Algoritmos Paralelos:

Transformada Rápida de Fourier

Calderon Jimenez David

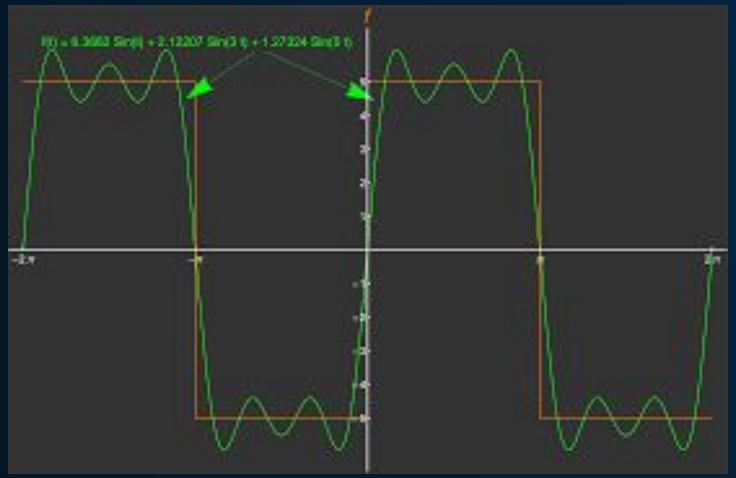
Hernandez Olvera Humberto Ignacio

Ordiales Caballero Iñaky

Equipo 4

Serie Trigonométrica de Fourier

$$f(t) = \frac{a(0)}{2} + \sum_{n=1}^{\infty} \left[a(n) \cos \frac{n\pi t}{L} + b(n) \sin \frac{n\pi t}{L} \right]$$



$$\begin{aligned} a(0) &= \frac{1}{L} \int_{-L}^{L} f(t)\, dx \\ a(n) &= \frac{1}{L} \int_{-L}^{L} f(t) \cos \frac{n\pi t}{L} dx \\ b(n) &= \frac{1}{L} \int_{-L}^{L} f(t) \sin \frac{n\pi t}{L} dx \end{aligned}$$

Serie Compleja de Fourier

$$C(n) = \frac{a_n - ib_n}{2} = \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{T}{2}} f(t)\, e^{-inw_0 t} dt$$

$$f(t) = \sum_{n=1}^{\infty} \left[C(n)\, e^{inw_0 t} \right] + \sum_{n=-1}^{-\infty} \left[C(n)\, e^{inw_0 t} \right]$$

$$f(t) = \sum_{n=-\infty}^{\infty} \left[C(n)\, e^{inw_0 t} \right]$$

Transformada de Fourier

$$f_{(t)} = \frac{1}{2\pi}\int_{-\infty}^{+\infty} F(\omega)e^{j\omega t}d\omega \quad \rightarrow \quad Antitransformada\ de\ Fourier$$

$$F(\omega) = \int_{-\infty}^{+\infty} f_{(t)}e^{-j\omega t}dt \quad \rightarrow \quad Transformada\ de\ Fourier$$

$$\hat{x}_k = \sum_{n=0}^{N-1}\left[x_n e^{\frac{-i2\pi}{N}kn}\right]$$

$$x_n = \sum_{n=0}^{N-1}\left[\hat{x}_n e^{\frac{i2\pi}{N}kn}\right]$$

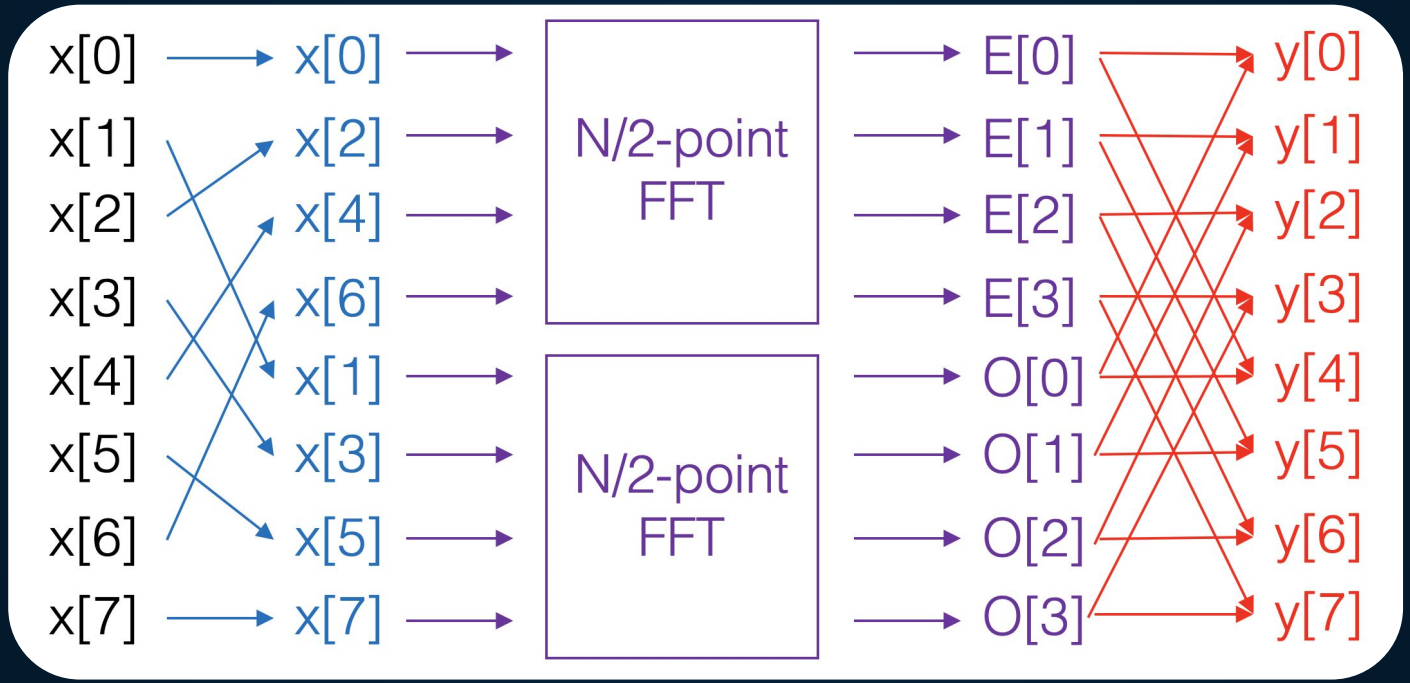
Transformada de Fourier Forma Vectorial

$$\begin{bmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \\ \dots \\ \hat{x}_n \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & w_n & w_n^2 & \dots & w_n^{n-1} \\ 1 & w_n^2 & w_n^4 & \dots & w_n^{2(n-1)} \\ . & . & . & . & . \\ 1 & w_n^{n-1} & w_n^{2(n-1)} & \dots & w_n^{(n-1)^2} \end{bmatrix}}_{Matriz\ de\ Transformacion\ Directa\ de\ Fourier\ \mathbf{DFT}} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix}$$

Transformada de Fourier Forma Vectorial Factorizada

$$\begin{bmatrix} \hat{x}_0 \\ \hat{x}_1 \\ \hat{x}_2 \\ \dots \\ \hat{x}_n \end{bmatrix} = \begin{bmatrix} [I_{\frac{n}{2}\times\frac{n}{2}}] & -[D_{\frac{n}{2}\times\frac{n}{2}}] \\ [I_{\frac{n}{2}\times\frac{n}{2}}] & -[D_{\frac{n}{2}\times\frac{n}{2}}] \end{bmatrix} \begin{bmatrix} [DFT_{\frac{n}{2}\times\frac{n}{2}}] & 0 \\ 0 & [DFT_{\frac{n}{2}\times\frac{n}{2}}] \end{bmatrix} \begin{bmatrix} [x_{par}] \\ [x_{impar}] \end{bmatrix}$$

FFT (Fast Fourier Transform)



Aplicaciones

- Filtrado de señales (ondas electromagnéticas)
- Análisis de señales
  - Radio
  - GPS
  - Internet
- Comprimir imagenes
- Comprimir audio

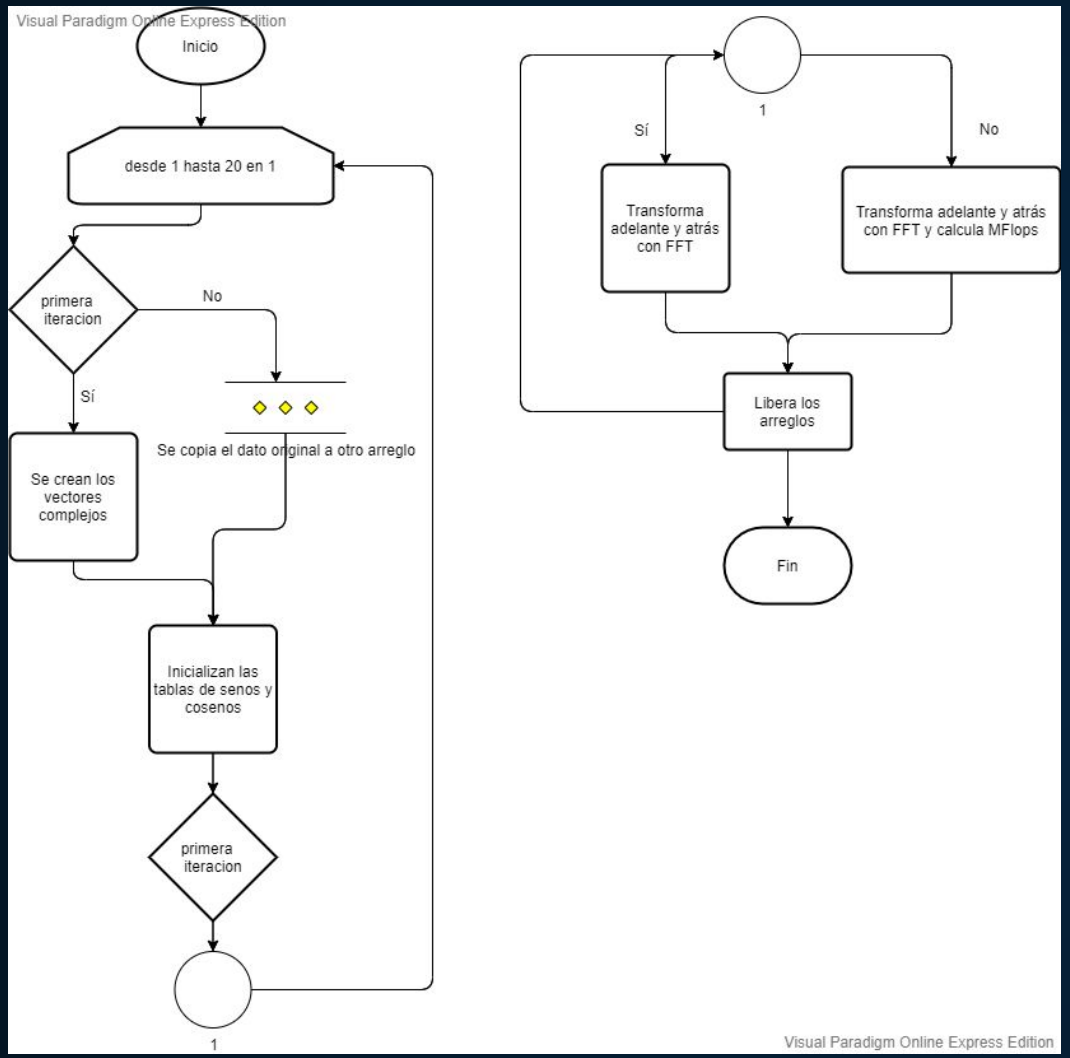
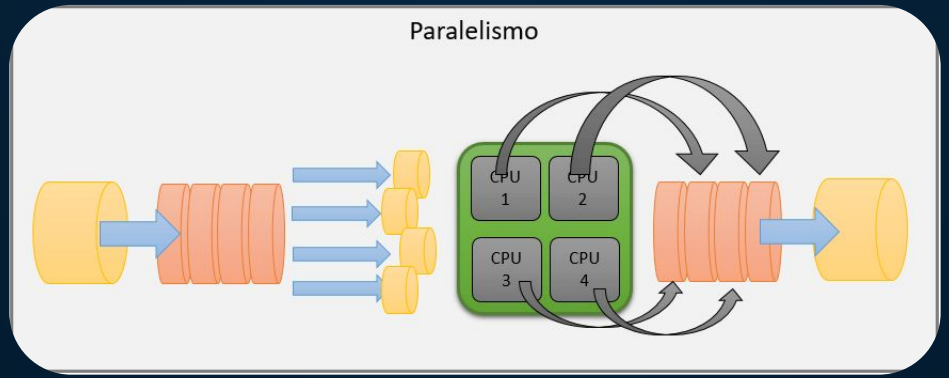


Diagrama de flujo del algoritmo FFT

Paralelización del algoritmo FFT

*Las subdivisiones de los vectores de datos.

*La multiplicación de matrices.

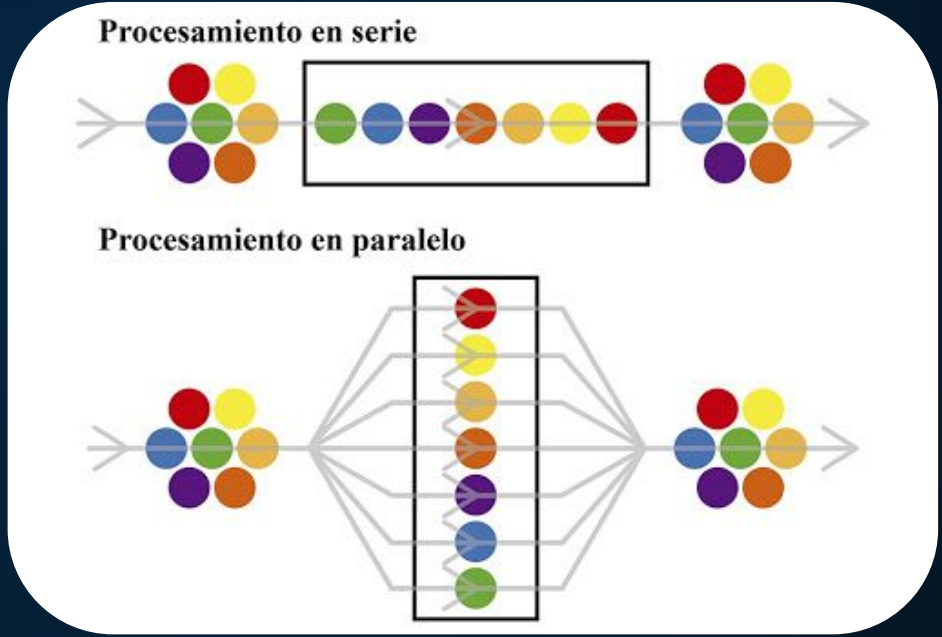


Posibles Resultados

*Mejoras en los tiempos de ejecución

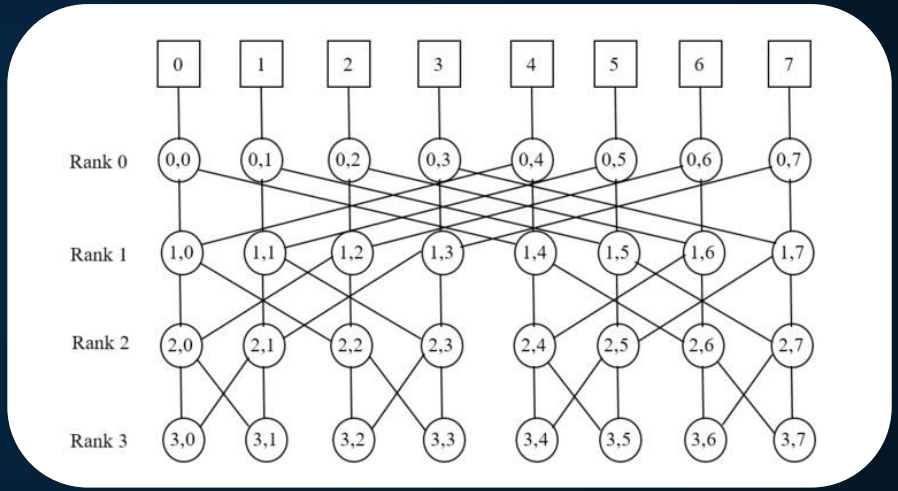
*Mejora en la eficiencia del programa

*Por consiguiente obtener una fracción serial muy baja



Topología de red de mariposa

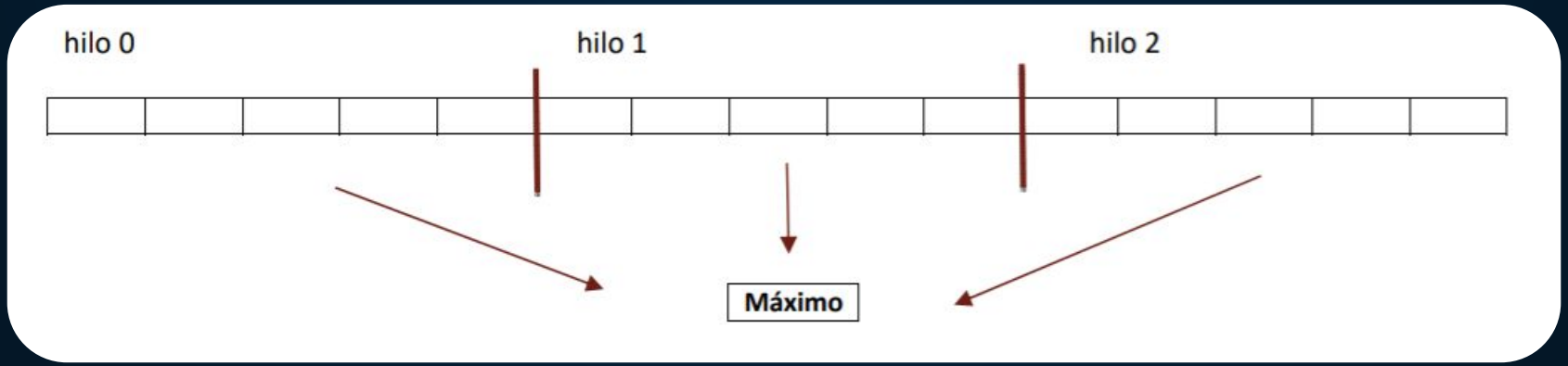
Se puede utilizar para conectar distintos nodos en un sistema de multiprocesador, la red interconectada que se utiliza para este sistema es de memoria compartida.



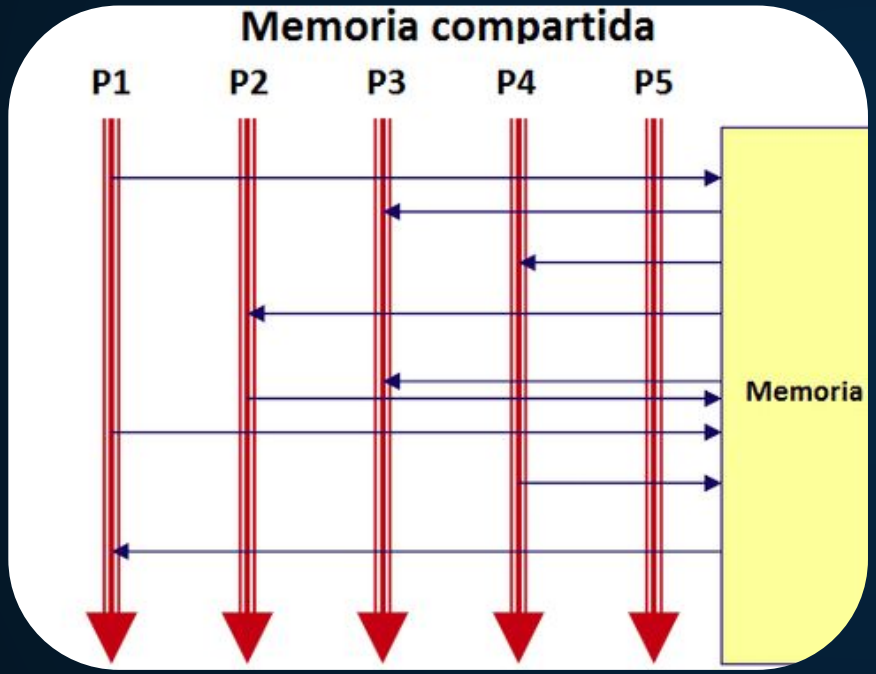
Tipo de paralelismo

Paralelismo de datos

Éste es un paradigma de la programación concurrente que consiste en subdividir el conjunto de datos de entrada a un programa, de manera que a cada procesador le corresponda un subconjunto de esos datos.



Forma de comunicación



El medio de comunicación es por medio de la memoria compartida, que es aquel tipo de memoria que puede ser accedida por múltiples programas, ya sea para comunicarse entre ellos o para evitar copias redundantes.

Granularidad

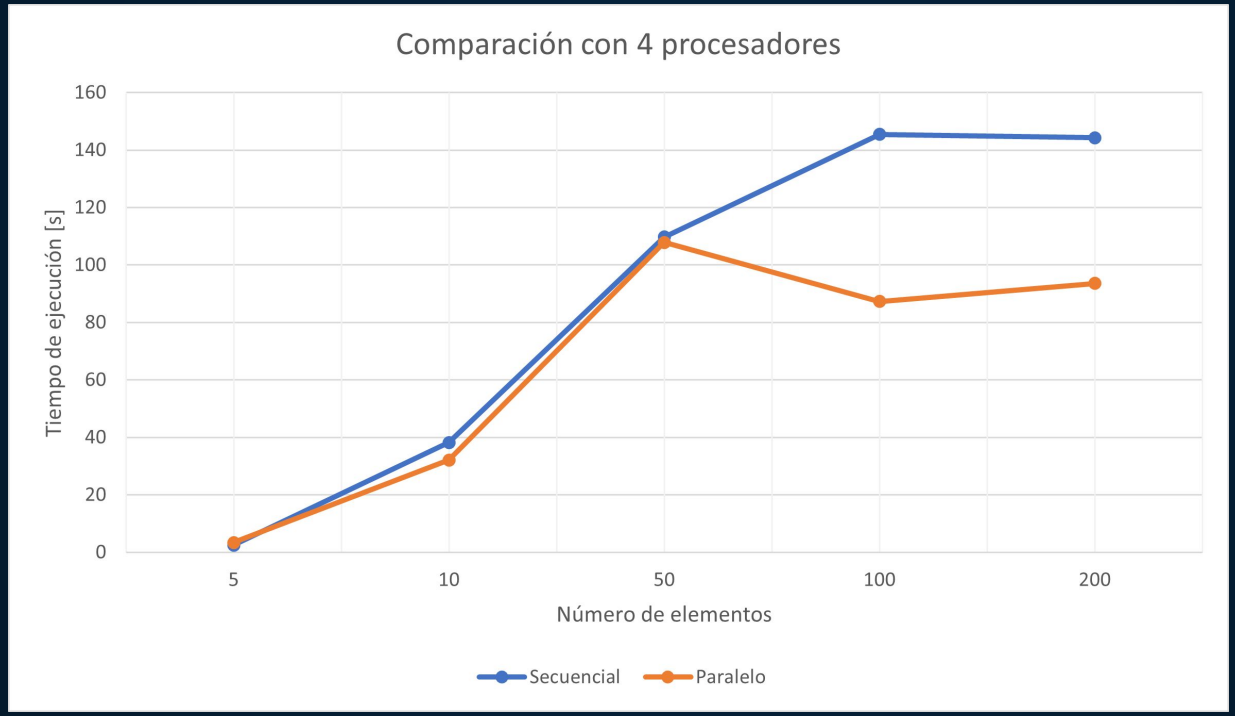
Gruesa

Fina

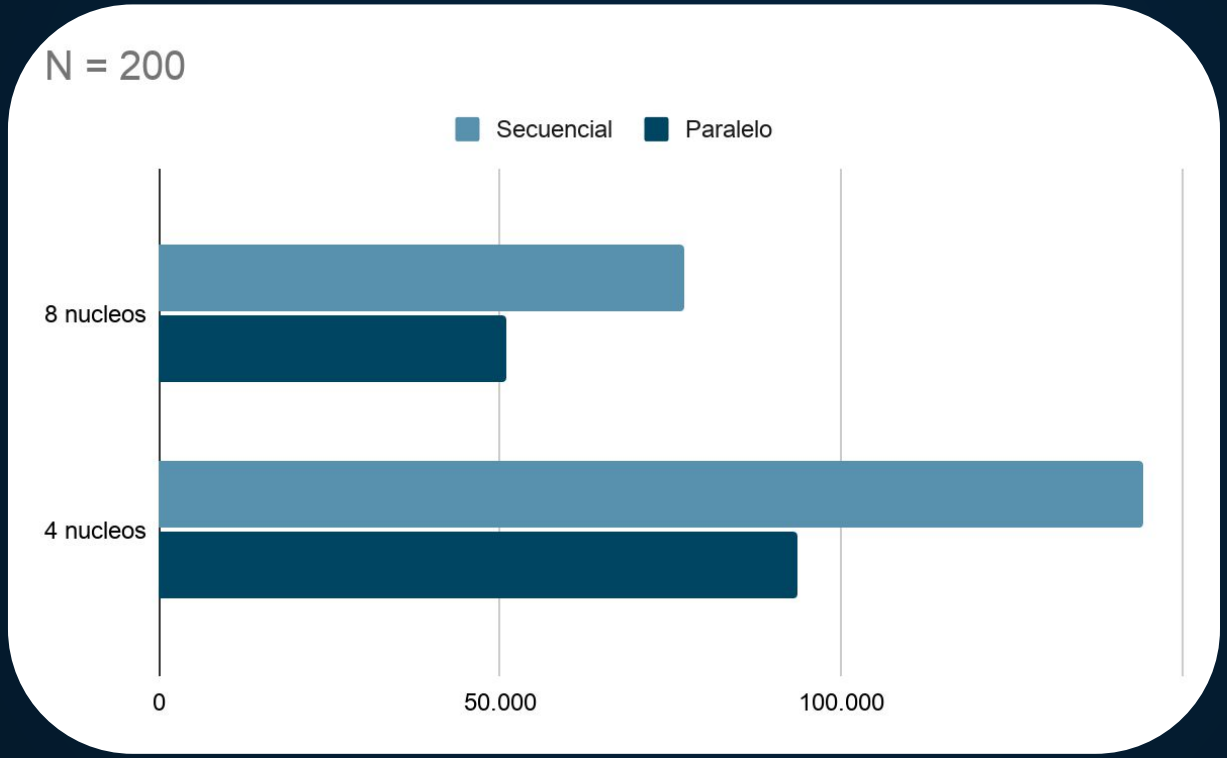
Más comunicación entre tareas

Mayor nivel de paralelismo

Comparación diferentes cantidades de datos

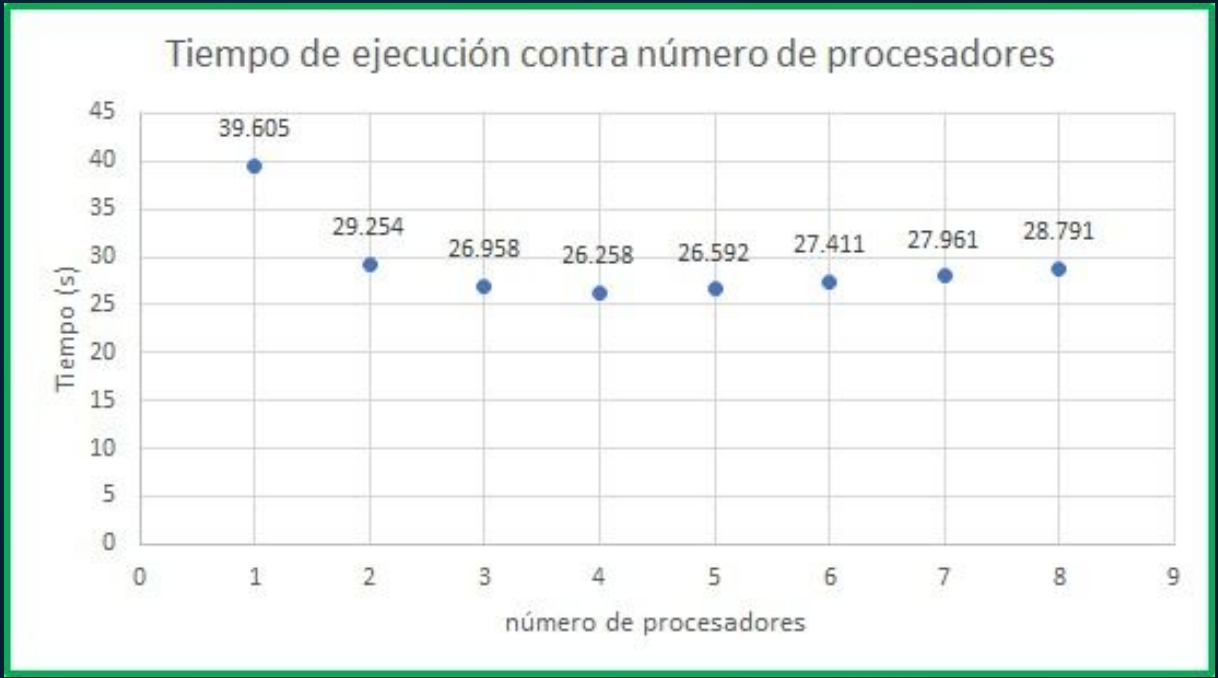


Tiempo de Ejecución

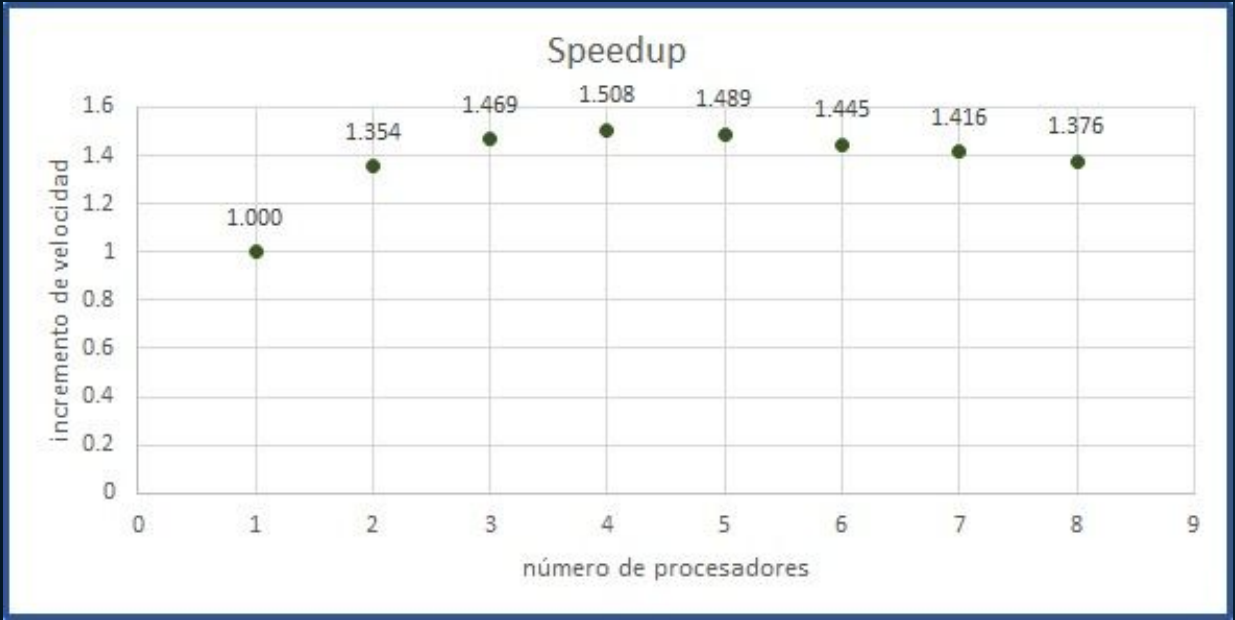


# procesadores	1	2	3	4	5	6	7	8
T(n) [s]	39.605	29.254	26.958	26.258	26.592	27.411	27.961	28.791

Tiempo de Ejecución



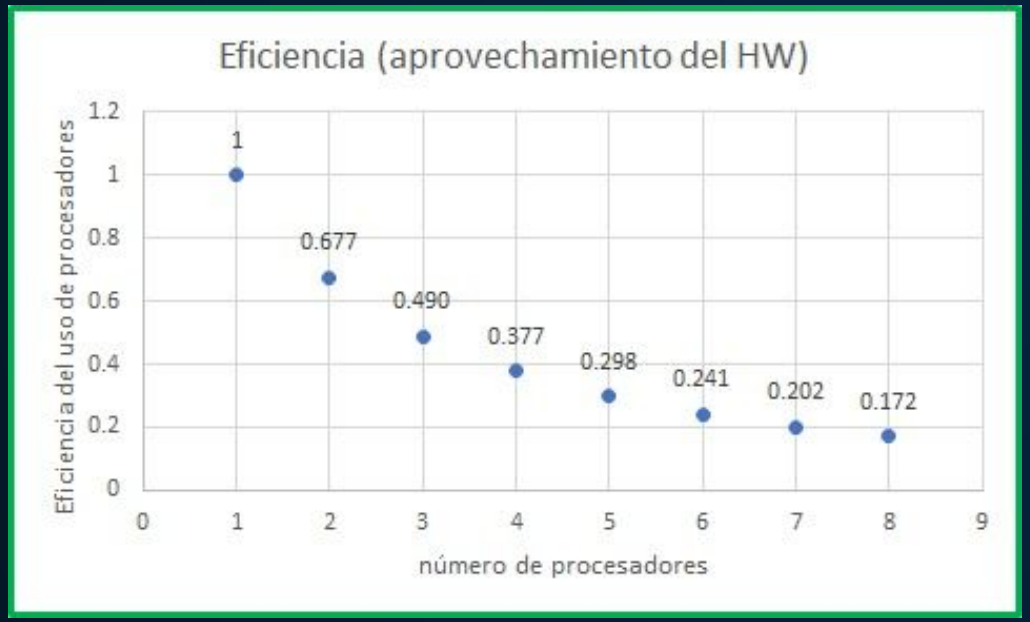
# procesadores	1	2	3	4	5	6	7	8
S(n)	**1**	**1.354**	**1.469**	**1.508**	**1.489**	**1.445**	**1.416**	**1.376**



Speedup

$$S(n) = \frac{T(1)}{T(n)}$$

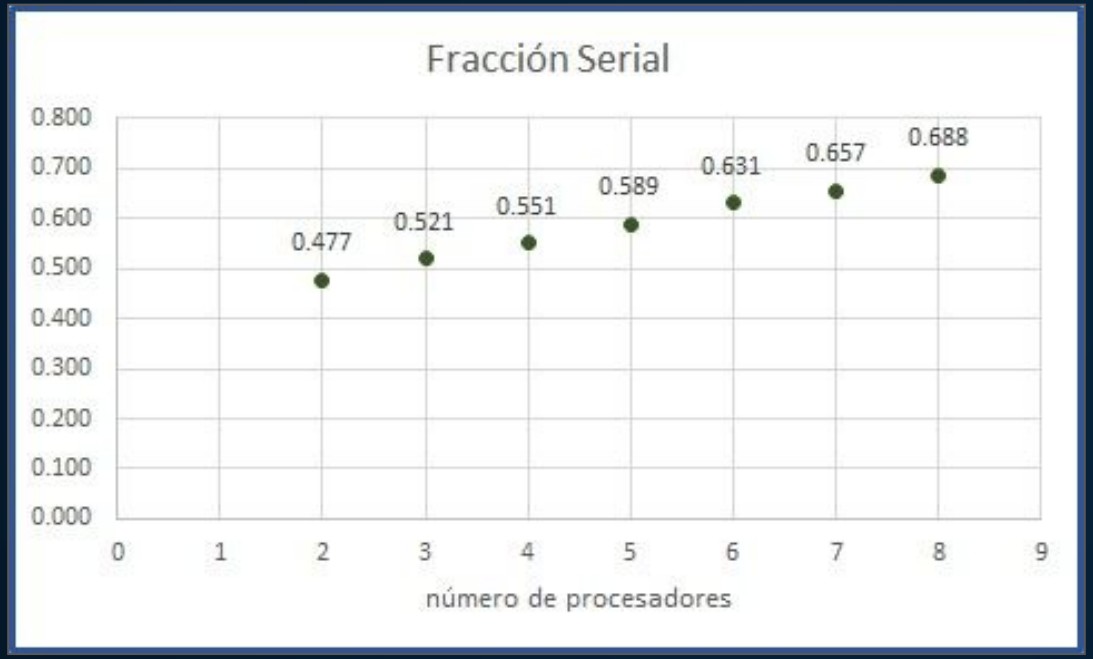
# procesadores	1	2	3	4	5	6	7	8
E(n)	**1**	**0.677**	**0.490**	**0.377**	**0.298**	**0.241**	**0.202**	**0.172**



Eficiencia

$$E(n) = \frac{T(1)}{n \cdot T(n)} = \frac{S(n)}{n}$$

# procesadores	1	2	3	4	5	6	7	8
f	.	0.477	0.521	0.551	0.589	0.631	0.657	0.688



Fracción Serial

$$f = \frac{\frac{1}{s} - \frac{1}{n}}{1 - \frac{1}{n}}$$

Conclusiones

- La implementación en paralelo de este algoritmo, ha probado **ser mejor** en comparación con su versión secuencial.

- La **eficiencia de paralelismo** de un algoritmo depende casi tanto del hardware usado (la topología, número de procesadores, etc.) como en el algoritmo mismo.

- **Características de la FFT:**
  - **Tipo de paralelismo:** de datos
  - **Topología:** de red de mariposas
  - **Comunicación:** memoria compartida
  - **Granularidad:** fina
- El **mejor número de procesadores** va a depender de tus necesidades, recursos y cantidad de datos.