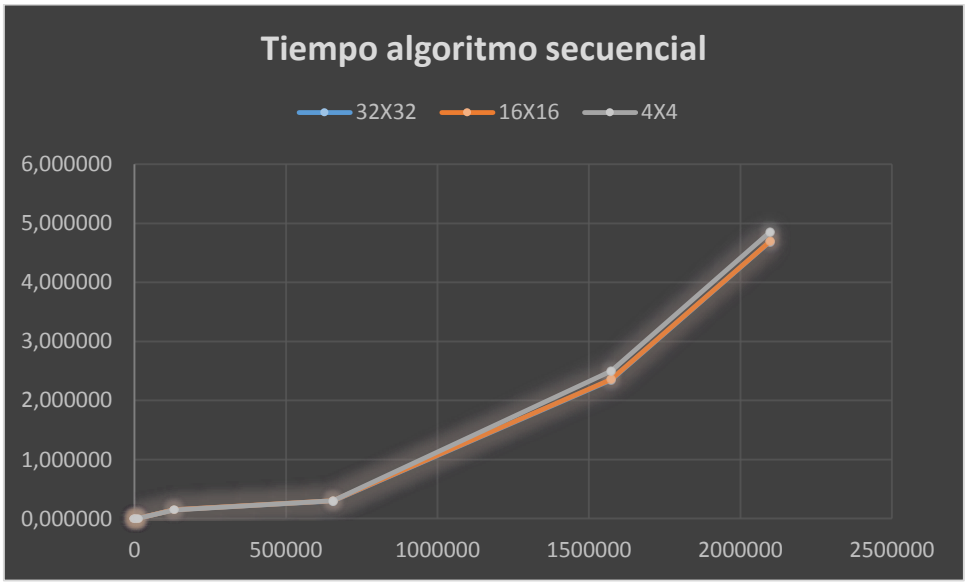
Gráfica 8: Gráfica de tiempos algoritmos paralelo y paralelo optimizado (# datos Vs. Tiempo)

En la gráfica 8 se puede observar una mejora en el tiempo de ejecución del algoritmo paralelo utilizando la técnica de tiling

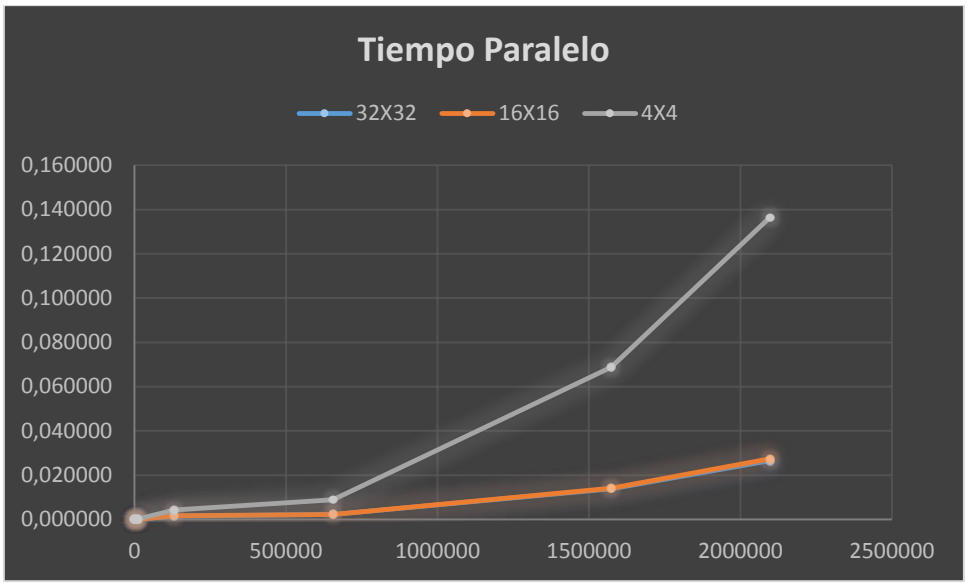


Gráfica 9: Gráfica de aceleración de los tres algoritmos (# de datos Vs. Aceleración)

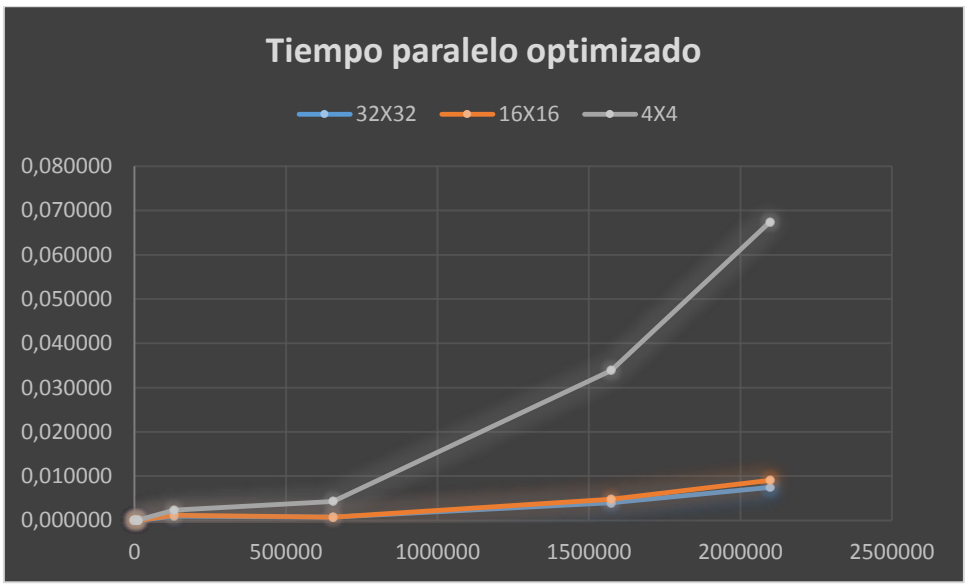
En la gráfica 9 se obtuvo que la aceleración del algoritmo paralelo utilizando tiling con respecto al secuencial fue de alrededor de 70X, del algoritmo paralelo sin optimizar con respecto al secuencial fue de alrededor de 35X y la aceleración del algoritmo utilizando tiles, en comparación con el que no lo utiliza, fue de alrededor de 2X, para una cantidad de datos total superior a 131 072.



Gráfica 10: Grafico de tiempos algoritmo secuencial para diferentes tamaños de bloque. (# de dato Vs. Tiempo)



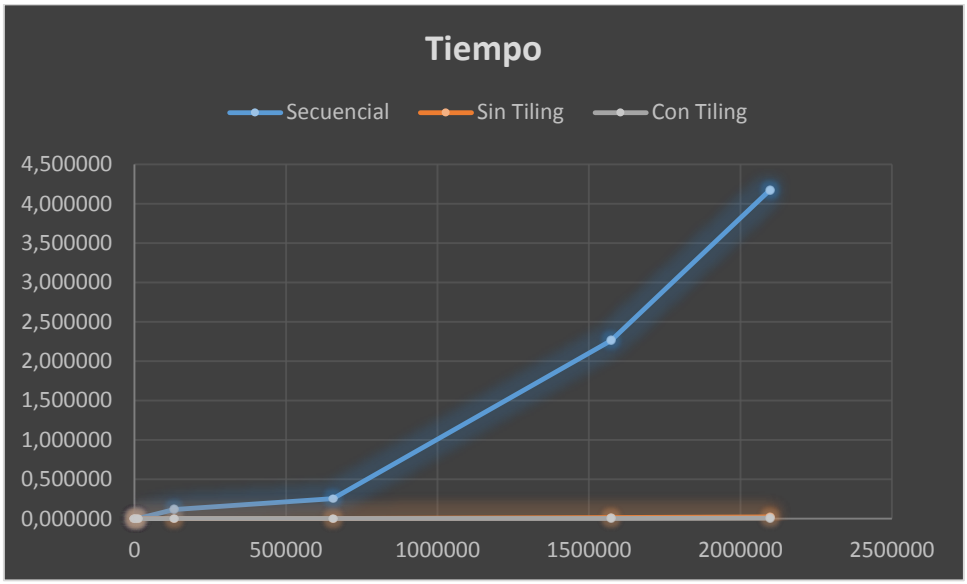
Gráfica 11: Grafico de tiempos algoritmo paralelo sin optimizar para diferentes tamaños de bloque. (# de dato Vs. Tiempo)



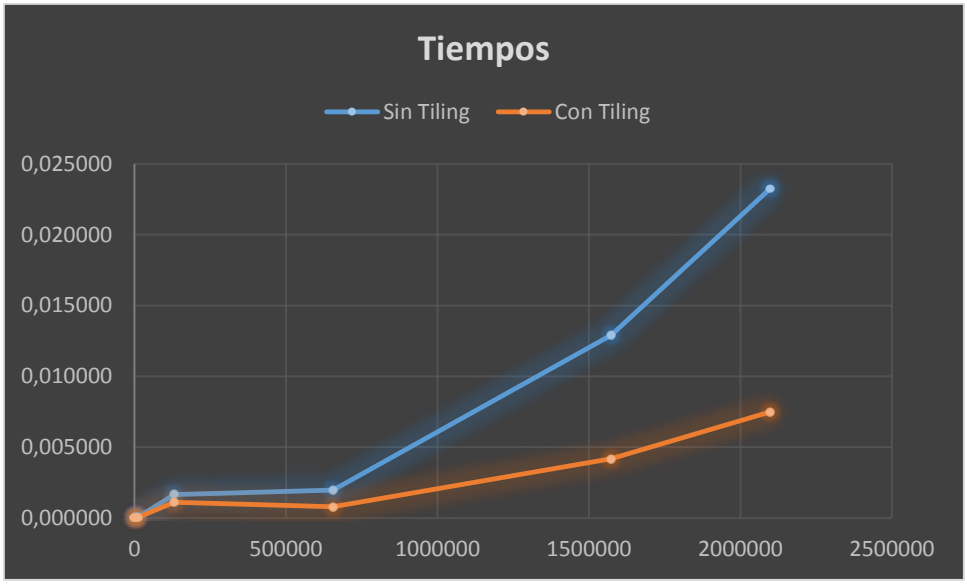
Gráfica 12: Grafico de tiempos algoritmo paralelo optimizado lanzado con diferentes tamaños de bloque. (# de dato Vs. Tiempo)

ENTEROS

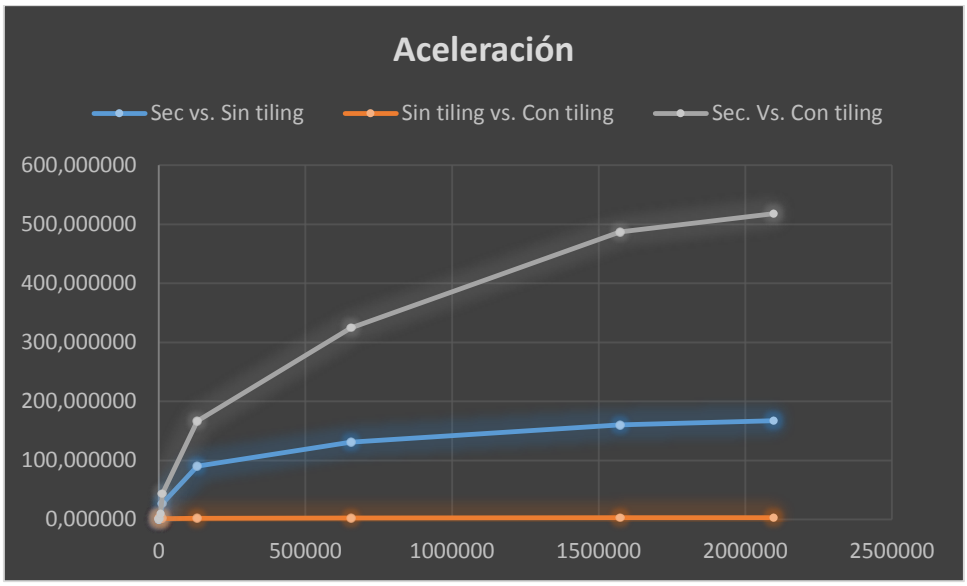
a. Tamaño de bloque 32X32



Grafica 13: Gráfica de tiempos de algoritmos Secuencial, paralelo y paralelo optimizado (# Datos Vs. Tiempo)

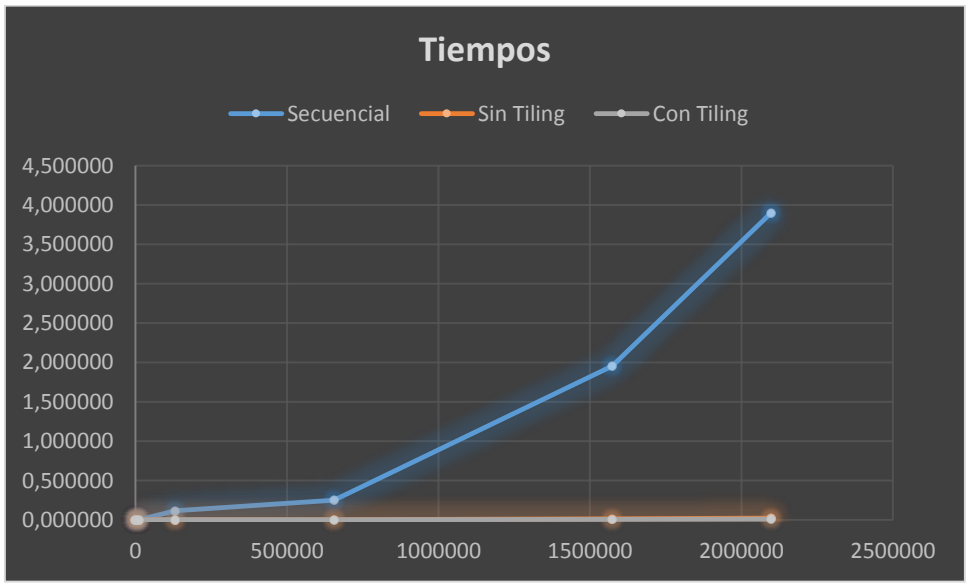


Gráfica 14: Gráfica de tiempos de algoritmo paralelo y paralelo optimizado



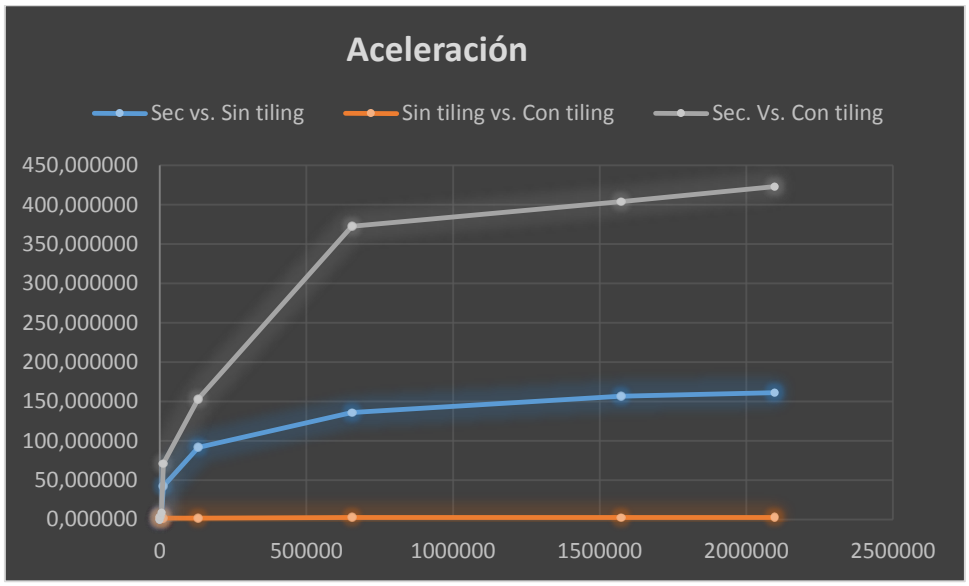
Gráfica 15: Gráfica de aceleración de los tres algoritmos (# de datos Vs. Aceleración)

b. Tamaño del bloque 16X16



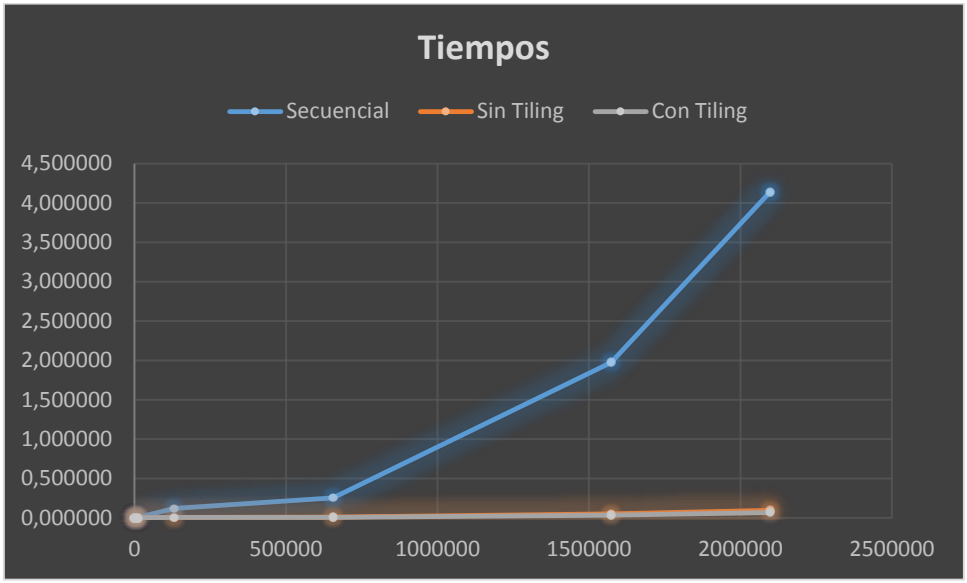
Gráfica 16: Gráfica de tiempos de algoritmos Secuencial, paralelo y paralelo optimizado (# Datos Vs. Tiempo)

Gráfica 17: Gráfica de tiempos de algoritmos paralelo y paralelo optimizado (# Datos Vs. Tiempo)

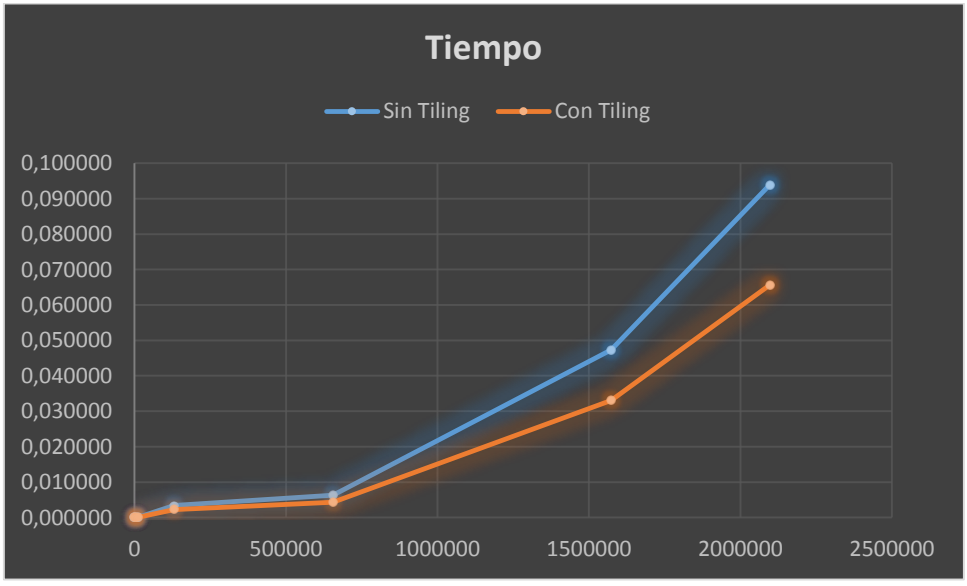


Gráfica 18: Gráfica de aceleración de los tres algoritmos (# de datos Vs. Aceleración)

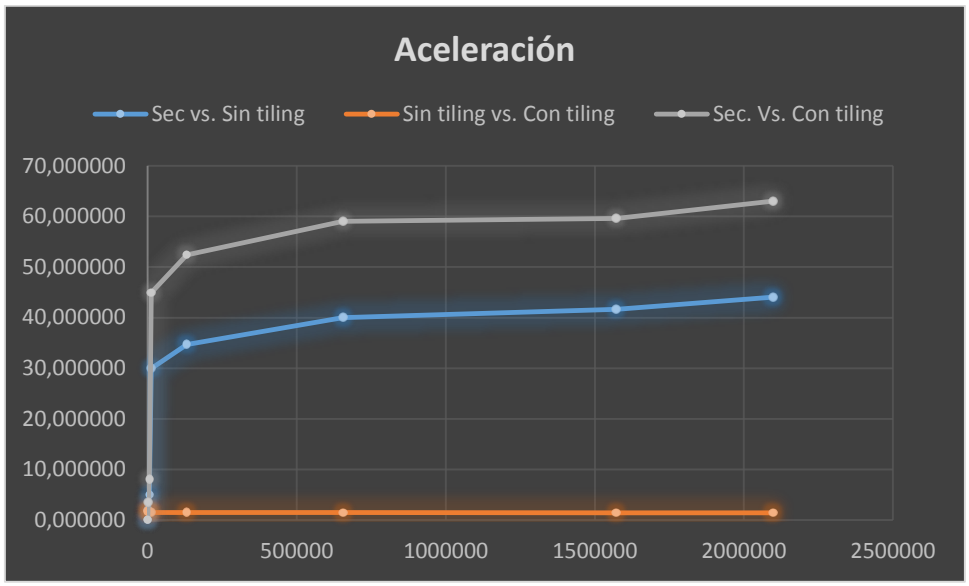
c. Tamaño de los bloques 4X4



Gráfica 19: Gráfica de tiempos de algoritmos Secuencial, paralelo y paralelo optimizado (# Datos Vs. Tiempo)



Gráfica 20: Gráfica de tiempos de algoritmos paralelo y paralelo optimizado (# Datos Vs. Tiempo)



Gráfica 21: Gráfica de aceleración de los tres algoritmos (# de datos Vs. Aceleración)

CONCLUSIONES

-En la gráfica 10 no se observa diferencia en cuanto a tiempo de ejecución al correr el algoritmo de multiplicación de matrices secuencial con diferentes tamaños de bloque (32X32, 16X16, 4X4)

-En la gráfica 11 se observa que el algoritmo de multiplicación de matrices en paralelo sin optimización lanzado con bloques de tamaño 4X4 registra el mayor tiempo de ejecución en comparación con el mismo algoritmo lanzado tanto en bloques de 16X16 y 32X32. No se aprecia diferencia en cuanto al tiempo de ejecución de estos dos últimos.

-En la gráfica 12 se observa que el algoritmo de multiplicación de matrices en paralelo optimizado, lanzado con bloques de tamaño 4X4 registra el mayor tiempo de ejecución. El algoritmo ejecutado en bloques de tamaño 16X16 registra un tiempo ligeramente mayor que el lanzado en bloques de 32X32. Finalmente aquel lanzado en bloques de 32X32 presentó el menor tiempo de ejecución para una cantidad de datos mayor a 1’600.000.

-Las gráficas que representan los tiempos de ejecución de los tres algoritmos, tanto para enteros como para punto flotante, permiten visualizar la optimización del tiempo de ejecución de los algoritmos genéticos.

-En las gráficas en las que se comparan los tiempos de ambos algoritmos paralelos, siempre el paralelo sin optimización tarda más tiempo en realizar la operación

-Las gráficas de aceleración en las que se comparan los 3 algoritmos muestran la efectiva aceleración lograda por el algoritmo paralelo utilizando la técnica de tiling.

-Las gráficas muestran que todos los valores de aceleración tienden a estabilizarse a medida que aumenta la cantidad de datos

-Se logró determinar que una menor cantidad de bloques afectan los algoritmos paralelos en la medida que aumentan su tiempo de ejecución.