Análisis Comparativo y Paralelización de Métodos de Diferencias y Elementos Finitos

Solución de la Ecuación de Poisson

Kevin Cruz

Universidad Distrital

Junio 2025

Contenido

El Problema Físico y los Casos de Estudio

Ecuación de Poisson

Se resuelve $-\nabla^2 u = f$ con condiciones de frontera de Dirichlet.

Caso 1: Poisson

$f(x, y) = (x^2 + y^2)e^{xy}$

Fronteras ($g(x, y)$):

- Izq./Inf.: $g = 1$
- Derecho: $g = e^{2y}$
- Superior: $g = e^x$

Caso 2: Laplace

$f(x, y) = 0$

Fronteras ($g(x, y)$):

- Izquierdo: $g = \ln(y^2 + 1)$
- Derecho: $g = \ln(y^2 + 4)$
- Inferior: $g = 2\ln(x)$
- Superior: $g = \ln(x^2 + 1)$

Implementación de FDM y Ubicación de Directivas OMP

Arquitectura Común

Todos los códigos FDM usan el método de Jacobi en una rejilla 2D. La diferencia reside en cómo se paraleliza el bucle principal de cálculo.

Ubicación de las Directivas OpenMP en FDM

- **'poisson**$_{p}arallel$' : `#pragma omp parallel for`$seaplicaalbuclesobrelasfilasi$.

Implementación de FEM (1/6): Discretización

La Malla de Nodos y Elementos

Nodos

- Se definen con `struct Node {double x, y; bool is_boundary;}`.
- La variable `is_boundary` es una **bandera** que marca si un nodo está en la frontera.

Elementos Triangulares

- No se almacenan en una lista. Se generan **"al vuelo"** dentro de un bucle, dividiendo cada cuadrilátero de la rejilla en dos triángulos.

Implementación de FEM (2/6): La Matriz 'grad'

La "Máquina" para Calcular la Inclinación del Plano

El Objetivo: Encontrar la Inclinación

Asumimos que el potencial dentro del triángulo es un plano: $u(x, y) = ax + by + c$. Nuestro objetivo es encontrar sus inclinaciones 'a' y 'b'.

El Rol de la Matriz 'grad'

La matriz 'grad' es la "máquina" que traduce las 3 alturas de las esquinas (u_0, u_1, u_2) en la inclinación del plano:

$$\underbrace{\begin{pmatrix} \text{Inclinación X} \\ \text{Inclinación Y} \end{pmatrix}}_{\nabla u} = \underbrace{\begin{pmatrix} \dots & \dots & \dots \\ \dots & \dots & \dots \end{pmatrix}}_{\text{Matriz } \textbf{grad} \text{ (2x3)}} \times \underbrace{\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix}}_{\text{Alturas}}$$

Implementación de FEM (3/6): Matriz de Rigidez Local

Traduciendo Geometría a Física

¿Qué es la Matriz de Rigidez Local (K_e)?

Es una matriz 3x3 que representa la rigidez"de un solo triángulo. Cuantifica cómo el potencial en cada nodo influye en los otros dos.

Cálculo en el Código

Una vez tenemos la matriz 'grad', usamos la fórmula que traduce la energía del elemento en una operación matricial:

```
Eigen::Matrix3d Ke = area * (grad.transpose() * grad);
```

Implementación de FEM (4/6): El Rol Clave de Eigen

Eigen se utiliza para todas las operaciones de álgebra lineal:

- **Tipos de Datos:** `VectorXd`, `Matrix3d` y, crucialmente, `SparseMatrix<double>` para la matriz global 'K'.
- **Operaciones de Ensamblaje:**
  - `grad.transpose() * grad`: Multiplicación matricial para obtener 'Ke'.
  - `Eigen::Triplet<double>`: Estructura para acumular valores de 'K' en paralelo.
  - `K.setFromTriplets(...)`: Construcción final de la matriz dispersa.
- **Solución del Sistema:**
  - `Eigen::SimplicialLLT<SpMat>`: El objeto solucionador directo.
  - `solver.solve(F)`: La llamada que resuelve el sistema $KU = F$.

Implementación de FEM (5/6): Ubicación de Directivas OMP

El Patrón de "Almacenamiento Local"

1. `#pragma omp parallel`: Se usa **una sola vez** para iniciar el equipo de hilos.
2. `#pragma omp for`: Se aplica al bucle principal que itera sobre los elementos para repartir el trabajo.
3. `#pragma omp critical`: Se usa **al final** del bucle, para proteger la sección donde cada hilo fusiona su lista local de tripletas con la global.

Implementación de FEM (6/6): Solución Final

Aplicación de Condiciones de Frontera

Para cada nodo frontera i, se fuerza la ecuación $1 \cdot u_i = g_i$:

- `K.coeffRef(i, i) = 1.0;`
- `F(i) = boundary_condition(...);`

Solución con Eigen

Se resuelve el sistema con una sola llamada a un **solucionador directo**:

```
solver.compute(K);
u = solver.solve(F);
```

Rendimiento de Referencia: Versión Serial

Comparación de FDM vs. FEM sin paralelismo

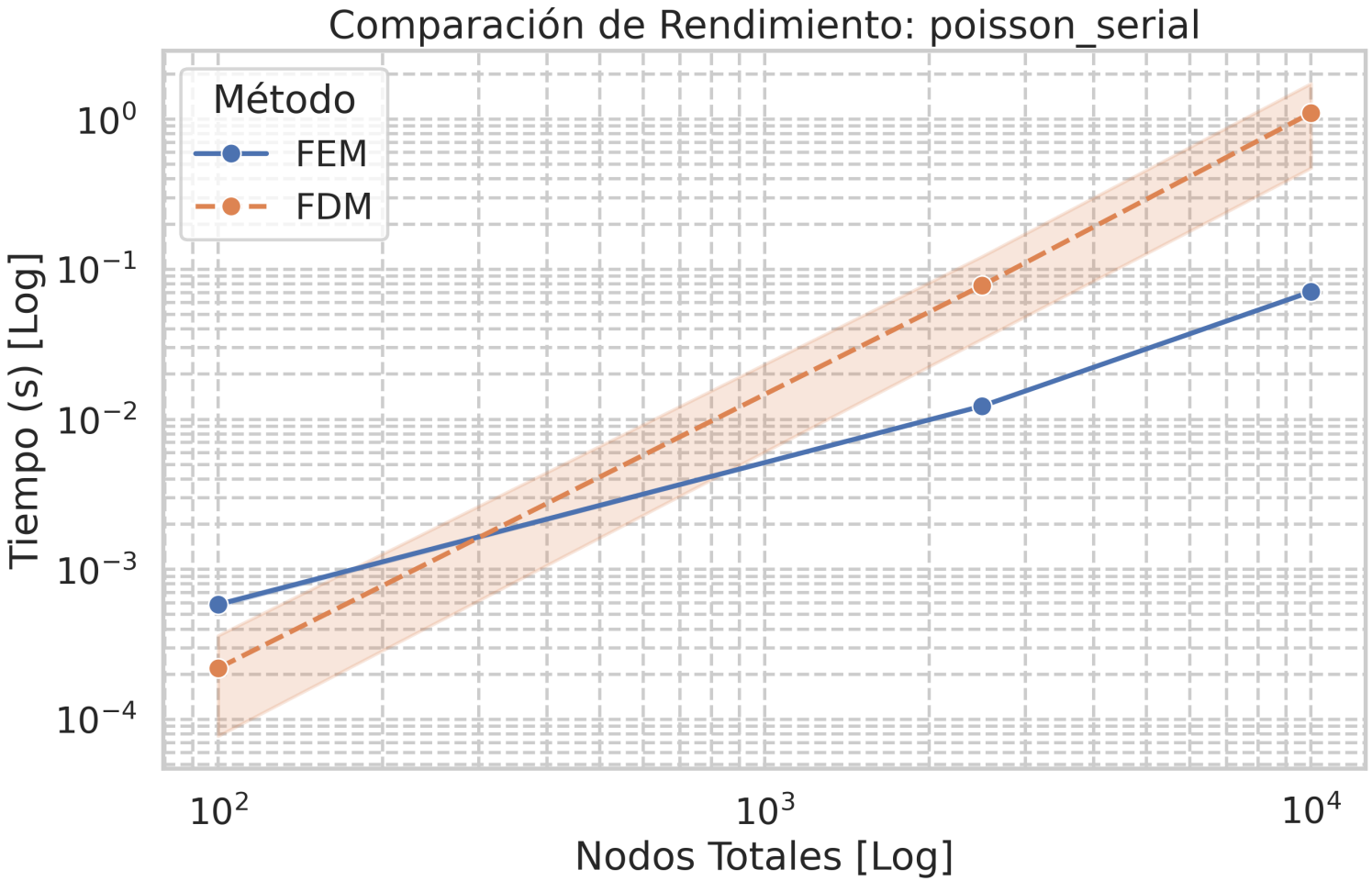

Figura: Tiempo de ejecución en función del tamaño de la malla.

Análisis del Paralelismo de Datos (16 Hilos)

Optimización con ‘collapse(2)‘

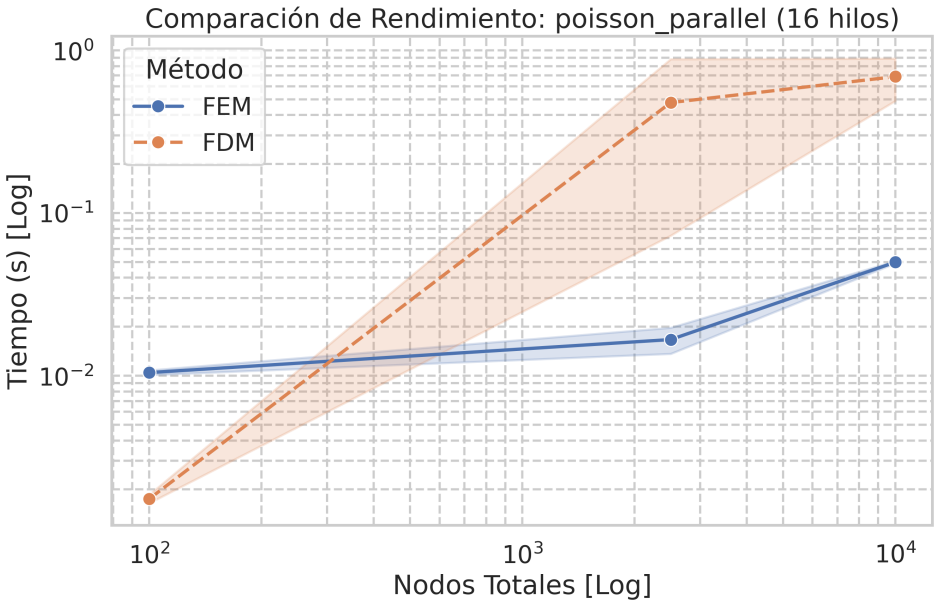

Figura: Bucle ‘for‘ Paralelo Básico

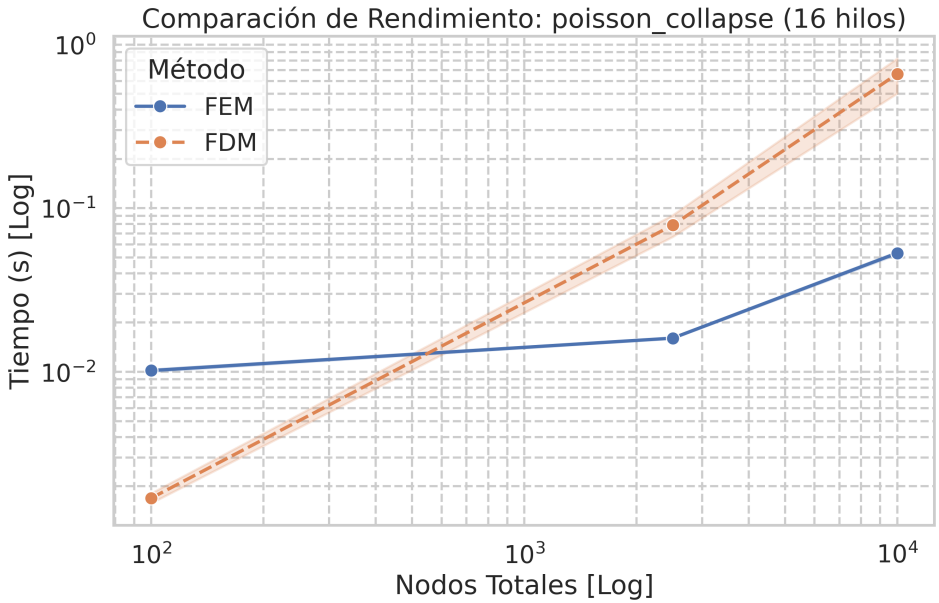

Figura: Con la Cláusula ‘collapse(2)‘

Observaciones Clave

La directiva `collapse(2)` trata los bucles anidados como uno solo, lo que permite a OpenMP un mejor reparto de carga entre los hilos y generalmente resulta en un rendimiento ligeramente superior.

Impacto de la Política de Reparto (16 Hilos)

'schedule(static)' vs. 'schedule(dynamic)'

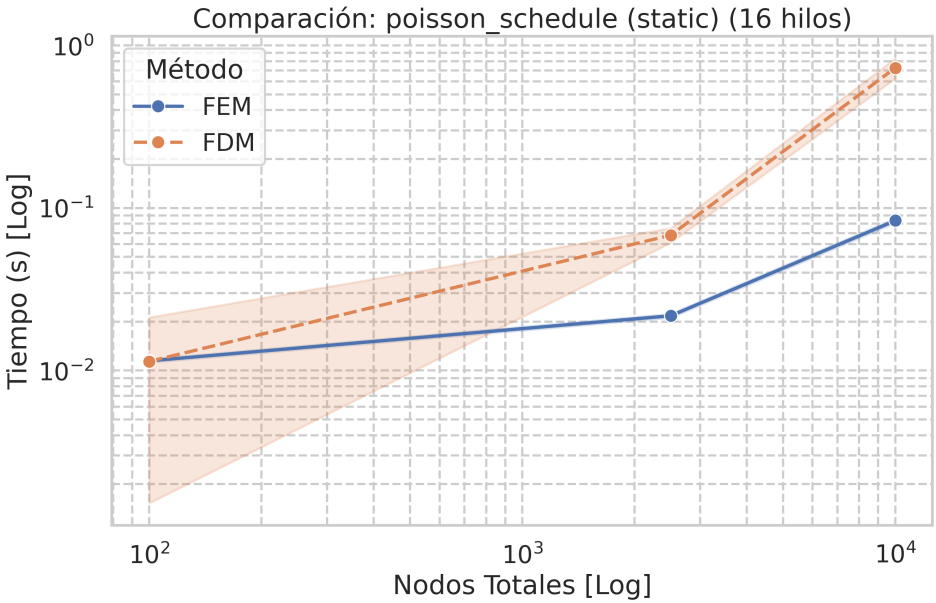

Figura: Reparto Estático

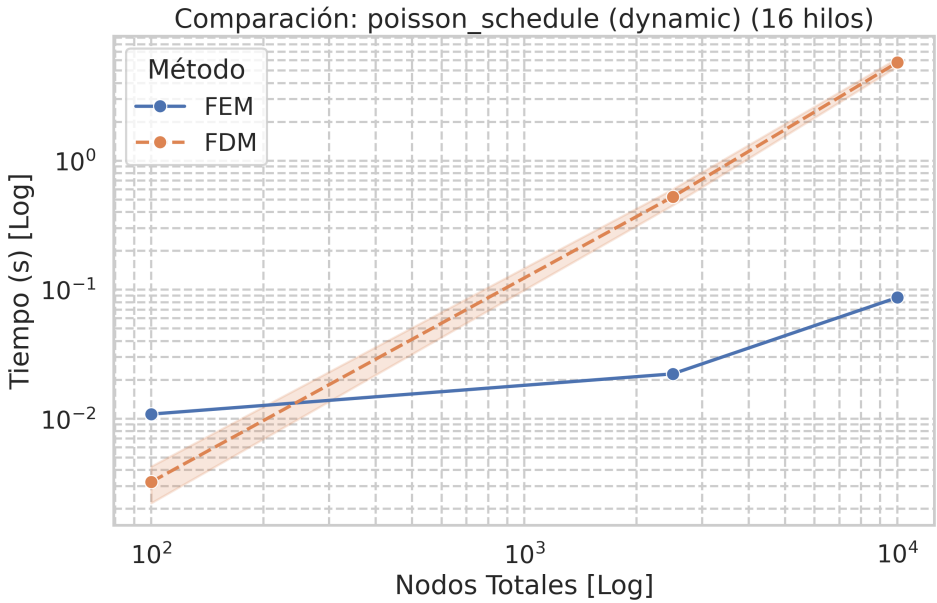

Figura: Reparto Dinámico

Observaciones Clave

- **Static:** Overhead bajo. Ideal para cargas de trabajo uniformes, como en este problema.
- **Dynamic:** Mayor overhead. Útil para bucles donde las iteraciones tienen

Estrategias Alternativas de Paralelización (16 Hilos)

Paralelismo por Tareas y Sincronización

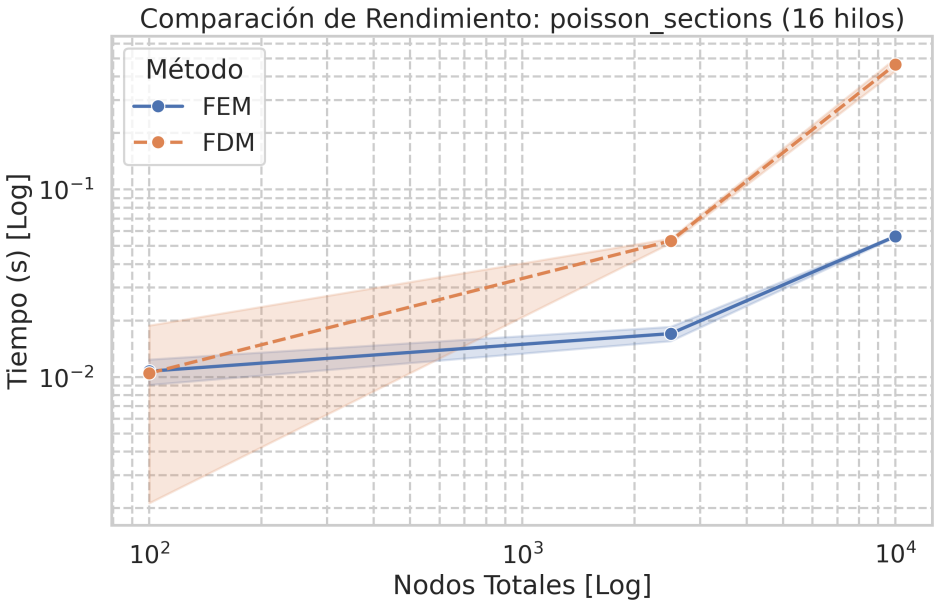

Figura: Paralelismo por Tareas (‘sections‘)

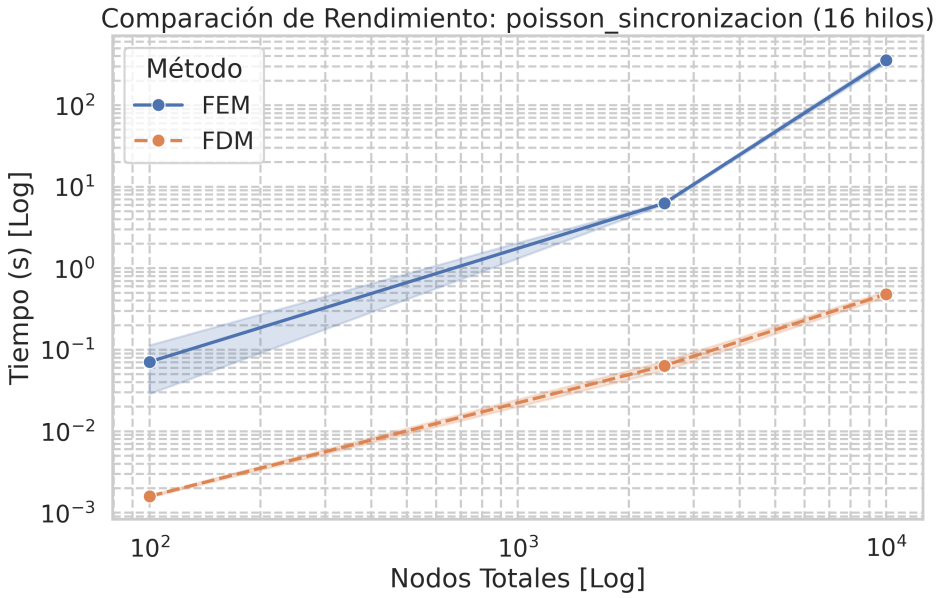

Figura: Versión de Sincronización

Observaciones Clave

Estas gráficas permiten analizar el impacto de estrategias más específicas:

- **Sections:** Mide el beneficio de solapar tareas de inicialización.
- **Sincronización:** Evalúa el rendimiento de una estrategia particular de

```
--- Rendimiento Promedio General (Paralelo) ---
Calculado con la malla más grande (100x100) y 16 hilos.
  metodo    tiempo
0    FDM  1.662926
1    FEM  0.065976
```

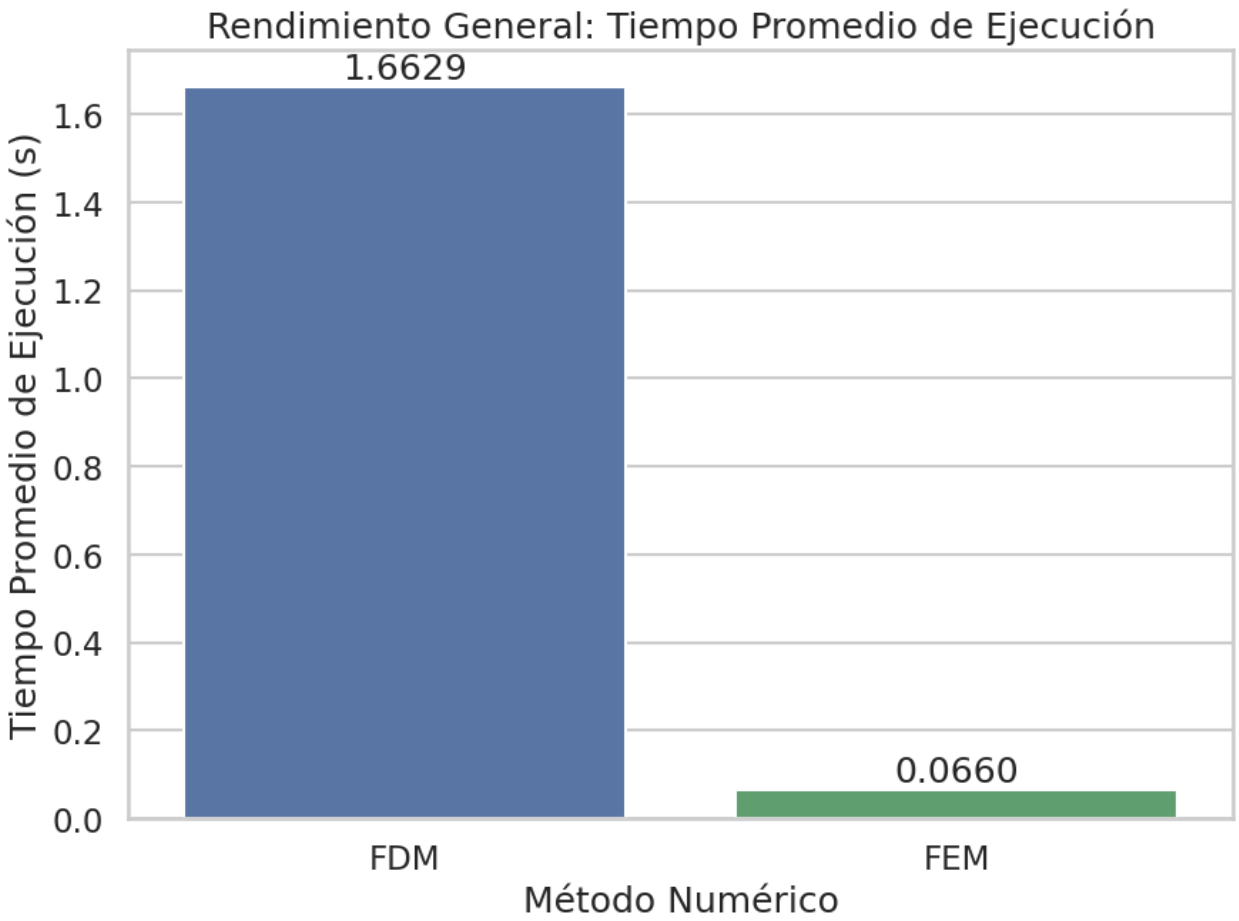

Figura: Enter Caption

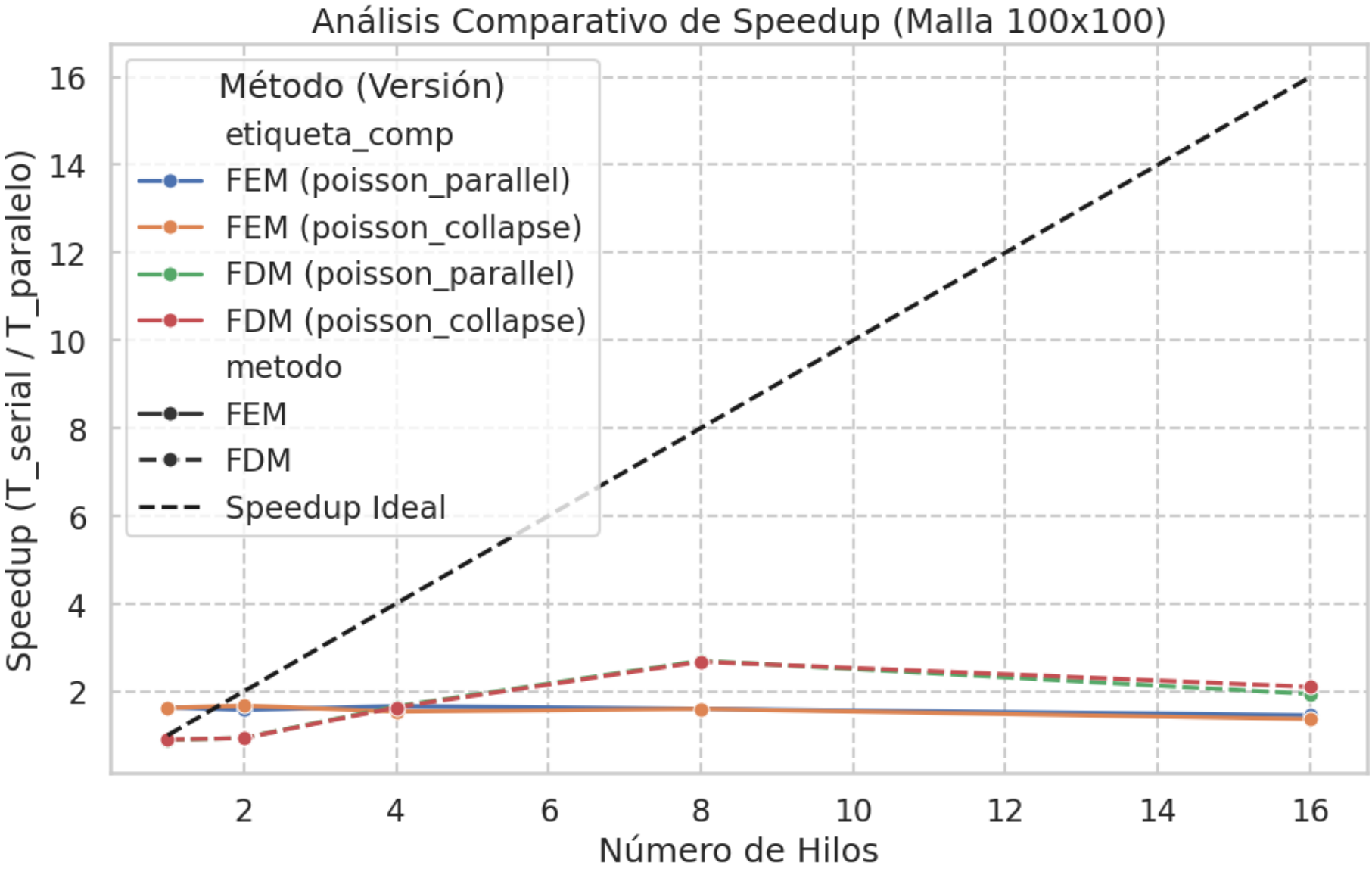

Figura: Enter Caption

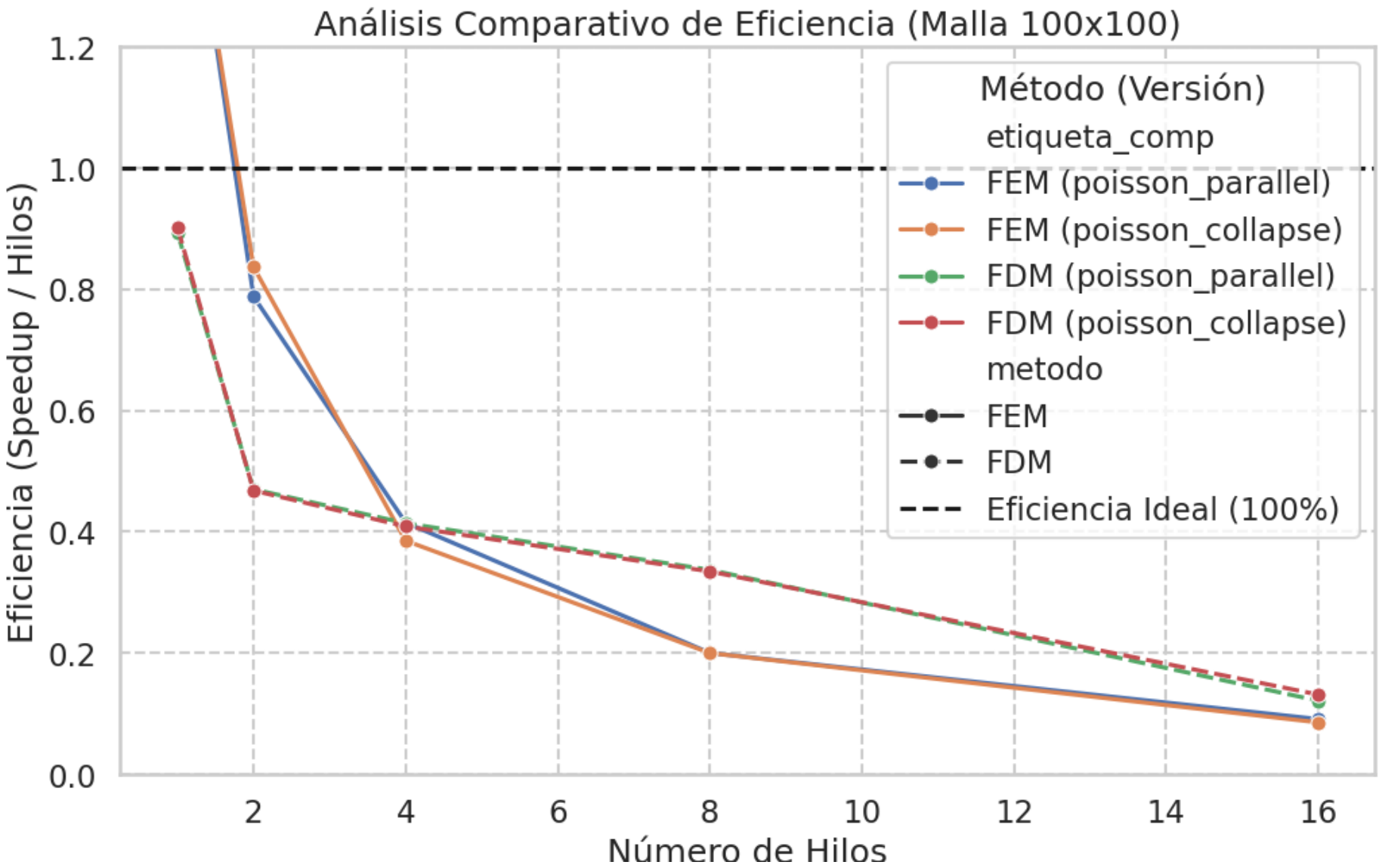

Figura: Enter Caption

Conclusiones

- **FEM** es el método **más eficiente** para problemas grandes, gracias a la potencia de los solucionadores de álgebra lineal directa.
- **FDM** es el método **más escalable**, ya que el método de Jacobi tiene mínima comunicación entre hilos.