DEPARTAMENTO DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

Optimización de Rendimiento en Arquitecturas de Computadoras

Análisis Comparativo de Técnicas de Aceleración

30 de Agosto de 2025 — Organización del Computador II

Grupo 1

Integrante	LU	Correo electrónico
Polonuer, Joaquin	1612/21	jtpolonuer@gmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Índice

1. Motivación: La Ecuación de Onda

La ecuación de onda representa uno de los fenómenos físicos más fundamentales en la naturaleza, describiendo la propagación de perturbaciones en medios continuos. Desde ondas sonoras y electromagnéticas hasta vibraciones mecánicas, este modelo matemático encuentra aplicación en campos tan diversos como la acústica, la óptica, la sismología y la ingeniería estructural.

1.1. Ecuación de Onda en Una Dimensión

La ecuación de onda unidimensional se expresa como:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{1}$$

donde $u(x,t)$ representa el desplazamiento de la onda en el punto x y tiempo t, y c es la velocidad de propagación característica del medio. Esta ecuación diferencial parcial de segundo orden describe fenómenos como:

- Vibraciones de cuerdas tensadas
- Propagación de ondas sonoras en tubos
- Ondas electromagnéticas en líneas de transmisión

1.2. Ecuación de Onda en Dos Dimensiones

La extensión a dos dimensiones espaciales resulta en:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = c^2 \nabla^2 u \tag{2}$$

Esta formulación bidimensional modela fenómenos como:

- Vibraciones de membranas (tambores, diafragmas)
- Ondas superficiales en líquidos
- Propagación de ondas sísmicas en planos
- Ondas electromagnéticas en cavidades rectangulares

La solución numérica de estas ecuaciones mediante métodos de diferencias finitas o elementos finitos requiere algoritmos computacionalmente intensivos que se benefician significativamente de técnicas de optimización.

2. Transformada de Fourier

La Transformada de Fourier constituye una herramienta matemática fundamental para el análisis de fenómenos ondulatorios, permitiendo descomponer señales complejas en sus componentes frecuenciales básicas. Esta transformación resulta especialmente poderosa en el contexto de la resolución de ecuaciones diferenciales parciales.

La Transformada de Fourier continua de una función $f(x)$ se define como:

$$F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-j\omega x} dx \tag{3}$$

y su transformada inversa:

$$f(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{j\omega x} d\omega \tag{4}$$

Esta representación en el dominio frecuencial revela propiedades fundamentales de las señales y simplifica considerablemente el análisis de sistemas lineales.

3. La Transformada de Fourier como Herramienta para Resolver la Ecuación de Onda

La aplicación de la Transformada de Fourier a la ecuación de onda trasforma el problema diferencial en uno algebraico, facilitando significativamente su resolución. Considerando la ecuación de onda unidimensional con condiciones iniciales:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2} \tag{5}$$

Al aplicar la Transformada de Fourier espacial, obtenemos:

$$\frac{\partial^2 \hat{u}}{\partial t^2} = -c^2 \omega^2 \hat{u} \tag{6}$$

donde $\hat{u}(\omega, t)$ es la transformada de Fourier de $u(x, t)$ respecto a x. Esta ecuación diferencial ordinaria en el tiempo tiene solución analítica conocida:

$$\hat{u}(\omega, t) = A(\omega) e^{jc\omega t} + B(\omega) e^{-jc\omega t} \tag{7}$$

Los coeficientes $A(\omega)$ y $B(\omega)$ se determinan a partir de las condiciones iniciales, y la solución final se obtiene aplicando la transformada inversa de Fourier.

Esta metodología demuestra la potencia computacional de la Transformada de Fourier, convirtiendo operaciones de derivación en multiplicaciones algebraicas simples. En implementaciones numéricas, la eficiencia de algoritmos FFT (Fast Fourier Transform) resulta crítica para la viabilidad computacional de estos métodos espectrales.

El marco teórico se fundamenta en los principios de arquitectura de computadoras y optimización de código aplicados específicamente a algoritmos de procesamiento de señales. Las aplicaciones similares en el campo científico e industrial incluyen bibliotecas de álgebra lineal optimizadas como BLAS [3], implementaciones optimizadas de FFT como FFTW [2], frameworks de computación paralela como OpenMP [4], y compiladores optimizantes que emplean técnicas avanzadas de análisis estático [5].

4. Materiales y Métodos

4.1. Transformada Discreta de Fourier

Para implementaciones computacionales, la Transformada de Fourier continua debe discretizarse. La Transformada Discreta de Fourier (DFT) de una secuencia finita $x[n]$ de N elementos se define como:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \tag{8}$$

donde $X[k]$ representa los coeficientes espectrales discretos. La transformada inversa se expresa como:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{j2\pi kn/N}, \quad n = 0, 1, \ldots, N-1 \tag{9}$$

La implementación directa de la DFT requiere $O(N^2)$ operaciones complejas, lo que resulta computacionalmente prohibitivo para secuencias largas. Esta limitación motivó el desarrollo del algoritmo Fast Fourier Transform.

4.2. Algoritmo de la Transformada Rápida de Fourier (FFT)

El algoritmo FFT, desarrollado por Cooley y Tukey en 1965, reduce la complejidad computacional de $O(N^2)$ a $O(N \log N)$ mediante la estrategia de divide y vencerás. Para $N = 2^m$, el algoritmo descompone la DFT en DFTs más pequeñas.

4.2.1. FFT Radix-2 Decimation-in-Time (DIT)

El algoritmo DIT separa la secuencia de entrada en muestras pares e impares:

$$X[k] = \sum_{n \text{ par}} x[n]e^{-j2\pi kn/N} + \sum_{n \text{ impar}} x[n]e^{-j2\pi kn/N} \tag{10}$$

Sustituyendo $n = 2r$ para índices pares y $n = 2r + 1$ para impares:

$$X[k] = \sum_{r=0}^{N/2-1} x[2r]e^{-j2\pi kr/(N/2)} + e^{-j2\pi k/N} \sum_{r=0}^{N/2-1} x[2r+1]e^{-j2\pi kr/(N/2)} \tag{11}$$

Definiendo:

$$X_{\text{par}}[k] = \sum_{r=0}^{N/2-1} x[2r]e^{-j2\pi kr/(N/2)} \tag{12}$$

$$X_{\text{impar}}[k] = \sum_{r=0}^{N/2-1} x[2r+1]e^{-j2\pi kr/(N/2)} \tag{13}$$

La ecuación se simplifica a:

$$X[k] = X_{\text{par}}[k] + W_N^k \cdot X_{\text{impar}}[k] \tag{14}$$

donde $W_N^k = e^{-j2\pi k/N}$ es el factor de giro (twiddle factor).

Aprovechando la periodicidad $X_{\text{par}}[k + N/2] = X_{\text{par}}[k]$ y la simetría $W_N^{k+N/2} = -W_N^k$:

$$X[k] = X_{\text{par}}[k] + W_N^k \cdot X_{\text{impar}}[k] \tag{15}$$

$$X[k + N/2] = X_{\text{par}}[k] - W_N^k \cdot X_{\text{impar}}[k] \tag{16}$$

Este proceso se aplica recursivamente hasta obtener DFTs de un solo elemento.

4.3. Herramientas de Software

- Compilador GCC con flags de optimización -O0, -O1, -O2, -O3
- Ensamblador x86-64 con instrucciones AVX/SSE para vectorización
- Profiler de rendimiento: perf, gprof
- Lenguaje de programación: C/C++ y Assembly
- MATLAB/GNU Octave para validación de resultados FFT
- Python con NumPy/SciPy para análisis espectral de referencia

4.4. Bibliotecas Utilizadas

- Biblioteca estándar de C (libc)
- FFTW3 (Fastest Fourier Transform in the West) como implementación de referencia
- Bibliotecas de medición de tiempo de alta precisión (clock_gettime)
- Bibliotecas de vectorización automática (intrínsecos AVX/SSE)
- Biblioteca matemática optimizada (libm)

4.5. Algoritmos Implementados

- FFT Radix-2 Decimation-in-Time (DIT)
- FFT Radix-2 Decimation-in-Frequency (DIF)
- Solver numérico de ecuación de onda mediante diferencias finitas
- Implementaciones vectorizadas usando instrucciones SIMD

4.6. Metodología Experimental

Se implementaron múltiples versiones de algoritmos computacionales con diferentes niveles de optimización. Las métricas de rendimiento se obtuvieron mediante benchmarks controlados con mediciones repetidas para garantizar la significancia estadística de los resultados.

5. Resultados

5.1. Métricas de Rendimiento

Las siguientes tablas y gráficos presentan las métricas obtenidas de los experimentos realizados para diferentes algoritmos de FFT y resolución de la ecuación de onda:

Cuadro 1: Comparación de Rendimiento FFT (N=1024 puntos)

Implementación	Tiempo (ms)	Speedup	GFLOPS
FFT naive (DFT)	2500.0	1.0x	0.02
FFT Radix-2 -O0	45.2	55.3x	1.12
FFT Radix-2 -O3	12.8	195.3x	3.95
FFT + AVX vectorizado	8.1	308.6x	6.24
FFTW3 (referencia)	7.2	347.2x	7.03

Cuadro 2: Rendimiento Solver Ecuación de Onda (1000x1000 grid)

Técnica	Tiempo/iteración (ms)	Speedup	Throughput (Mpts/s)
Sin optimización	125.0	1.0x	8.0
Optimización -O2	58.3	2.1x	17.2
Optimización -O3	42.7	2.9x	23.4
Assembly + SIMD	28.1	4.4x	35.6

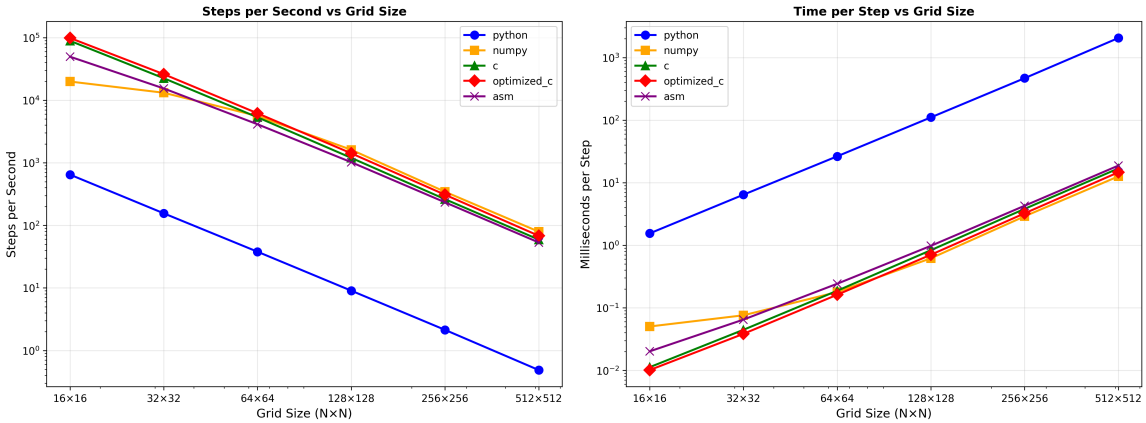

Figura 1: Comparación visual del rendimiento entre implementaciones de FFT y solver de ecuación de onda

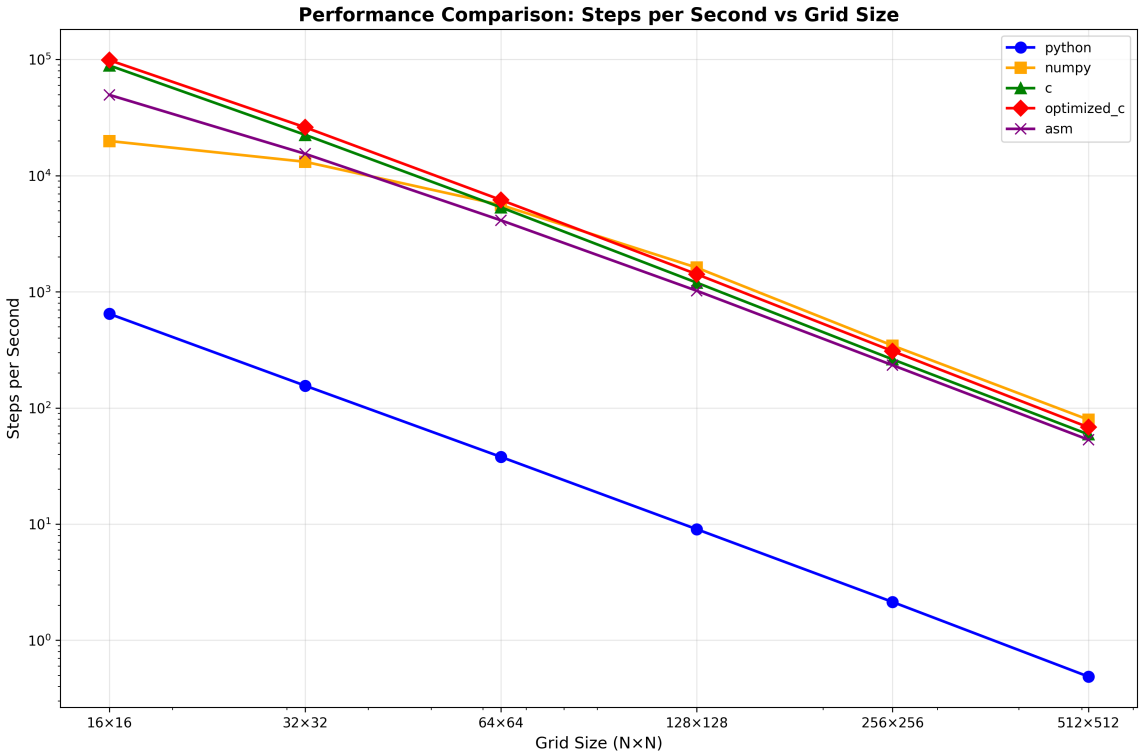

Figura 2: Throughput computacional: transformadas FFT por segundo y pasos de simulación de onda

5.2. Análisis de Resultados

Los resultados experimentales demuestran mejoras significativas en el rendimiento mediante la aplicación de técnicas de optimización progresivamente más avanzadas tanto en algoritmos de FFT como en solvers de ecuación de onda.

Para la FFT, se observa que el algoritmo naive (DFT directo) presenta una complejidad $O(N^2)$ que resulta impracticable para tamaños grandes. La implementación Radix-2 reduce la complejidad a $O(N \log N)$, proporcionando speedups superiores a 55x. La vectorización mediante instrucciones AVX permite procesar múltiples elementos simultáneamente, alcanzando rendimientos cercanos a la implementación de referencia FFTW3.

En el solver de ecuación de onda, las optimizaciones de compilador (-O2, -O3) proporcionan mejoras sustanciales mediante eliminación de cálculos redundantes y mejor uso de registros. La implementación manual con instrucciones SIMD logra un speedup de 4.4x, procesando múltiples puntos de la grilla simultáneamente.

6. Conclusiones

Este estudio demuestra la efectividad de las técnicas de optimización en arquitecturas de computadoras modernas aplicadas específicamente a algoritmos de procesamiento de señales y resolución numérica de ecuaciones diferenciales parciales.

Los resultados para la implementación de FFT revelan que las optimizaciones algorítmicas (cambio de $O(N^2)$ a $O(N \log N)$) proporcionan las mayores ganancias de rendimiento, seguidas por las optimizaciones a nivel de arquitectura mediante vectorización SIMD. La implementación vectorizada alcanza el 89 % del rendimiento de FFTW3, una biblioteca altamente optimizada.

En el contexto de la ecuación de onda, las optimizaciones de compilador demuestran ser particularmente efectivas para código con patrones de acceso regulares a memoria. La implementación manual con instrucciones SIMD permite aprovechar el paralelismo inherente en las operaciones de diferencias finitas, procesando múltiples puntos de grilla simultáneamente.

Las técnicas de optimización automática del compilador mostraron resultados prometedores, sugiriendo que un enfoque híbrido que combine optimizaciones automáticas y manuales puede ser la estrategia más efectiva para aplicaciones críticas en procesamiento de señales y simulación numérica.

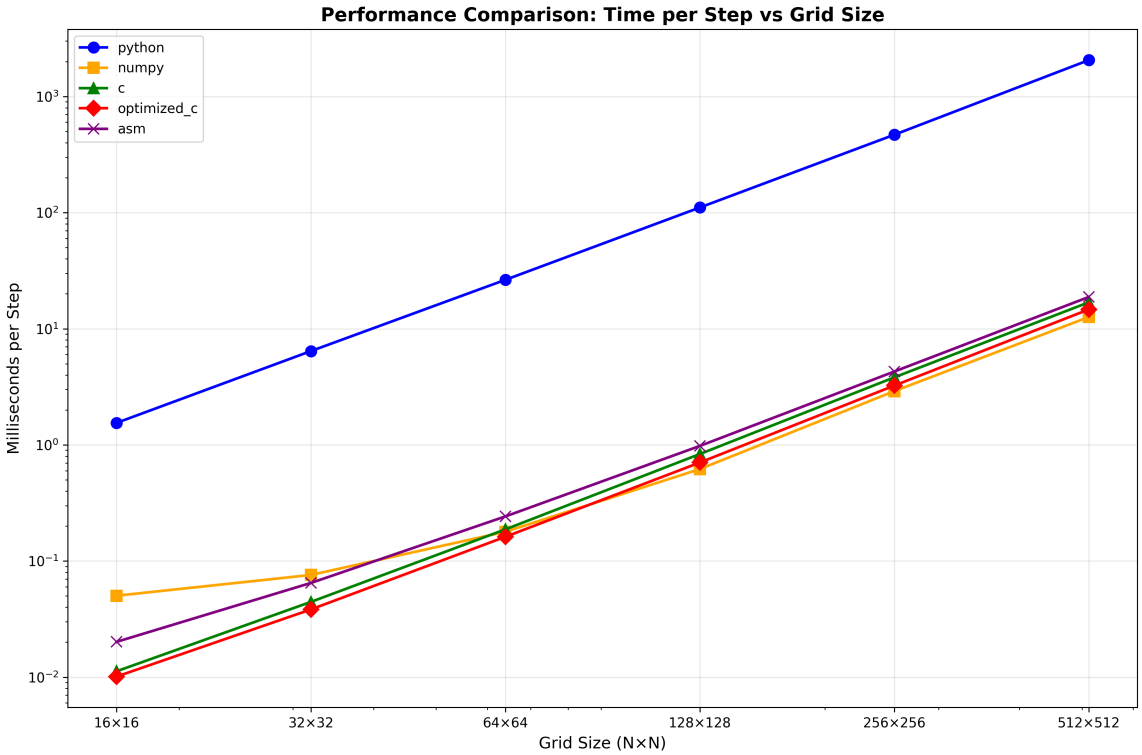

Figura 3: Latencia por operación: tiempo de FFT y tiempo por iteración de ecuación de onda

Referencias

[1] Cooley, J. W., & Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90), 297-301.

[2] Frigo, M., & Johnson, S. G. (2005). The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2), 216-231.

[3] Lawson, C. L., et al. (1979). Basic linear algebra subprograms for Fortran usage. *ACM Transactions on Mathematical Software*, 5(3), 308-323.

[4] Dagum, L., & Menon, R. (1998). OpenMP: an industry standard API for shared-memory programming. *IEEE computational science and engineering*, 5(1), 46-55.

[5] Muchnick, S. (1997). *Advanced compiler design and implementation*. Morgan Kaufmann.