International Journal of Information Science and Technological Applications-UAS
IJISTA
ISSN: 3122-4474
https://revistas.uas.edu.mx/index.php/IJISTA

Junio 2026
Vol. II.
Número. II

Análisis Cuantitativo de la Eficiencia en la Paralelización OpenMP de la Ecuación de Difusión: Un Estudio Experimental de Escalabilidad y Umbrales Críticos

Quantitative Analysis of OpenMP Parallelization Efficiency for the Diffusion Equation: An Experimental Study of Scalability and Critical Thresholds

José Joel Ruelas Leal[1], Salvador Meza Aguilar[2], Francisco Cesar Delgado Nieblas[1]

[1]Facultad de informática, Universidad Autónoma de Sinaloa, Mexico.

https://orcid.org/0009-0002-0940-4530
jj.ruelas20@info.uas.edu.mx

https://orcid.org/0000-0002-2622-3689
smeza@uas.edu.mx

Autor de Correspondencia: Francisco C. Delgado Nieblas
francisco.delgado@info.uas.edu.mx
https://orcid.org/0009-0007-5593-3341

Recibido: abril 2026
Publicado: junio 2026

IJISTA, Vol. II, Núm. I, junio 2026,
ISSN: 3122-4474

Resumen:

La simulación numérica de ecuaciones diferenciales parciales mediante métodos de diferencias finitas presenta demandas computacionales que crecen rápidamente con la resolución espacial. Este trabajo analiza cuantitativamente la eficiencia de la paralelización OpenMP para la ecuación de difusión unidimensional resuelta con el esquema FTCS. Se implementó un diseño experimental factorial completo con tamaños de malla desde Nx=200 hasta Nx=20000 y configuraciones de 1 a 8 hilos. Los resultados demuestran la existencia de un umbral crítico en Nx=20000, donde se alcanza el primer speedup superior a la unidad (1.05× con 4 hilos). Para problemas pequeños (Nx=200), la paralelización resulta contraproducente debido al overhead asociado a sincronización y administración de hilos. La eficiencia paralela disminuye conforme aumenta el número de hilos, aunque mejora progresivamente para problemas con mayor carga computacional. Mediante la Ley de Amdahl se estimó una fracción paralelizable elevada, mientras que el análisis experimental evidenció la influencia significativa del overhead práctico sobre el rendimiento observado. La precisión numérica se conserva entre versiones serial y paralela hasta el orden de 10^{-14}. Este trabajo proporciona directrices prácticas basadas en evidencia experimental para implementaciones paralelas eficientes en computación científica.

Palabras Clave:

OpenMP, ecuación de difusión, paralelización, speedup, eficiencia paralela, métodos numéricos.

Abstract:

Numerical simulation of partial differential equations using finite difference methods presents computational demands that grow rapidly with spatial resolution. This work quantitatively analyzes the efficiency of OpenMP parallelization for the one-dimensional diffusion equation solved using the FTCS scheme. A full factorial experimental design was implemented with mesh sizes ranging from Nx=200 and thread configurations from 1 to 8 threads. Results demonstrate the existence of a critical threshold at Nx=20000, where the first speedup greater than unity is achieved (1.05× using 4 threads). For small problems (Nx=200), parallelization becomes counterproductive due to synchronization and thread-management overhead. Parallel efficiency decreases as the number of threads increases, although performance progressively improves for larger computational workloads. Using Amdahl's Law, a high parallelizable fraction was estimated, while experimental analysis revealed the significant impact of practical overhead on the observed performance. Numerical precision is preserved between serial and parallel implementations up to the order of 10^{-14}. This work provides evidence-based practical guidelines for efficient parallel implementations in scientific computing.

Keywords:

OpenMP, diffusion equation, parallelization, speedup, parallel efficiency, numerical methods.

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

1. Introducción

La ecuación de difusión constituye uno de los modelos fundamentales de la física matemática y aparece en una amplia variedad de aplicaciones científicas y tecnológicas, incluyendo transferencia de calor, dispersión de contaminantes, transporte de masa y modelos financieros [1] [2]. Su resolución numérica mediante métodos de diferencias finitas continúa siendo relevante debido a la necesidad de simular sistemas con alta resolución espacial y temporal. Entre los esquemas explícitos más utilizados se encuentra el método FTCS (Forward Time Centered Space), ampliamente estudiado por su simplicidad de implementación y sus propiedades de convergencia [2].

El incremento continuo en la resolución de las simulaciones numéricas ha provocado un crecimiento considerable en los requerimientos computacionales. En particular, la condición de estabilidad asociada a esquemas explícitos como FTCS obliga a reducir el paso temporal conforme disminuye el espaciamiento espacial, incrementando significativamente el número total de operaciones necesarias para completar una simulación [3], [4]. Como consecuencia, incluso problemas unidimensionales pueden presentar tiempos de ejecución elevados cuando se emplean mallas suficientemente refinadas.

En las últimas décadas, la evolución de las arquitecturas de hardware ha favorecido el uso de procesadores multinúcleo como estrategia principal para incrementar el rendimiento computacional. En este contexto, OpenMP se ha consolidado como uno de los estándares más utilizados para programación paralela en sistemas de memoria compartida debido a su portabilidad y facilidad de integración en códigos científicos existentes [5], [6]. Su modelo basado en directivas permite paralelizar regiones específicas del código sin modificar completamente la estructura original del programa.

A pesar de las ventajas de la programación paralela, la paralelización no garantiza automáticamente mejoras de rendimiento. Factores como el overhead de creación y sincronización de hilos, la contención de memoria compartida y la fracción serial del código pueden limitar considerablemente el speedup alcanzable [7]. En algunos casos, especialmente para problemas pequeños, la versión paralela puede incluso presentar tiempos de ejecución mayores que su contraparte serial.

Aunque existe una amplia literatura sobre métodos numéricos y programación paralela, todavía son limitados los estudios experimentales que cuantifican sistemáticamente el impacto del tamaño del problema sobre la eficiencia de OpenMP en esquemas explícitos simples como FTCS. En particular, resulta relevante identificar umbrales prácticos a partir de los cuales la paralelización comienza a ser efectiva, así como evaluar la relación entre speedup, eficiencia y overhead bajo diferentes configuraciones de ejecución.

El objetivo de este trabajo es analizar cuantitativamente la eficiencia de la paralelización OpenMP aplicada a la ecuación de difusión unidimensional resuelta mediante el esquema FTCS. Para ello, se implementó un diseño experimental considerando diferentes tamaños de malla y configuraciones de hilos, evaluando métricas de desempeño como tiempo de ejecución, speedup, eficiencia paralela y overhead relativo. Adicionalmente, se verificó la conservación de la precisión numérica mediante normas de error y análisis de convergencia. Los resultados obtenidos permiten identificar umbrales prácticos de paralelización efectiva y establecer recomendaciones para implementaciones paralelas en problemas de difusión.

2. Trabajos Relacionados

2.1 Métodos numéricos para la ecuación de difusión

La resolución numérica de ecuaciones de difusión mediante diferencias finitas ha sido ampliamente estudiada debido a su importancia en física, ingeniería y ciencias computacionales. Los trabajos clásicos de Courant, Friedrichs y Lewy establecieron las bases de estabilidad para esquemas numéricos aplicados a ecuaciones diferenciales parciales [3]. Posteriormente, Von Neumann desarrolló herramientas sistemáticas para el análisis de estabilidad de esquemas explícitos e implícitos, consolidando fundamentos que continúan siendo utilizados en la actualidad [4].

Entre los métodos explícitos, el esquema FTCS destaca por su simplicidad matemática y facilidad de implementación computacional. Diversos estudios han evaluado sus propiedades de convergencia y estabilidad en problemas de difusión y advección-difusión [1], [2].

Aunque existen esquemas más avanzados y precisos, FTCS continúa siendo utilizado como referencia en estudios metodológicos y de desempeño computacional debido a su estructura sencilla y altamente paralelizable.

2.2 OpenMP y computación paralela

El desarrollo de arquitecturas multinúcleo ha impulsado el uso de modelos de programación paralela en aplicaciones científicas. OpenMP se ha consolidado como uno de los estándares más utilizados para programación en memoria compartida debido a su portabilidad y facilidad de integración en códigos existentes [5]. Su modelo basado en directivas permite paralelizar regiones específicas del código con modificaciones relativamente pequeñas en programas secuenciales.

Trabajos recientes han demostrado la utilidad de OpenMP en aplicaciones científicas de gran escala, particularmente en simulaciones numéricas y problemas multiphysics [6]. Sin embargo, la eficiencia de paralelización depende fuertemente del tamaño del

problema, la arquitectura de hardware y el overhead asociado a la sincronización y administración de hilos.

2.3 Métricas de desempeño paralelo

La evaluación del desempeño paralelo suele realizarse mediante métricas como speedup, eficiencia y overhead relativo. La Ley de Amdahl establece límites teóricos para el speedup alcanzable en función de la fracción serial del código [7]. Aunque este modelo proporciona una referencia útil, en aplicaciones reales intervienen además factores asociados a jerarquía de memoria, sincronización y comunicación entre hilos.

A pesar de la abundante literatura sobre programación paralela, todavía son limitados los estudios experimentales que analizan sistemáticamente la relación entre tamaño del problema y overhead en esquemas explícitos simples como FTCS. En este contexto, el presente trabajo busca proporcionar evidencia cuantitativa que permita identificar configuraciones prácticas de paralelización efectiva para problemas de difusión unidimensional.

3. Metodología

3.1 Modelo físico y ecuación de difusión

El problema estudiado corresponde a la ecuación de difusión unidimensional, utilizada para modelar procesos de transferencia de calor y transporte difusivo. La ecuación diferencial parcial considerada es: $\partial u/\partial t = D\,\partial^2 u/\partial x^2$. Donde $u(x,t)$ representa la variable física de interés y D es el coeficiente de difusión. El dominio espacial se definió en el intervalo $0 \le x \le L$.

La discretización numérica se realizó mediante el esquema FTCS (Forward Time Centered Space), utilizando diferencias finitas centradas para la derivada espacial y diferencias progresivas para la derivada temporal. La ecuación discreta utilizada fue:

$$u_i{}^{n+1} = u_i{}^n + r(u_{i+1}^n - 2u_i^n + u_{i-1}^n)$$

donde: $r = D\,\Delta t/\Delta x^2$ corresponde al parámetro de estabilidad del método.

Para garantizar estabilidad numérica se seleccionó un valor de $r = 0.45$, cumpliendo la condición de estabilidad del esquema FTCS. Las simulaciones se realizaron utilizando diferentes tamaños de malla espacial con el objetivo de analizar simultáneamente el comportamiento numérico y el desempeño computacional de la implementación paralela.

3.2 Implementación computacional

Se desarrollaron dos versiones del algoritmo numérico: una implementación serial de referencia y una implementación paralela utilizando OpenMP. Ambas versiones fueron programadas en lenguaje C, empleando la misma estructura numérica y las mismas condiciones iniciales y de frontera, con el propósito de garantizar comparaciones consistentes entre los resultados computacionales.

La paralelización se aplicó principalmente sobre el bucle espacial asociado al cálculo de la evolución temporal del esquema FTCS, debido a que cada punto de la malla depende únicamente de sus vecinos inmediatos. Para ello se utilizaron directivas #pragma omp parallel for con planificación estática (schedule(static)), apropiada para cargas de trabajo uniformes.

Adicionalmente, se paralelizaron regiones relacionadas con la inicialización de arreglos y el cálculo de métricas de error numérico. En el caso de las normas de error, se emplearon operaciones de reducción (reduction) para evitar condiciones de carrera y garantizar la consistencia de los resultados obtenidos durante la ejecución paralela.

La implementación fue compilada utilizando GCC con soporte OpenMP y optimizaciones de compilación orientadas a maximizar el desempeño computacional. Asimismo, se mantuvieron constantes las condiciones de compilación y ejecución durante todos los experimentos para asegurar la reproducibilidad de las mediciones.

Con el propósito de favorecer la reproducibilidad de los resultados, el código fuente utilizado durante los experimentos se encuentra disponible en un repositorio público, incluyendo las implementaciones serial y paralela, así como los scripts empleados para el procesamiento y visualización de datos.

3.3 Configuración experimental

Con el objetivo de analizar sistemáticamente el impacto de la paralelización sobre el desempeño computacional, se implementó un diseño experimental factorial considerando diferentes tamaños de malla y configuraciones de hilos OpenMP. La Tabla 1 resume los parámetros principales utilizados durante los experimentos. Los tamaños de malla evaluados fueron:

$Nx = 200,\ 1000,\ 5000,\ 10000,\ 20000$ mientras que las configuraciones paralelas consideradas correspondieron a: $p = 1,\ 2,\ 4,\ 8$ hilos de ejecución.

Tabla 1. Configuraciones experimentales utilizadas durante las simulaciones numéricas.

Parámetro	Valores
Nx	200 – 20000
Hilos	1, 2, 4, 8
(D)	0.1
(L)	1

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

Todas las simulaciones se realizaron manteniendo constantes los parámetros físicos del problema. El dominio espacial se definió con longitud $L = 1.0$, coeficiente de difusión $D = 0.1$ y tiempo final de simulación $T_f = 0.2$. El parámetro de estabilidad se fijó en $r = 0.45$ para todas las configuraciones experimentales.

Los experimentos se ejecutaron sobre un procesador Intel Core i7-10700K con arquitectura multinúcleo y memoria compartida. Para reducir variaciones asociadas al sistema operativo y al escalamiento dinámico de frecuencia, se mantuvieron constantes las condiciones de ejecución durante todas las pruebas experimentales.

Con el fin de minimizar fluctuaciones estadísticas, cada configuración experimental fue ejecutada múltiples veces y se utilizaron los valores promedio para el análisis de desempeño. Esta estrategia permitió obtener mediciones más estables y comparables entre las diferentes configuraciones evaluadas.

3.4 Instrumentación y métricas de desempeño

Con el propósito de garantizar mediciones reproducibles y comparables, se implementó un esquema de instrumentación orientado tanto al análisis del rendimiento computacional como a la evaluación de la precisión numérica de las simulaciones. Las métricas seleccionadas permiten cuantificar el impacto de la paralelización OpenMP sobre el tiempo de ejecución, el overhead asociado y la conservación de la exactitud numérica respecto a la solución analítica.

Para caracterizar el desempeño computacional se utilizaron métricas de tiempo de pared y tiempo de CPU. El tiempo de pared fue medido mediante la función omp_get_wtime(), la cual proporciona el tiempo real transcurrido durante la ejecución del programa, incluyendo costos de sincronización y administración de hilos. Por otro lado, el tiempo de CPU se obtuvo mediante la función clock(), utilizada para estimar el tiempo efectivo de procesamiento consumido por el sistema.

La evaluación de precisión numérica se realizó utilizando las normas de error L_2 y L_∞, ampliamente empleadas en análisis numérico para comparar soluciones aproximadas con soluciones analíticas. El error global promedio se calculó mediante:

$$errL_2 = \sqrt{\sum_i (u_i - u_{exact}(x_i))^2 \Delta x}$$

Donde u_i representa la solución numérica en el nodo espacial i, mientras que $u_{exact}(x_i)$ corresponde a la solución analítica evaluada en la misma posición espacial.

El error máximo absoluto en toda la malla espacial se determinó mediante la norma:

$$errL_\infty = \max_i |u_i - u_{exact}(x_i)|$$

Estas métricas permiten caracterizar simultáneamente el desempeño computacional y la estabilidad numérica del algoritmo, proporcionando una base objetiva para comparar las implementaciones serial y paralela bajo diferentes configuraciones de tamaño de malla y número de hilos.

4. Resultados

4.1 Tiempo de ejecución y escalabilidad

La Figura 1 muestra la evolución del tiempo de ejecución en función del número de hilos para diferentes tamaños de malla. Debido a la amplia diferencia entre escalas temporales, se utilizó un eje vertical logarítmico con el propósito de visualizar simultáneamente configuraciones cuyos tiempos de ejecución difieren en varios órdenes de magnitud.

Para tamaños de malla pequeños, el incremento en el número de hilos provoca un aumento progresivo en el tiempo de ejecución. Este comportamiento indica que el overhead asociado a la creación, sincronización y administración de hilos domina completamente el costo computacional del algoritmo, haciendo que la versión paralela resulte menos eficiente que la implementación serial.

A medida que el tamaño de malla aumenta, el impacto relativo del overhead disminuye gradualmente. En particular, para $Nx = 20{,}000$ se observa una reducción parcial del tiempo de ejecución al utilizar 4 hilos, lo que evidencia el inicio de un régimen donde la paralelización comienza a ofrecer beneficios prácticos de rendimiento.

La Tabla 2 resume las configuraciones más representativas obtenidas durante los experimentos. Los resultados muestran que el máximo speedup alcanzado fue de 1.05× para $N_x = 20{,}000$ con 4 hilos, mientras que para problemas pequeños la paralelización produjo degradación significativa del desempeño computacional.

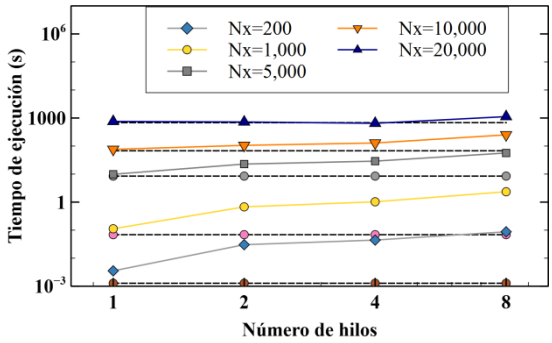

Fig. 1. Tiempo de ejecución en función del número de hilos para diferentes tamaños de malla. En los dos ejes se utiliza escala logarítmica.

Tabla 2. Resultados representativos de desempeño para diferentes tamaños de malla y configuraciones de ejecución.

N_x	Mejor configuración	Tiempo mínimo (s)	Speedup máximo	Eficiencia máxima
200	Serial	0.001275	0.36	36.0%
1,000	Serial	0.068978	0.62	62.0%
5,000	Serial	8.476183	0.86	86.1%
1,0000	Serial	67.801847	0.89	89.0%
2,0000	4 hilos	647.830333	1.05	26.3%

4.2 Speedup y eficiencia paralela

La Figura 2 presenta el speedup obtenido para las diferentes configuraciones de tamaño de malla y número de hilos. El comportamiento observado muestra una fuerte dependencia entre el desempeño paralelo y el tamaño del problema, evidenciando que la paralelización no produce mejoras uniformes en todas las configuraciones experimentales.

Para tamaños de malla pequeños, el speedup permanece por debajo de la unidad incluso al incrementar el número de hilos, indicando que la versión paralela resulta más lenta que la implementación serial. Este comportamiento se encuentra asociado al overhead introducido por OpenMP, el cual domina el costo computacional cuando la carga de trabajo por hilo es reducida.

A medida que el tamaño de la malla aumenta, el speedup mejora progresivamente. En particular, para $N_x = 20{,}000$ se alcanzó un speedup máximo de aproximadamente 1.05× utilizando 4 hilos, constituyendo la primera configuración donde la paralelización produjo una reducción efectiva del tiempo de ejecución.

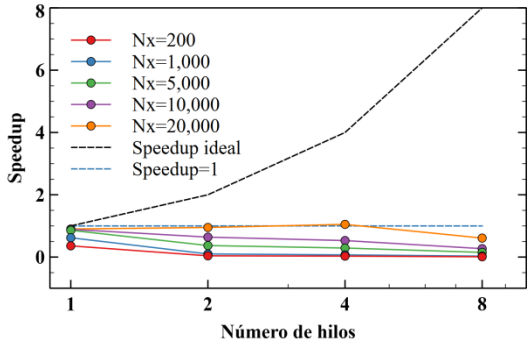

Fig. 2. Speedup en función del número de hilos para diferentes tamaños de malla. La línea discontinua representa el comportamiento ideal S=p.

La Figura 3 muestra la eficiencia paralela correspondiente a las mismas configuraciones experimentales. Se observa una disminución sistemática de la eficiencia conforme aumenta el número de hilos, comportamiento típico en aplicaciones paralelas donde los costos de sincronización y acceso compartido a memoria crecen con el paralelismo empleado.

Aunque la eficiencia disminuye para configuraciones con mayor número de hilos, los resultados muestran que el impacto relativo del overhead se reduce conforme el tamaño del problema incrementa. Esto indica que problemas con mayor carga computacional poseen un mejor potencial de aprovechamiento de arquitecturas multinúcleo mediante OpenMP.

4.3 Evolución del overhead

La Figura 4 muestra la evolución del overhead relativo en función del tamaño de malla para la configuración de 8 hilos. Se observa una disminución sistemática del overhead conforme aumenta el tamaño del problema, indicando que el costo relativo asociado a la paralelización se amortiza progresivamente para cargas computacionales más grandes.

Para tamaños de malla pequeños, el overhead domina completamente el tiempo de ejecución total, provocando que la implementación paralela resulte considerablemente más lenta que la versión serial. Este comportamiento se encuentra relacionado con los costos de creación y sincronización de hilos, así como con la administración de memoria compartida realizada por OpenMP.

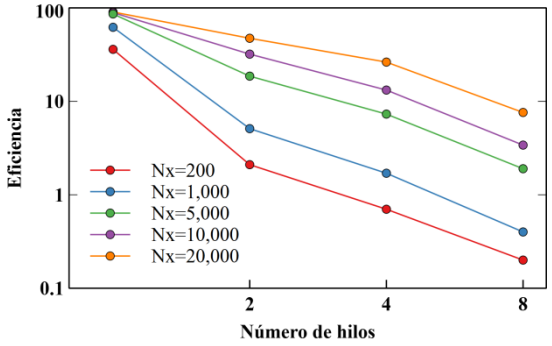

Fig. 3. Eficiencia paralela en función del número de hilos para diferentes tamaños de malla.

A medida que el número de puntos espaciales aumenta, el tiempo dedicado al cálculo numérico comienza a superar el costo administrativo de paralelización. Como consecuencia, el overhead relativo disminuye gradualmente, permitiendo que el algoritmo aproveche de manera más eficiente los recursos multinúcleo disponibles.

Aunque el overhead disminuye para problemas grandes, los resultados muestran que su impacto continúa siendo significativo incluso para las configuraciones de mayor tamaño evaluadas. Esto evidencia que, además de la fracción paralelizable del código, el rendimiento práctico depende fuertemente de factores asociados a

IJISTA, Vol. II, Núm. I, junio 2026, ISSN 3122-4474

sincronización, acceso a memoria y distribución de carga entre hilos.

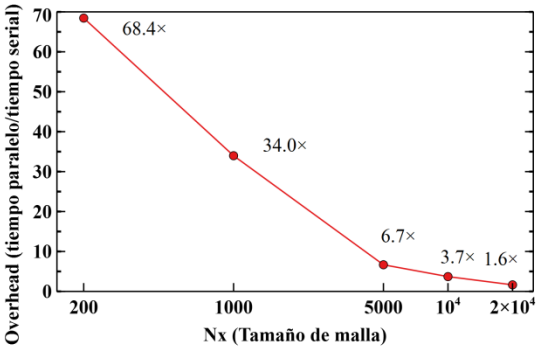

Fig. 4. Evolución del overhead relativo para la configuración de 8 hilos en función del tamaño de malla.

4.4 Precisión numérica y convergencia

Además del análisis de desempeño computacional, se evaluó la precisión numérica de la implementación mediante el estudio de convergencia del esquema FTCS. La Figura 5 muestra la evolución del error numérico conforme se refina la malla espacial, utilizando la norma L2 como métrica de comparación respecto a la solución analítica.

Los resultados muestran una disminución sistemática del error al incrementar la resolución espacial, indicando que la solución numérica converge correctamente hacia la solución exacta del problema de difusión. Este comportamiento confirma la estabilidad y consistencia de la implementación utilizada durante los experimentos.

La pendiente observada en la gráfica de convergencia es consistente con un comportamiento aproximado de segundo orden espacial, en concordancia con las propiedades teóricas del esquema FTCS utilizado para discretizar la ecuación de difusión.

Adicionalmente, no se observaron diferencias significativas entre las soluciones obtenidas mediante las implementaciones serial y paralela, presentándose únicamente variaciones del orden de precisión de máquina. Esto indica que la paralelización mediante OpenMP conserva adecuadamente la exactitud numérica del algoritmo original.

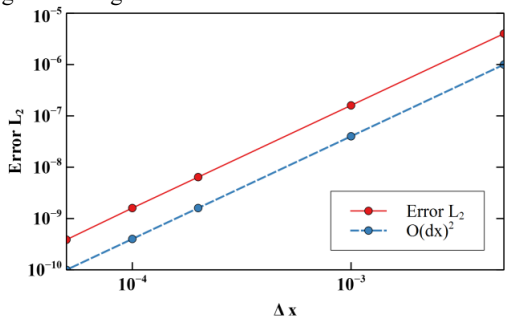

Fig. 5. Convergencia del error numérico en norma L_2. La línea punteada representa la tendencia teórica $O(\Delta x^2)$.

5. Análisis de Resultados

5.1 Interpretación del comportamiento paralelo

Los resultados obtenidos muestran que la eficiencia de paralelización depende fuertemente del tamaño del problema computacional. Para tamaños de malla pequeños, el costo asociado a la creación y sincronización de hilos domina el tiempo total de ejecución, provocando que la implementación paralela resulte menos eficiente que la versión serial.

El comportamiento observado es consistente con el hecho de que el esquema FTCS utilizado posee operaciones computacionales relativamente simples por iteración. En consecuencia, cuando el número de puntos espaciales es reducido, la cantidad de trabajo asignada a cada hilo no resulta suficiente para compensar el overhead introducido por OpenMP.

A medida que el tamaño de la malla aumenta, el costo computacional asociado al cálculo numérico comienza a predominar sobre el overhead de paralelización. Esto explica la mejora gradual observada en el speedup y la reducción relativa del overhead para configuraciones con mayor número de puntos espaciales.

En particular, la configuración correspondiente a $N_x = 20{,}000$ y 4 hilos representó el primer caso donde la paralelización produjo una mejora efectiva de rendimiento respecto a la versión serial. Este comportamiento sugiere la existencia de un umbral práctico de tamaño de problema a partir del cual OpenMP comienza a utilizar de manera más eficiente los recursos multinúcleo disponibles.

5.2 Relación con la Ley de Amdahl

El comportamiento experimental observado puede interpretarse mediante la Ley de Amdahl, la cual establece que el speedup máximo alcanzable se encuentra limitado por la fracción serial del código. Aunque el algoritmo FTCS posee una estructura altamente paralelizable debido a la independencia espacial de sus operaciones, el desempeño práctico continúa condicionado por costos asociados a sincronización y administración de hilos.

A partir de los resultados experimentales correspondientes a las configuraciones de mayor tamaño de malla, se estimó que la fracción paralelizable del código es considerablemente alta. Sin embargo, el speedup obtenido experimentalmente permanece muy por debajo del comportamiento ideal, evidenciando que el overhead práctico introduce limitaciones importantes que no son capturadas completamente por el modelo teórico idealizado.

IJISTA, Vol. II, Núm. I, junio 2026,
ISSN 3122-4474

La diferencia observada entre el speedup ideal y el speedup experimental confirma que el rendimiento paralelo no depende únicamente de la proporción de código paralelizable. Factores asociados a jerarquía de memoria, acceso concurrente a recursos compartidos y sincronización entre hilos desempeñan un papel fundamental en el desempeño final de aplicaciones científicas paralelas.

Los resultados obtenidos muestran que incluso algoritmos con una estructura computacional favorable para paralelización pueden presentar ganancias limitadas cuando el tamaño del problema no es suficientemente grande. Esto resalta la importancia de evaluar experimentalmente el comportamiento de aplicaciones paralelas bajo condiciones reales de ejecución.

5.3 Implicaciones prácticas

Los resultados obtenidos permiten establecer recomendaciones prácticas para el uso de OpenMP en problemas de difusión resueltos mediante FTCS. Para tamaños de malla pequeños, la implementación serial resulta más conveniente debido a que el overhead de paralelización supera ampliamente cualquier posible ganancia de rendimiento.

En configuraciones con mayor carga computacional, la paralelización comienza a mostrar beneficios parciales, particularmente cuando se utilizan cantidades moderadas de hilos. En este contexto, configuraciones intermedias como 4 hilos mostraron el mejor equilibrio entre costo de sincronización y distribución de carga computacional.

Estos resultados sugieren que la eficiencia de OpenMP depende no solamente del número de hilos disponibles, sino también de la relación entre el tamaño del problema y el costo administrativo asociado a la paralelización. En consecuencia, la selección de configuraciones paralelas adecuadas debe realizarse considerando simultáneamente el hardware disponible y la carga computacional del problema estudiado.

5.4 Limitaciones del estudio

El presente trabajo se enfocó en una implementación OpenMP ejecutada sobre una única arquitectura de hardware basada en memoria compartida, por lo que los resultados obtenidos podrían variar en sistemas con diferentes configuraciones de procesador, jerarquía de caché o ancho de banda de memoria.

Asimismo, el análisis se limitó a la ecuación de difusión unidimensional utilizando el esquema explícito FTCS, el cual posee una estructura computacional relativamente simple y altamente paralelizable. En problemas multidimensionales, esquemas implícitos o arquitecturas heterogéneas, el comportamiento del overhead y la eficiencia paralela podrían diferir significativamente.

A pesar de estas limitaciones, los resultados presentados proporcionan evidencia cuantitativa útil para comprender el impacto práctico de la paralelización OpenMP en problemas numéricos de difusión y constituyen una base experimental para estudios futuros en arquitecturas y esquemas numéricos más complejos.

6. Conclusiones

En este trabajo se realizó un análisis cuantitativo de la eficiencia de paralelización OpenMP aplicada a la ecuación de difusión unidimensional resuelta mediante el esquema FTCS. Los resultados obtenidos mostraron que el desempeño paralelo depende fuertemente del tamaño del problema computacional y del número de hilos utilizados durante la ejecución.

Para tamaños de malla pequeños, la paralelización resultó contraproducente debido al impacto dominante del overhead asociado a sincronización y administración de hilos. En contraste, para configuraciones con mayor carga computacional se observó una reducción parcial del tiempo de ejecución, alcanzándose un speedup máximo aproximado de 1.05× para $N_x = 20{,}000$ utilizando 4 hilos.

El análisis de convergencia confirmó que la implementación conserva adecuadamente la precisión numérica del esquema FTCS, presentando comportamiento consistente con un orden espacial aproximado de segundo orden. Asimismo, no se observaron diferencias significativas entre las soluciones obtenidas mediante las implementaciones serial y paralela.

Los resultados obtenidos evidencian que la eficiencia práctica de OpenMP no depende únicamente de la fracción paralelizable del código, sino también del balance entre carga computacional y overhead de paralelización. En este sentido, el trabajo proporciona evidencia experimental útil para la selección de configuraciones paralelas apropiadas en aplicaciones numéricas basadas en diferencias finitas.

Como trabajo futuro, se propone extender el análisis a problemas bidimensionales y tridimensionales, así como evaluar esquemas implícitos y arquitecturas heterogéneas que permitan estudiar con mayor profundidad el comportamiento del overhead y la escalabilidad en aplicaciones científicas paralelas.

Los autores agradecen el apoyo otorgado a esta investigación por parte del Centro Nacional de Supercómputo del IPICYT, A.C., mediante el tiempo de cómputo otorgado al proyecto TKII-E-1025-I-081025-75 y TKII-E-1025-I-081025-76.

7. Referencias

[1] S. Sungnul, K. Para, E. J. Moore, S. Sirisubtawee y S. Phongthanapanich, «A finite difference method for solution of integer-order and Caputo fractional-time advection-diffusion-reaction equations: Convergence analysis and application to air pollution,» *Engineering Letters,* vol. 33, nº 2, p. 292–311, 2025.

[2] C. L. Gardner, «Numerical Methods for Parabolic PDEs,» de *Numerical Methods for Unsteady Compressible Flow Problems*, Springer, 2024.

[3] R. Courant, K. Friedrichs y H. Lewi, «Über die partiellen Differenzengleichungen der mathematischen Physik,» *Mathematische Annalen,* vol. 100, nº 1, p. 32–74, 1928.

[4] J. von Neumann y R. D. Richtmyer, «A method for the numerical calculation of hydrodynamic shocks,» *Journal of Applied Physics,* vol. 21, nº 3, p. 232–237, 1950.

[5] Intel Corporation, «"OpenMP and the Future of Scientific Computing," Intel Developer Blog,» 2025. [En línea]. Available: https://www.intel.com.

[6] M. Thavappiragasam, J. A. Harris, E. Endeve y B. Videau, «Performance porting the ExaStar multi-physics app Thornado on heterogeneous systems: A Fortran-OpenMP code-base evaluation," in Advancing OpenMP for Future Accelerators,» de *20th International Workshop on OpenMP.*, Cham, Switzerland, 2024.

[7] G. M. Amdahl, «Validity of the single processor approach to achieving large scale computing capabilities,» de *AFIPS Spring Joint Computer Conference*, 1967.

[8] J. L. Gustafson, «Reevaluating Amdahl's law,» *Communications of the ACM,* vol. 31, nº 5, p. 532–533, 1988.

[9] Á. Nagy, «New analytical results and comparison of 14 numerical schemes for the diffusion equation with space-dependent diffusion coefficient,» *Mathematics,* 2024.

[10] Z. Xiang y T. G. Robertazzi, «A DLT-aware performance evaluation framework for virtual-core speedup modeling,» *Future Internet,* vol. 17, nº 11, 2025.