DSC014 - Tarea N° 2
Plazo de entrega: 14/11/2025 - 22:00

En esta segunda tarea grupal, el objetivo es aplicar los conceptos que hemos discutido en clase para investigar y resolver un problema concreto. Deben explorar el problema propuesto en profundidad de manera independiente, incluyendo investigación propia sobre los conceptos involucrados para la resolución del problema y la interpretación de la(s) solución(es). La lista de tareas a realizar es orientativa y consiste en lo mínimo esperado de la tarea.

El informe solicitado debe presentarse en un Jupyter Notebook, entregado en dos formatos: `.ipynb` (para su edición y revisión de código) y `.html` (para una visualización estática y portable). La estructura del notebook debe seguir el formato de un informe técnico o académico, incluyendo secciones bien definidas como introducción, marco teórico, metodología, análisis de resultados, discusión y conclusiones. En caso de utilizar programas `.py` separados como los vistos en clases, deben ser entregados junto el notebook para su posterior revisión.

Cada sección debe contener una descripción exhaustiva de los fenómenos estudiados, justificando los métodos y técnicas aplicadas. Se espera una exposición detallada de los pasos seguidos, junto con una discusión crítica de los hallazgos. Las visualizaciones dinámicas deben incluirse en la entrega en formato `.mp4` y ser etiquetadas con nombres que sean claros, concisos y descriptivos para facilitar su identificación.

Objetivo general de la tarea

En esta segunda tarea grupal, el objetivo es aplicar los conceptos de cómputo paralelo para resolver un problema de evolución temporal descrito por una Ecuación en Derivadas Parciales parabólica. Deberán implementar, paralelizar y analizar el rendimiento de una simulación de transferencia de calor, un problema fundamental en la ingeniería de hardware. Además, deberán comparar su implementación explícita con un método implícito, y analizar críticamente los desafíos que este último presenta para la paralelización.

Análisis Térmico de un Procesador (CPU/GPU) bajo Carga Variable

El rendimiento de los procesadores modernos está fundamentalmente limitado por la disipación de calor. Un procesador (CPU o GPU) genera calor (efecto Joule) cuando realiza cálculos. Esta generación de calor no es constante; depende directamente de la carga computacional, la cual varía milisegundo a milisegundo.

Su tarea es modelar la disipación de calor en la base de un disipador (un "heat spreader") que está en contacto con el *die* de silicio del procesador. La temperatura $T(x, y, t)$ en esta placa 2D se rige por la Ecuación del Calor vista en clases, con un término fuente variable en el tiempo:

$$\frac{\partial T}{\partial t} = \alpha \left(\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} \right) + S(x, y, t) \,, \tag{1}$$

donde

- $T(x, y, t)$ es la temperatura en la posición (x, y) en el tiempo t, relativa a la temperatura ambiente.
- α es la difusividad térmica del material (ej. cobre).
- $S(x, y, t)$ es el término fuente: el calor generado por el procesador.

Condiciones del Problema

Considere el problema físico dentro del siguiente rango de parámetros:

- **Dominio:** Una placa cuadrada de $L \times L$. Asuma $L = 5\,\text{cm}$.
- **Difusividad Térmica:** Asuma que la placa es de cobre, con $\alpha = 1.1 \times 10^{-4}$ m^2/s.
- **Condición Inicial (IC):** La placa comienza a temperatura ambiente. Asumiremos $T_{\text{ambiente}} = 0$, por lo que $T(x, y, 0) = 0$.
- **Condiciones de Borde (BC):** Los cuatro bordes de la placa ($x = 0, x = L, y = 0, y = L$) están en contacto con un disipador ideal, que los mantiene a temperatura ambiente (condiciones de Dirichlet).
- **Término Fuente $S(x, y, t)$:** El calor se genera en un hotspot cuadrado en el centro (el *die* del procesador). Se modela como

$$S(x, y, t) = P(t) \times \begin{cases} 145.0 & 0.4L \leq x \leq 0.6L \text{ y } 0.4L \leq y \leq 0.6L \\ 0.0 & \text{en otro caso} \end{cases}$$

  Note que con $L = 5$cm, la región $0.4L \to 0.6L$ define un *die* de 1cm $\times$ 1cm en el centro. El valor de 145.0 K/s es una estimación física basada en una potencia de 150W disipada a través de un *die* de 1cm^2 en una placa de cobre de 3mm de espesor.

- **Carga Variable $P(t)$:** Simule una carga de trabajo pulsante con una frecuencia f:

$$P(t) = \frac{1}{2} \left(1 + \text{sign}(\sin(2\pi f t))\right)$$

  Explore al menos dos frecuencias: una lenta (ej. $f = 0.5$ Hz, simulando ráfagas de carga) y una rápida (ej. $f = 10$ Hz).

Tareas mínimas a realizar

1 - Formulación Numérica (Método Explícito)

Discretice el dominio en una grilla uniforme de $N \times N$ celdas ($N \leq 512^2$) y utilice el método explícito visto en Clase 6.

1. Derive la ecuación de actualización explícita para $T_{i,j}^{n+1}$ en función de sus vecinos en el tiempo n.
2. Determine el **criterio de estabilidad** para esta discretización 2D. Justifique su respuesta basándose en la extensión del criterio 1D. Elija un Δt que satisfaga esta condición. Note que el criterio de estabilidad difiere respecto del caso 1D. Realice una búsqueda bibliográfica para el caso 2D del método.

2 - Implementación Serial (Base)

Escriba un programa que resuelva el problema de forma serial. Este será su código de referencia (T_1) para medir el *speedup*.

3 - Implementación Paralela (MPI)

Paralice la simulación serial (3.2) utilizando `mpi4py` y una estrategia de **descomposición de dominio**.

1. Divida la grilla 2D en P franjas (horizontales, verticales o mixtas).
2. Cada proceso debe ser responsable de su propia franja.
3. Implemente el intercambio de información entre **celdas fantasma** necesario en cada paso de tiempo.

4 - Análisis de Rendimiento

Para una moderada (ej. $N = 256 \times 256$ o $N = 512 \times 512$):

1. Mida el tiempo de ejecución serial T_1.
2. Mida el tiempo de ejecución paralelo T_p (MPI) para $p = 2, 4, 8, \ldots$ procesadores (o el máximo que su laptop le permita).
3. Grafique el *Speedup* ($S_p = T_1/T_p$) vs. p.
4. **Discusión:**
    - ¿Por qué su curva de *speedup* se desvía del ideal $S_p = p$?
    - No se limite a nombrar la Ley de Amdahl o el *overhead* de comunicación. Debe estimar (idealmente medir, si es posible) el **ratio entre el tiempo de cómputo (bytes/flops) y el tiempo de comunicación (bytes/segundo)** en su implementación.
    - Explique cómo este ratio (cómputo vs. comunicación) cambia a medida que p aumenta (los dominios se hacen más pequeños) y cómo esto impacta directamente la *eficiencia paralela* $E_p = S_p/p$.

5 - Método Implícito (Crank-Nicholson)

El método anterior está limitado por un Δt muy pequeño. Implemente el método de **Crank-Nicholson** (C-N) para este problema 2D.

1. Escriba la ecuación de discretización de C-N. Notará que esto genera un sistema de ecuaciones lineales de la forma $A\vec{T}^{n+1} = \vec{b}$ en cada paso de tiempo, donde $\vec{T}^{n+1}$ es el vector de todas las temperaturas $T_{i,j}^{n+1}$.

2. Resuelva este sistema de forma **serial** en cada paso de tiempo. (Sugerencia: use `scipy.sparse.linalg.spsolve` para resolver eficientemente el sistema matricial).

3. **Comparación de Métodos:**

   - Demuestre que C-N es incondicionalmente estable. Use un Δt que sea $10\times$ o $100\times$ mayor que el límite de estabilidad del método inicial y muestre que la solución se mantiene estable.
   - Compare el tiempo de ejecución total entre métodos para alcanzar el mismo tiempo de simulación t_{final}. ¿Cuál es el *trade-off* entre el costo computacional por paso y el tamaño del paso de tiempo?
   - ¿Por qué la estrategia de descomposición de dominio con *ghost cells* **no funciona** para paralelizar el método de Crank-Nicholson con MPI? ¿Qué problema fundamental introduce la naturaleza *implícita* del método para el paralelismo de memoria distribuida? (Pista: Piense en la dependencia de los datos).

6 - Análisis Físico y Multimedia

1. Genere una animación que muestre la evolución de la temperatura 2D a lo largo del tiempo.

2. Grafique la temperatura del punto más caliente $T(L/2, L/2, t)$ en función del tiempo, superpuesta con la función de potencia $P(t)$.

   - Describa el retraso térmico (*thermal lag*) que observa entre la aplicación de la potencia y el peak de temperatura.
   - Compare los gráficos para la frecuencia de pulso lenta y la rápida. ¿Puede concluir algo sobre la *inercia térmica* del sistema?
   - ¿Alcanza el sistema un estado estacionario, o un equilibrio oscilatorio? Discuta cómo la frecuencia de la fuente impacta la amplitud de la oscilación de la temperatura.

Resumen de Entregables

Resumen de los entregables (ver párrafo inicial de la tarea).
El informe debe presentarse en un Jupyter Notebook, entregado en formatos `.ipynb` y `.html`. La estructura debe seguir un informe técnico (introducción, metodología, análisis de resultados, discusión, conclusiones).

- **Metodología:** Incluya la derivación de sus esquemas numéricos y los criterios de estabilidad. Explique sus estrategias de paralelización y la estructura que empleará en las comunicaciones.
- **Análisis de Resultados:** Muestre las simulaciones (imágenes y animación). Presente sus gráficos de *speedup* y el análisis físico de la temperatura.
- **Discusión Crítica:** Responda en detalle las preguntas solicitadas y completemente con cualquier otra pregunta que decidan explorar.
- **Código:** Incluya sus implementaciones serial, MPI, y la serial de C-N. El código debe ser claro y estar bien comentado.
- **Multimedia:** Incluir las animaciones en formato `.mp4`.