Práctica 03

Programación Concurrente y Paralela (PCP) # GIITIN

11/08/22

Rodríguez López, Alejandro # UO281827

Tabla de Contenidos

Tabla de Figuras

Tabla de Funciones

Introducción

Las capacidades de un sistema no son ilimitadas. El tiempo de ejecución puede dispararse muy fácilmente con el algoritmo adecuado, claramente esta es una situación que debe evitarse a todo riesgo, y por ello es importante conocer las diferentes estrategias algorítmicas para diseñar programas óptimos.

Sin embargo, en muchos casos puede ser de interés ir más allá y reducir los tiempos de ejecución de estos programas. Para ello, los procesadores actuales poseen varios núcleos, y en ellos hilos que se pueden utilizar de forma paralela unos con otros para completar una tarea común.

Esta estrategia conocida como paralelismo queda documentada en este documento como parte de la entrega de la práctica 3 de Programación Concurrente y Paralela.

Rodríguez López, Alejandro.

UO281827.

Análisis Teórico.

Práctica 1.

Cálculo del TPP de cada sistema.

$$TPP_{dp} = chasis \cdot nodos_{chasis} \cdot sockets_{nodo} \cdot cores_{socket} \cdot clock_{GHz} \cdot \frac{n^o\, flop}{ciclo}_{dp}$$

Function 1: TPP_{dp}

De acuerdo con ésta fórmula y sabiendo los datos de cada procesador, podemos calcular el TPP de cada sistema:

$TPP_{i3} = 1*1*1*2*3.1*8 = 49.6\,(GFlop)$

$TPP_{Ryzen} = 1*1*1*8*3.6*16 = 460.8\,(GFlop)$

$TPP_{Xeon} = 1*1*2*4*2.4*8 = 153.6\,(GFlop)$

Sabiendo el TPP (número de flops por segundo) del sistema, podemos conocer el tiempo que tarda en realziar un flop.

$$t_c = \frac{1}{TPP_{dp}}$$

Function 2: t_c

Table 1: TPP_{dp}

T_c Paralelo	T_c Secuencial
$t_{ci3} = \frac{1}{TPP_{i3} \cdot 10^9} = 2.01613\text{E-}11\,(seg)$	$t_{ci3} = \frac{2}{TPP_{i3} \cdot 10^9} = 4.0323\text{E-}11\,(seg)$
$t_{cRyzen} = \frac{1}{TPP_{Ryzen} \cdot 10^9} = 2.1701\text{E-}12\,(seg)$	$t_{cRyzen} = \frac{8}{TPP_{Ryzen} \cdot 10^9} = 1.7361\text{E-}12\,(seg)$
$t_{cXeon} = \frac{1}{TPP_{Xeon} \cdot 10^9} = 6.5104\text{E-}12\,(seg)$	$t_{cXeon} = \frac{8}{TPP_{Xeon} \cdot 10^9} = 5.2083\text{E-}11\,(seg)$

Complejidad temporal y espacial de los problemas.

La complejidad temporal del problema $C = \beta C + \alpha AB$ sería $T(n,m,k) = n \cdot m \cdot (2+k) \cdot t_c$.

La complejidad temporal del problema transponer una $matriz_{nxm}$ es $T(n,m) = n \cdot m \cdot t_c$.

Respecto a las complejidades espaciales, para el problema $C = \beta C + \alpha AB$ sería $n \cdot m + m \cdot k + n \cdot k$ y la del problema de transponer una $matriz_{nxm}$ sería $n \cdot m$.

Práctica 3.

Complejidades.

Se deja constancia de que todas las complejidades calculadas a continuación no tienen en cuenta el cálculo de la matriz traspuesta.

La complejidad de MyDGEMM al utilizar `#pragma omp parallel for`, se reduce a:

$$T(n,m,k,p)=\frac{m}{p}\cdot n\cdot(2+k)\cdot t_c$$

Function 3: MyDGEMM

La complejidad de MyDGEMMT es equivalente a la de MyDGEMM por cómo se han organizado sus tareas, en el caso de MyDGEMMT se ha paralelizado por columnas, por lo que se reduce a:

$$T(n,m,k,p)=\frac{n}{p}\cdot m\cdot(2+k)\cdot t_c$$

Function 4: MyDGEMMT

La complejidad de MyDGEMMB depende de el tamaño de las matrices y del tamaño del bloque (*blk*) en el que éstas sean divididas, en ésta práctica se ha tomado una décima parte del tamaño como tamaño del bloque. Entonces, la complejidad se reduce a:

$$T(n,m,k,p,blk)=\frac{n\cdot m\cdot k}{blk^3}\cdot\frac{blk^2}{p}\cdot(2+blk)\cdot t_c=\frac{n\cdot m\cdot k}{blk\cdot p}\cdot(2+blk)\cdot t_c=\frac{10\cdot n^2}{p}\cdot(2+\frac{n}{10})\cdot t_c=\frac{20\cdot n^2+n^3}{p}\cdot t_c=\frac{n^2\cdot(20+n)}{p}\cdot t_c$$

Function 5: MyDGEMMB

O en términos más genéricos ya que $n=m=k$ y el tamaño del bloque (λ) podría ser cualquier otro valor divisor de *m*, *n* y *k*:

$$T(n,p,\lambda)=\frac{n^2\cdot(\lambda+n)}{p}\cdot t_c$$

Function 6: MyDGEMMB Bis

Speedups.

Para calcular el *Speedup* de un sistema utilizamos $S(p)=\frac{1}{p}$ siendo p el número de hilos del ordenador.

Por ello, los *speedup* de las máquinas utilizadas en prácticas sería:

- i3: $\frac{1}{2}=0.5$, el sistema i3 sería el doble de rápido si se utilizan todos sus hilos.
- Ryzen: $\frac{1}{8}=0.125$, el sistema Ryzen sería ocho veces más rápido si se utilizan todos sus hilos.
- Xeon: $\frac{1}{8}=0.125$, el sistema Xeon sería ocho veces más rápido si se utilizan todos sus hilos.

Las utilizaciones anteriores son suponiendo que el problema es balanceado y no hay comunicaciones.

Tablas.

Python.

Table 2: Python i3.

M	N	K	Empírico
1000	1001	999	9.52124E-02
2000	2001	1999	7.20527E-01
3000	3001	2999	2.39482E+00

Table 3: Python Ryzen.

M	N	K	Empírico
1000	1001	999	3.95281E-02
2000	2001	1999	2.66782E-01
3000	3001	2999	8.81437E-01

Table 4: Python Xeon.

M	N	K	Empírico
1000	1001	999	2.33121E-01
2000	2001	1999	1.78903E+00
3000	3001	2999	6.00941E+00

MyDGEMM.

i3.

Table 5: MyDGEMM i3 Secuencial.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	4.0323E-11	4.04E-02	3.92045E+00	6.01172E-01
2000	2001	1999	8.01E+09	4.0323E-11	3.23E-01	3.09193E+01	4.47876E+00
3000	3001	2999	2.70E+10	4.0323E-11	1.09E+00	1.04149E+02	1.59972E+01

Table 6: MyDGEMM i3 Paralelo.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	2.0161E-11	2.02E-02	2.07663E+00	4.67559E-01
2000	2001	1999	8.01E+09	2.0161E-11	1.61E-01	1.55578E+01	3.18711E+00
3000	3001	2999	2.70E+10	2.0161E-11	5.45E-01	5.33643E+01	1.18693E+01

Ryzen.

Table 7: MyDGEMM Ryzen Secuencial.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	2.1701E-12	1.74E-03	2.55021E+00	1.40990E-01
2000	2001	1999	8.01E+09	2.1701E-12	1.39E-02	2.06225E+01	2.90934E+00
3000	3001	2999	2.70E+10	2.1701E-12	4.69E-02	6.99707E+01	9.60528E+00

Table 8: MyDGEMM Ryzen Paralelo.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	1.7361E-11	2.17E-03	3.44029E-01	2.72094E-02
2000	2001	1999	8.01E+09	1.7361E-11	1.74E-02	2.75589E+00	7.46803E-01
3000	3001	2999	2.70E+10	1.7361E-11	5.86E-02	9.45080E+00	2.72924E+00

Xeon.

Table 9: MyDGEMM Xeon Secuencial.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	6.5101E-12	5.22E-02	6.77470E+00	6.61624E-01
2000	2001	1999	8.01E+09	6.5101E-12	4.17E-01	5.55177E+01	7.60962E+00
3000	3001	2999	2.70E+10	6.5101E-12	1.41E+00	1.86966E+02	3.12989E+01

Table 10: MyDGEMM Xeon Paralelo.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	5.2083E-11	6.52E-03	8.57141E-01	1.01605E-01
2000	2001	1999	8.01E+09	5.2083E-11	5.21E-02	6.98273E+00	2.59760E+00
3000	3001	2999	2.70E+10	5.2083E-11	1.76E-01	2.40591E+01	8.53595E+00

MyDGEMMT.

Table 11: MyDGEMMT Ryzen Paralelo.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1001	999	1.00E+09	1.7361E-11	2.17E-03	5.84983E-01	2.66100E-02
2000	2001	1999	8.01E+09	1.7361E-11	1.74E-02	4.66382E+00	7.36234E-01
3000	3001	2999	2.70E+10	1.7361E-11	2.70E+10	1.59156E+01	2.72349E+00

MyDGEMMB.

Table 12: MyDGEMMB Ryzen Secuencial.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1000	1000	1.02E+09	2.1701E-12	1.77E-03	2.69226E+00	5.40702E-01
2000	2000	2000	8.08E+09	2.1701E-12	1.40E-02	2.15109E+01	4.34962E+00
3000	3000	3000	2.72E+10	2.1701E-12	4.72E-02	7.25988E+01	1.46601E+01

Table 13: MyDGEMMB Ryzen Paralelo.

M	N	K	Nº Flop	Tc	Teórico	Empírico O0	Empírico O3
1000	1000	1000	1.02E+09	1.7361E-11	2.17E-03	3.68673E-01	3.43643E-02
2000	2000	2000	8.08E+09	1.7361E-11	1.74E-02	2.92983E+00	2.41841E-01
3000	3000	3000	2.72E+10	1.7361E-11	5.86E-02	9.56684E+00	8.03686E-01

Discusión.

Tras realizar numerosas mediciones y varios estudios teóricos sobre los sistemas que se han utilizado para las ejecuciones así como sobre los algoritmos en sí, se han llegado a los siguientes datos y conclusiones.

Teóricamente, las ejecuciones secuenciales llevan más tiempo que las paralelas. Esta norma se cumple incluso entre distintos ordenadores, a excepción del i3, cuyo tiempo de ejecución paralelo es mayor que el del ryzen en secuencial.

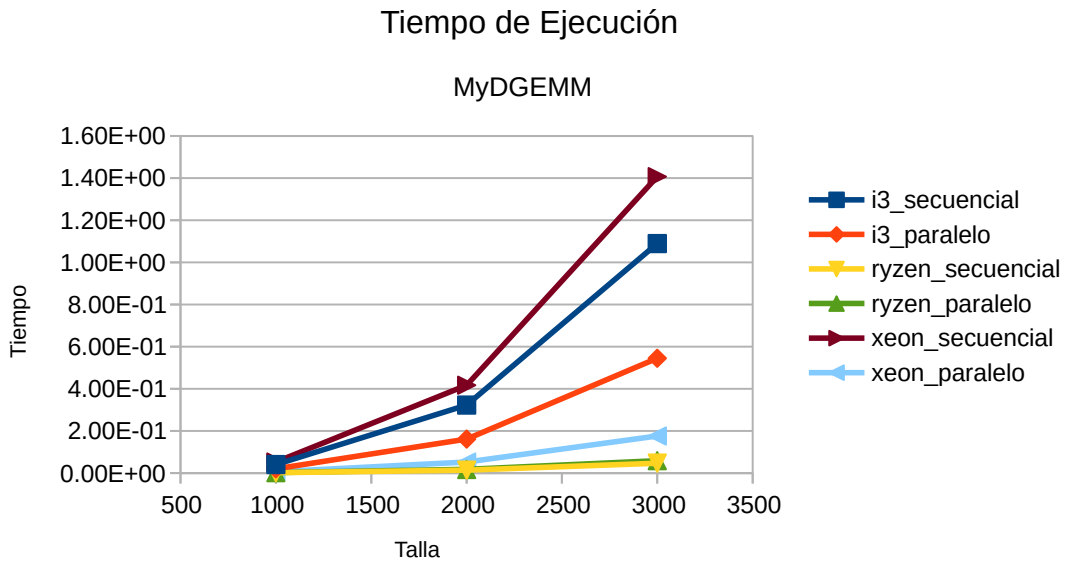

Figure 1: MyDGEMM Teórico.

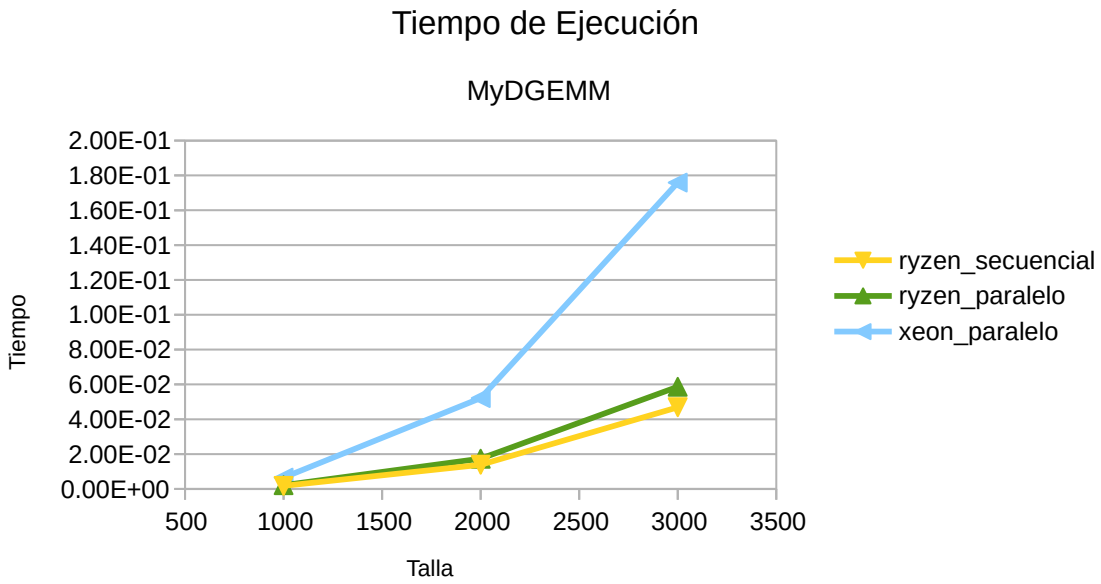

Figure 2: MyDGEMM Teórico Bis.

De los sistemas, el que más parecería beneficiarse del paralelismo sería claramente el Xeon, que gracias a la paralelización del algoritmo, reduce su tiempo en un 7’99% aproximadamente esto se debe al uso del segundo socket.

Si observamos entonces los datos empíricos, obtenemos unos resultados similares, pero con un mayor tiempo de ejecución del esperado, ésta es una moda que se repetirá en la mayoría de resultados.

Dicho aumento en el tiempo de respuesta puede deberse a que los calculos analíticos hayan sido realizados con fuentes optimistas sobre el hardware o fallos en la implementación del algoritmo. Otras razones menores pueden ser el uso que debe hacer el operativo de los componentes para organizar el sistema, el estado de los mismos o su desgaste.

Las diferencias entre los sistemas son, lógicamente, explicadas por su hardware, son sistemas diferentes con memorias y procesadores distintos.

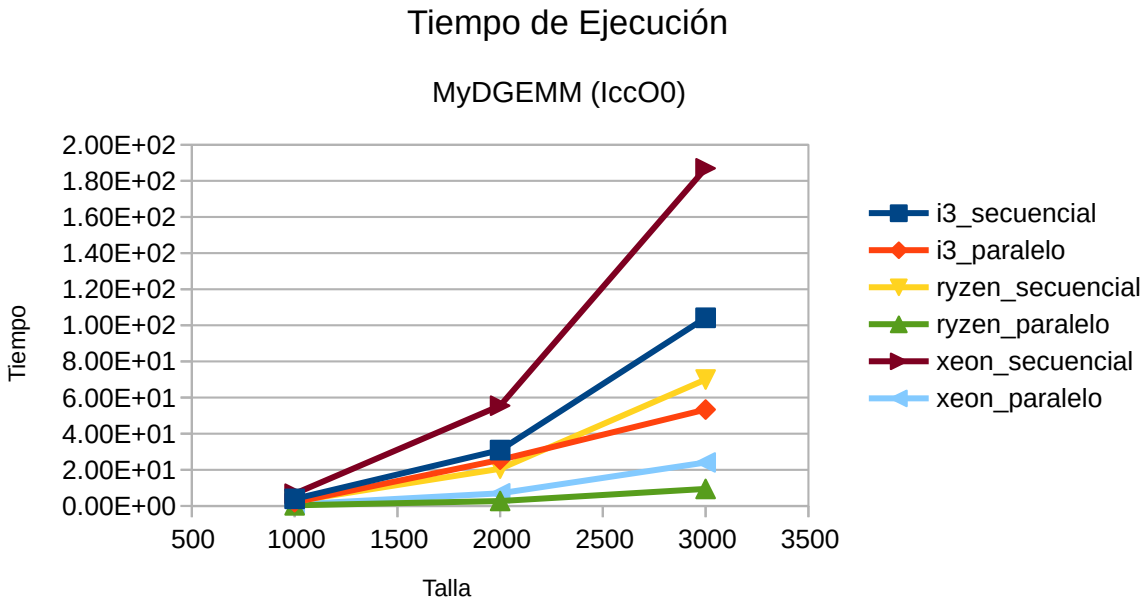

Figure 3: MyDGEMM Empírico IccO0.

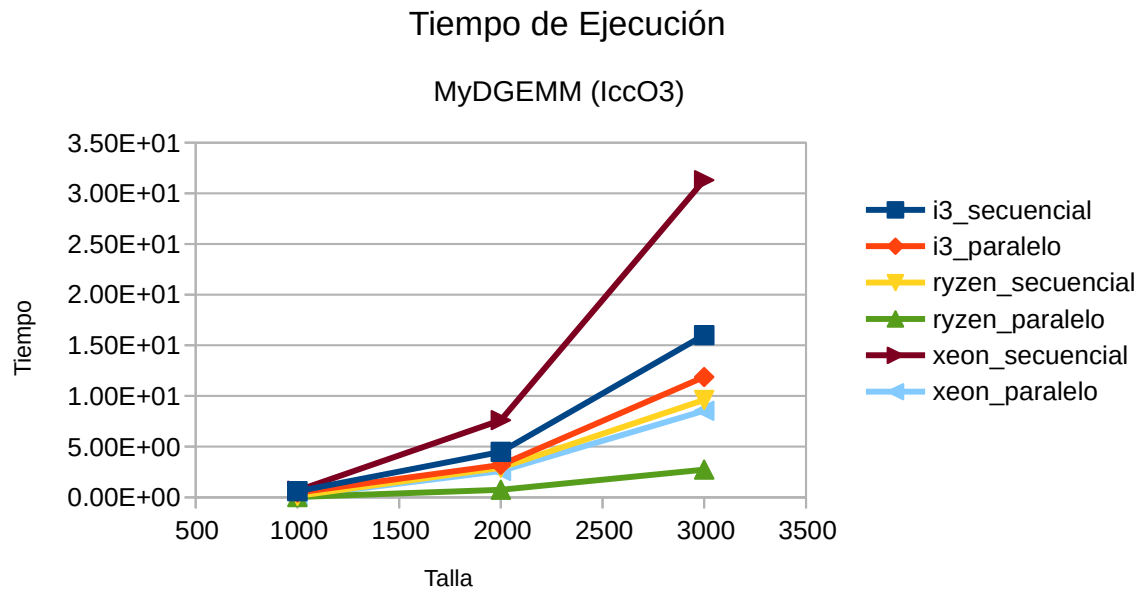

Figure 4: MyDGEMM Empírico IccO3.

Si buscamos revisar el comportamiento del Xeon para comprobar la hipótesis que obtuvimos en los cálculos analíticos, observaremos que tiene una aceleración del 7’81% y 3.52% en IccO0 e IccO3 respectivamente.

A continuación se observa la gráfica analítica del tiempo de ejecución para el algoritmo MyDGEMMB. Como era de esperar, el tiempo secuencial del Ryzen supera al paralelo (0’79% de aceleración).

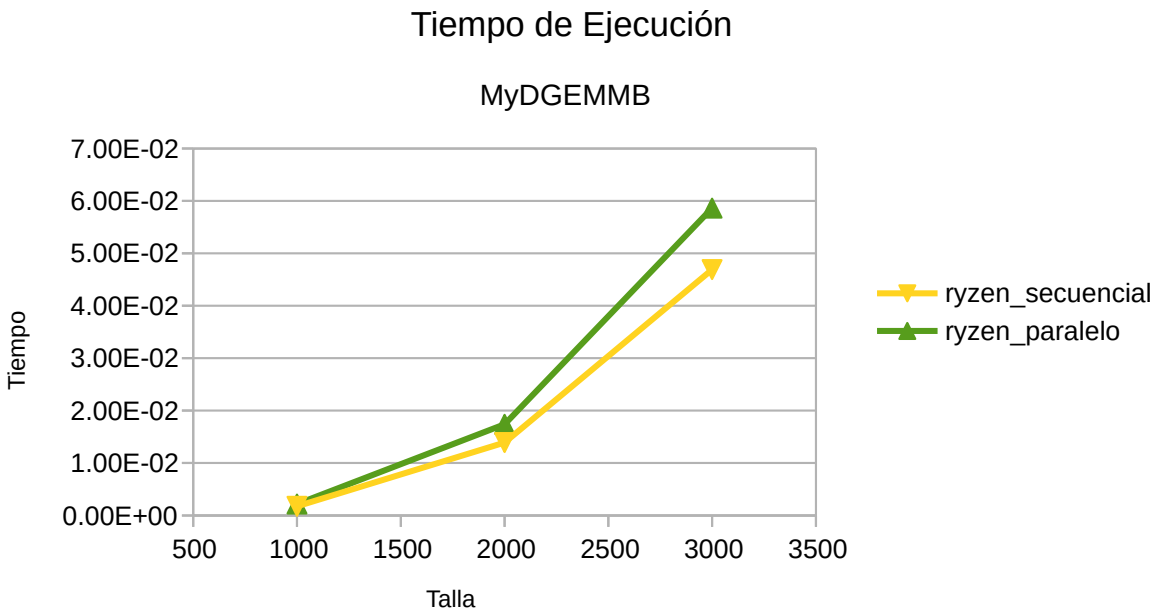

Figure 5: MyDGEMMB Teórico.

Si comparamos con los datos empíricos, observamos que en realidad, el algoritmo secuencial es casi tan rápido como el paralelo cuando se comparan las versiones IccO3 e IccO0 respectivamente de los mismos.

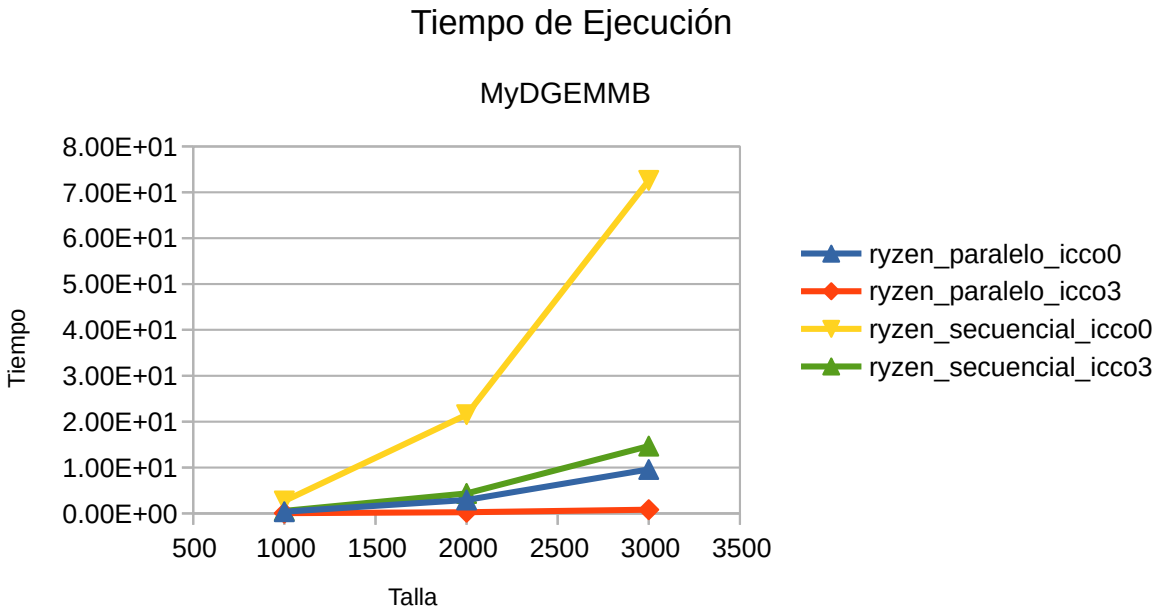

Figure 6: MyDGEMMB Empírico.

Si comparamos las distintas aceleraciones, obtenemos:

1. En secuencial, el IccO3 frente al IccO0 tiene una aceleración de 4’95%.
2. En Paralelo, el IccO3 frente al IccO0 tiene un acelreación de 11’91%.
3. En IccO0, el paralelo frente al secuencial tiene una aceleración de 7’52%.
4. En IccO3, el paralelo frente al secuencial tiene una aceleración de 18’10%.

Finalmente, observamos el tiempo de ejecución de la función MyDGEMMT, en la que podemos observar la aceleración de IccO3 frente a IccO0 una vez más. Dicha aceleración sería 6’07%.

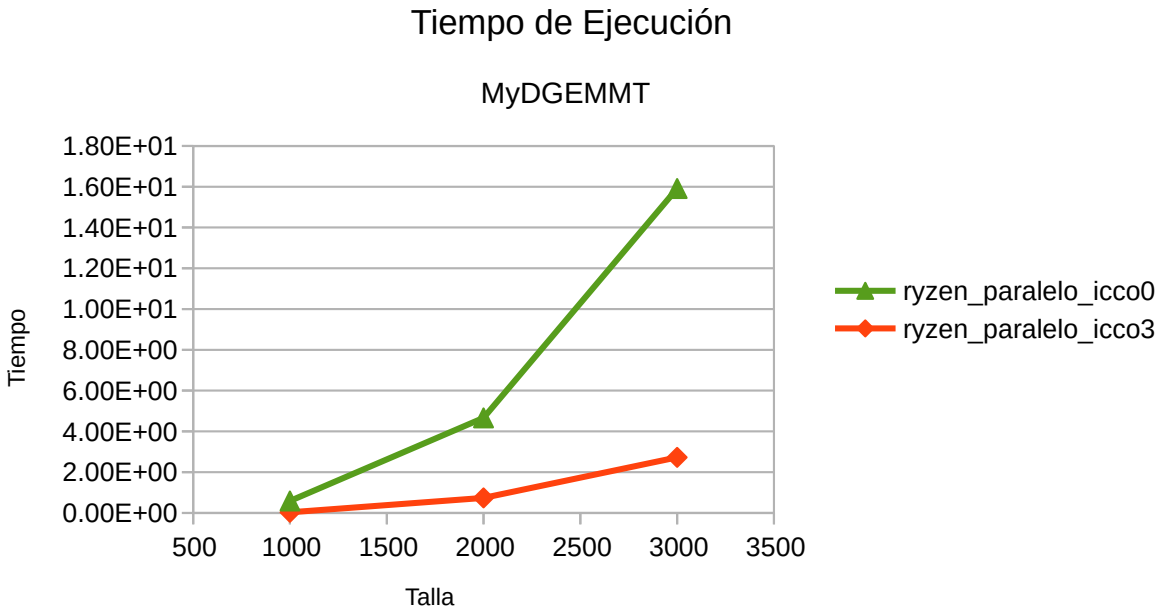

Figure 7: MyDGEMMT Empírico.

Conclusión.

Para concluir, se ha visto a lo largo de varios ejemplos que dos programas que sean eficaces, puden tener muy distintas eficiencias en función de cómo sean programados.

A pesar de que es cierto que el hardware del sistema toma una gran importancia en lo que respecta a las capacidades máximas del mismo, es posible que un ordenador tenga notables diferencias si el software que trata de ejecutar hace un buen uso del paralelismo.