Centro de Investigación en Matemáticas

Maestría en Ciencias con especialidad en Computación y Matemáticas Industriales
Métodos Numéricos (2019)

Tarea 08 - Paralelización

Reporte

Marco Antonio Esquivel Basaldua (MCMESQ)

Dentro de la ejecución de códigos de programa existen procedimientos los cuales son posibles de paralelizar, esto es, instrucciones o subrutinas del programa que pueden realizarse de manera simultanea. La paralelización es posible gracias a la tecnología $multi-core$ que hace referencia a computadoras de procesadores con dos o más unidades de procesamiento separadas. El objetivo de la paralelización es reducir el tiempo de ejecución del programa. Para llevar a cabo este proceso de paralelización se hace uso de la interfaz de programación OpenMP.

En el presente reporte se muestran los resultados de aplicar la paralelización usando OpenMP en los programas de: suma de vectores, multiplicación de vectores elemento a elemento, producto punto, multiplicación matriz vector y multiplicación matriz matriz. Estos programas son ejecutados en una computadora con dos núcleos físicos cambiando el número de $threads$ de 1 hasta 4. Al hablar de $threads$ nos referimos secuencias de tareas encadenadas que pueden ser ejecutadas por un sistema operativo. Ya que el interés es analizar el tiempo de ejecución más que las operaciones citadas, al declarar los vectores y matrices a operar y al asignar los espacios de memoria, estos no se inicializarán sino que se dejará el valor que exista en ese momento en ese espacio de memoria. Para demostrar que las operaciones son correctas al ser realizadas por un número distinto de $threads$ se calcula la norma del vector generado para demostrar que en todos los casos es el mismo.

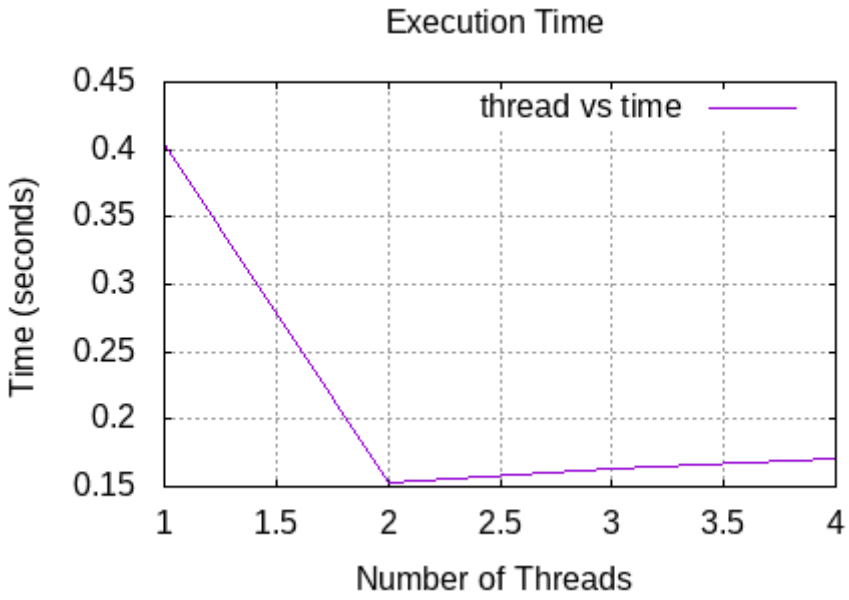
Suma de vectores en paralelo

Al ejecutar la suma de vectores en paralelo se obtienen los siguientes resultados para $1, 2, 3$ y 4 $threads$ utilizando vectores de 25000000 elementos:

Número de $threads$	Tiempo de ejecución (segundos)
1	0.403504
2	0.152566
3	0.163188
4	0.171261

Al comparar el resultado de la norma para cada uno de los vectores calculados con diferente número de $threads$ se aprecia que estos coinciden.

La información presentada en la tabla puede apreciarse en el siguiente gráfico.



Se puede apreciar que existe una diferencia notable entre utilizar 1 $thread$ y 2. Ya que el ordenador en el que fueron ejecutados los programas consta únicamente de 2 $cores$ físicos, al configurar la ejecución con 3 y 4 $cores$ el tiempo de ejecución aumenta con respecto a 2, sin embargo el tiempo en comparación a 1 solo $thread$ sigue siendo más bajo.

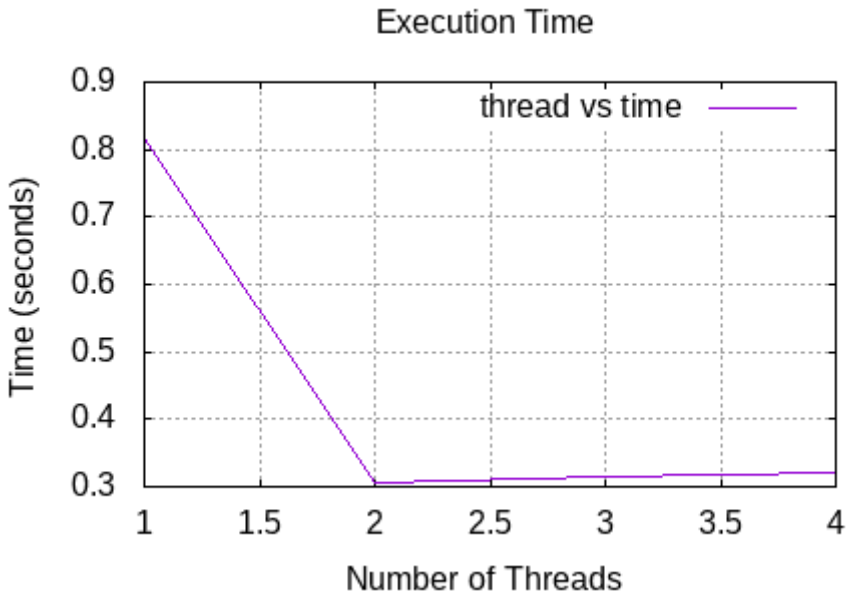
Multiplicación de vectores elemento a elemento

Al ejecutar la multiplicación de vectores elemento a elemento en paralelo se obtienen los siguientes resultados para $1, 2, 3$ y 4 $threads$ utilizando vectores de 25000000 elementos:

Número de $threads$	Tiempo de ejecución (segundos)
1	0.817798
2	0.306872
3	0.316136
4	0.321213

Al comparar el resultado de la norma para cada uno de los vectores calculados con diferente número de $threads$ se aprecia que estos coinciden.

La información presentada en la tabla puede apreciarse en el siguiente gráfico.



Se puede apreciar que existe una diferencia notable entre utilizar 1 $thread$ y 2. Ya que el ordenador en el que fueron ejecutados los programas consta únicamente de 2 $cores$ físicos, al configurar la ejecución con 3 y 4 $cores$ el tiempo de ejecución aumenta con respecto a 2, sin embargo el tiempo en comparación a 1 solo $thread$ sigue siendo más bajo.

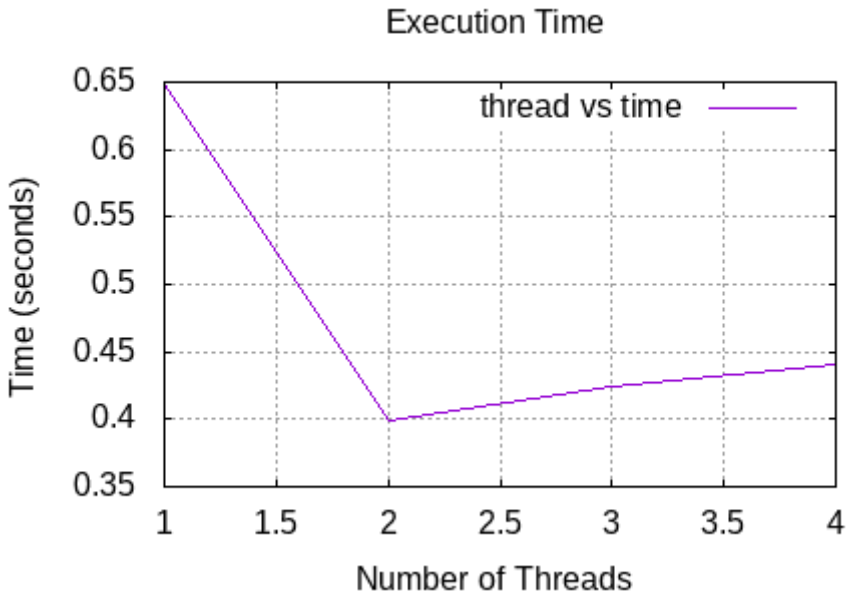
Producto punto

Al ejecutar el producto punto en paralelo se obtienen los siguientes resultados para $1, 2, 3$ y 4 $threads$ utilizando vectores de 25000000 elementos:

Número de $threads$	Tiempo de ejecución (segundos)
1	0.647863
2	0.399077
3	0.423622
4	0.440699

Al comparar el resultado del producto punto para cada uno de los vectores calculados con diferente número de $threads$ se aprecia que estos coinciden.

La información presentada en la tabla puede apreciarse en el siguiente gráfico.



Se puede apreciar que existe una diferencia notable entre utilizar 1 $thread$ y 2. Ya que el ordenador en el que fueron ejecutados los programas consta únicamente de 2 $cores$ físicos, al configurar la ejecución con 3 y 4 $cores$ el tiempo de ejecución aumenta con respecto a 2, sin embargo el tiempo en comparación a 1 solo $thread$ sigue siendo más bajo.

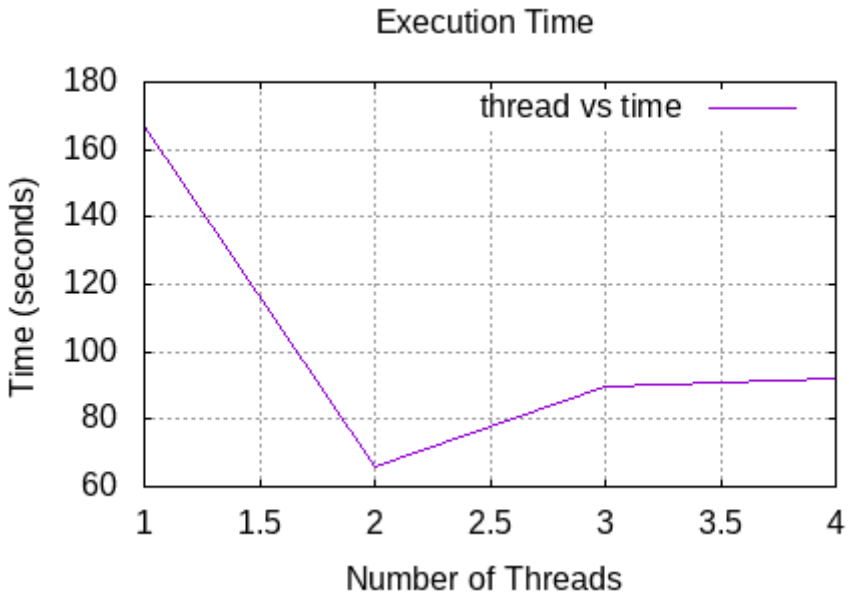
Multiplicación matriz por vector

Al ejecutar la multiplicación de una matriz por un vector en paralelo se obtienen los siguientes resultados para $1, 2, 3$ y 4 $threads$ utilizando una matriz cuadrada de 100000 filas y un vector de ese mismo tamaño:

Número de $threads$	Tiempo de ejecución (segundos)
1	166.643987
2	65.743499
3	89.704300
4	92.017953

Al comparar el resultado de la norma para cada uno de los vectores calculados con diferente número de $threads$ se aprecia que estos coinciden.

La información presentada en la tabla puede apreciarse en el siguiente gráfico.



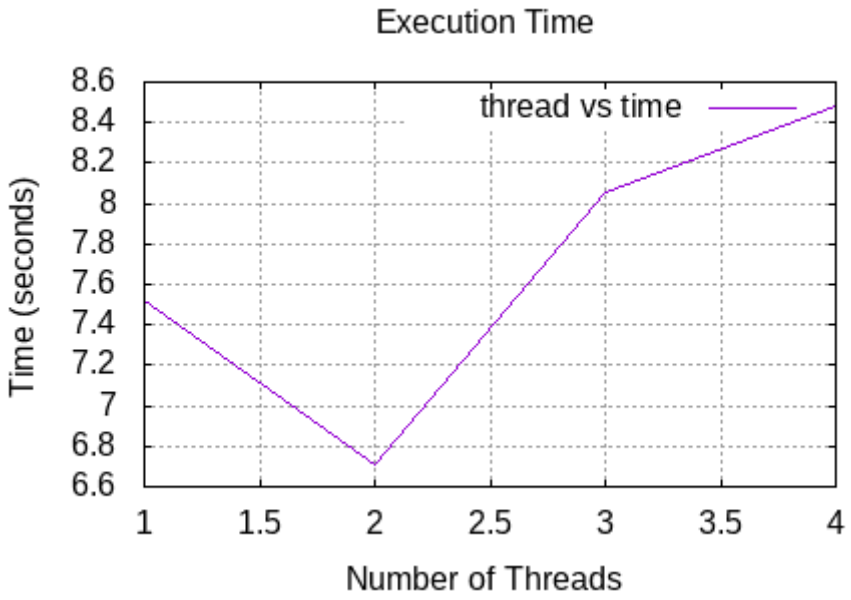
Se puede apreciar que existe una diferencia notable entre utilizar 1 $thread$ y 2. Ya que el ordenador en el que fueron ejecutados los programas consta únicamente de 2 $cores$ físicos, al configurar la ejecución con 3 y 4 $cores$ el tiempo de ejecución aumenta con respecto a 2, sin embargo el tiempo en comparación a 1 solo $thread$ sigue siendo más bajo.

Multiplicación de matrices

Al ejecutar la multiplicación de matrices en paralelo se obtienen los siguientes resultados para $1, 2, 3$ y 4 $threads$ utilizando dos matrices cuadradas de 800 filas:

Número de $threads$	**Tiempo de ejecución (segundos)**
1	7.517756
2	6.713129
3	8.050512
4	8.477012

La información presentada en la tabla puede apreciarse en el siguiente gráfico.



Se puede apreciar que existe una diferencia notable entre utilizar 1 $thread$ y 2. Ya que el ordenador en el que fueron ejecutados los programas consta únicamente de 2 $cores$ físicos, al configurar la ejecución con 3 y 4 $cores$ el tiempo de ejecución aumenta con respecto a 1 y con 2 cores.