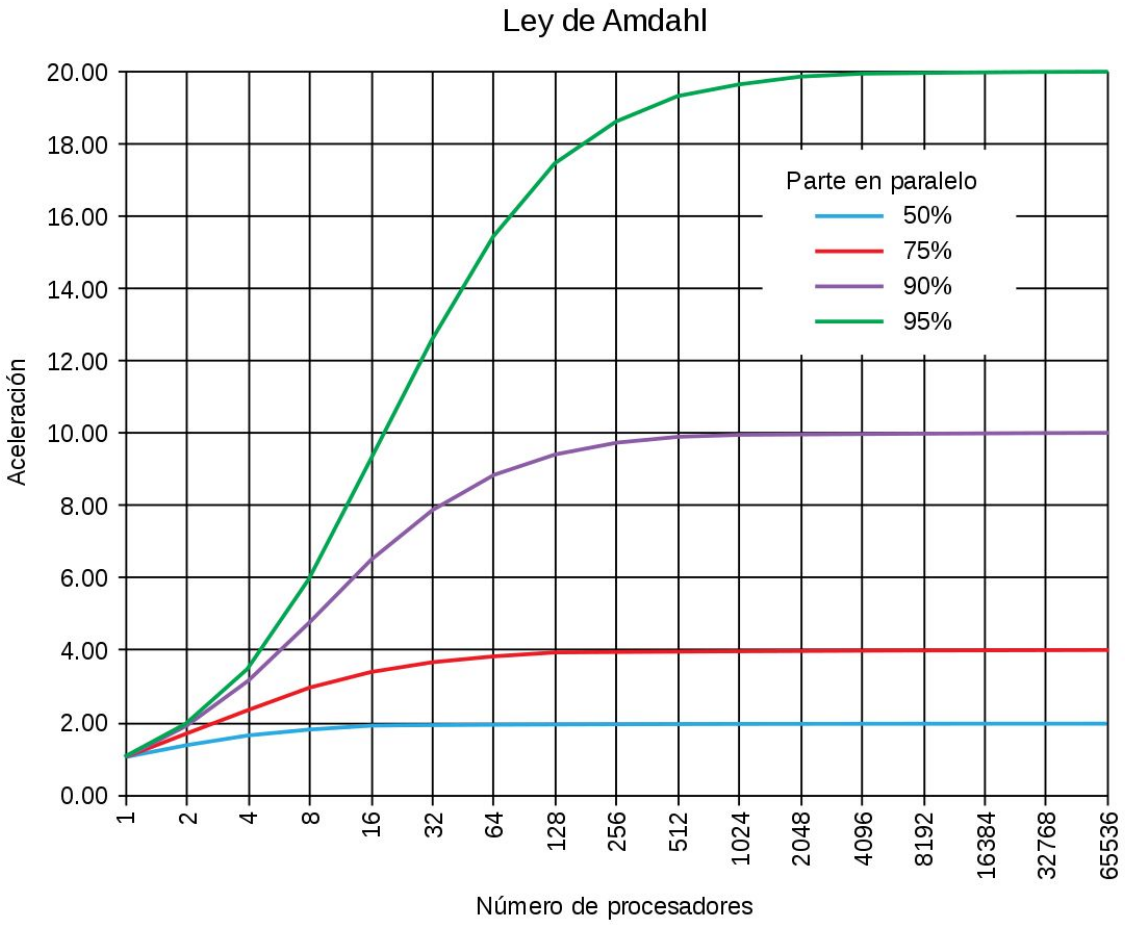
PRIMER INFORME DE HPC

Implementación y prueba de eficiencia para algoritmos secuencial y paralelo.



Karen Stefanny Lopez Segura

5 de septiembre de 20019

INTRODUCCIÓN

Con el fin de probar y comparar la eficiencia de algoritmos secuenciales y paralelos, así como también, averiguar la mejora máxima de los mismo, se realizó una aplicación de un algoritmo tanto secuencial como paralelo, en este caso se usaron hilos para la implementación del algoritmo paralelo; también se probó su mejora máxima utilizando la ley de Amdahl.

CONTENIDO

Para la comprobación de la eficacia, eficiencia y mejora máxima posible al resolver e implementar el problema de la multiplicación de matrices, se realizaron los siguientes pasos:

1. Implementación **de un algoritmo secuencial** para la multiplicación de 2 matrices de tamaño NXN.

   En este paso se realizó la aplicación de un algoritmo para la multiplicación de dos matrices de tamaño NXN, esto se realizó en el lenguaje c++ con un algoritmo de complejidad $O(n^3)$.

2. **Probar la efectividad** del algoritmo secuencial en la multiplicación de matrices.

   Se prueba la efectividad del algoritmo realizado, enviándole al programa tamaños de matrices pequeñas de las que podamos sacar la respuesta manualmente y hacer una comparación.

3. **Implementación de un algoritmo paralelo utilizando hilos** para realizar la multiplicación de una matriz NXN.

   Utilizando el lenguaje c++ y su librería *pthread.h* se realizó una implementación del algoritmo de multiplicación de matrices de complejidad $O(n^3)$ donde se utilizaron 4 hilos los cuales eran utilizados para multiplicar dos matrices de la siguiente manera:

   - cada matriz se divide en cuatro secciones para su multiplicación por bloques de filas.
   - los primeros 3 hilos tendrán la misma cantidad de filas para multiplicar y el último hilo multiplicará todas las filas restantes así estas sean más que

en los primeros 3 hilos.

4. **Probar la efectividad** del código para la multiplicación de matrices NXN utilizando hilos.

   De la misma manera que se hizo con el algoritmo secuencial en el paralelo se hicieron pruebas con matrices pequeñas y se comprobó manualmente si las respuestas fueron correctas.

5. **Realizar un shell script** para optimizar pruebas.

   Se realizó un shell script para facilitar la puesta en marcha de las ejecuciones.

6. **Medir los tiempos** de ejecución en los algoritmos realizados anteriormente y calcular su mejora máxima con la ley de Amdahl .

   Para sacar los tiempos de ejecución de los programas realizados primero se eligieron al azar 9 tamaños diferentes para las matrices que van a ser multiplicadas y se contó cuántos segundos tarda en ejecutar cada caso, dando los siguientes resultados:

Programa Secuencial

tamaño de matriz	tiempo de ejecución
10	6,50470E-05
60	8,86471E-03
100	2,49777E-02
300	5,84765E-01
500	3,78031E+00
700	1,09277E+01
1000	3,57289E+01
1200	7,11370E+01
2000	3,39982E+02

Programa Paralelo

tamaño de matriz	tiempo de ejecución
10	0,000939958
60	0,003275790
100	0,010500900
300	0,226093000
500	1,298230000
700	3,353540000
1000	11,936500000
1200	20,653200000
2000	96,910600000

Para poder tener mayor precisión en los datos, se repitió este proceso otras 9 veces y se promediaron los valores de tiempo de ejecución resultantes, se obtuvieron estos datos:

Programa Secuencial

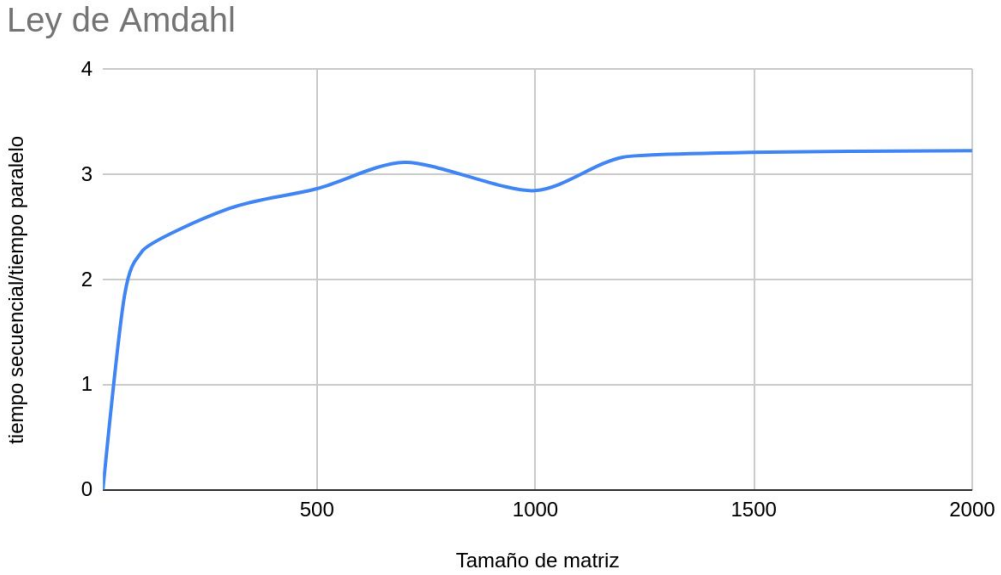
tamaño de matriz	tiempo de ejecución
10	3,67740E-05
60	5,62741E-03
100	2,37912E-02
300	6,89849E-01
500	4,14377E+00
700	1,18956E+01
1000	3,81635E+01
1200	7,39954E+01
2000	3,49842E+02

Programa Paralelo

tamaño de matriz	tiempo de ejecución
10	0,012610036
60	0,003023149
100	0,010472183
300	0,257427900
500	1,445123000
700	3,815371000
1000	13,395220000
1200	23,364150000
2000	108,343460000

Una vez tenemos los tiempos promedios pasamos a graficar la siguiente ecuación basada en la ley de Amdahl:

$$\frac{\textit{tiempo de ejecución secuencial}}{\textit{tiempo ejecución paralelo}}$$



CONCLUSIÓN

Los algoritmos paralelos como se evidencio en las tablas de tiempo de ejecución son considerablemente más rápidos que los algoritmos secuenciales, pero aun así, entre mayor se vaya volviendo el tamaño de los datos, menor es el aumento en la velocidad de ejecución, hasta llegar a un punto donde simplemente deja de mejorar la velocidad de las ejecuciones y termina con una misma velocidad que un programa secuencial.

ANEXOS:

-codigo, información del equipo donde se realizó la ejecución y datos completos de las ejecuciones: https://github.com/karen-lopez/hpc