Temas para proyectos finales

Marlon Brenes*

Los siguientes son los enunciados de los proyectos de investigación para el proyecto final del curso FS-0432. En todos estos proyectos, se espera que cada uno de los grupos implemente las prácticas estudiadas en clase. Las cuales incluyen, como mínimo:

- Uso apropiado de los recursos de memoria
- Paradigmas de programación implementados de forma correcta en la medida de lo posible
- Uso apropiado de los principios de la OOP (object-oriented programming)
- Métodos de optimización (accesos de memoria, etc)
- Paralelismo
- Documentación y buenas prácticas de colaboración (git)

El proyecto se presenta a todos los grupos del curso la semana que comienza el día 7-7-2025. El lugar de la presentación se define días antes de la prueba. Cada grupo dispone de 20 minutos para su presentación.
Los siguientes son los productos a entregar:

1. Repositorio que contiene la implementación
2. Página de dcoumentación detallando el problema, recusos de implementación, ejemplos de uso y metodología
3. Presentación (exposición) usando los recursos que desee

I. ECUACIÓN DE CALOR EN DOS DIMENSIONES

La idea del proyecto corresponde a resolver de manera numérica la ecuación de calor en dos dimensiones.
Suponga que $u = u(x, y, t)$ es una variable escalar que define la temperatura de una región de dos dimensiones en el plano Cartesiano (x, y) como función del tiempo. Bajo condiciones ideales (sin fuentes de energía externas, capacidad calórica uniforme, aislamiento perfecto), la ecuación de movimiento está dada por

$$\frac{\partial u}{\partial t} = c^2 \left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right], \tag{1}$$

para alguna región acotada $x \in [0, a]$ y $y \in [0, b]$. Las condiciones de frontera pueden cambiar y la dinámica está determinada por éstas y por las condiciones iniciales.
Milestones:

- Encontrar una metodología numérica para resolver el sistema de ecuaciones
- Implementar la solución en `Python`
- Implementar la solución en `C++`
- Utilice distintas condiciones iniciales
- En un gráfico de dos dimensiones con mapa de colores, encuentre una forma de visualizar la solución a través del tiempo
- Encontrar una forma de acelerar la aplicación utilizando paralelismo de memoria compartida

* marlon.brenes@ucr.ac.cr

II. MODELO DE ISING CUÁNTICO UNIDIMENSIONAL EN UNA GRILLA DE N ESPINES: DINÁMICA DE MUCHOS CUERPOS

La dinámica de un estado puro para un sistema cuántico aislado se rige bajo la ecuación Schrödinger ($\hbar = 1$)

$$\frac{\partial \left|\psi(t)\right\rangle}{\partial t} = -\mathrm{i}\hat{H}\left|\psi(t)\right\rangle, \tag{2}$$

cuya solución formal está dada por

$$\left|\psi(t)\right\rangle = e^{-\mathrm{i}\hat{H}(t-t_0)}\left|\psi(t=t_0)\right\rangle. \tag{3}$$

Es decir, la solución involucra resolver de manera numérica la ecuación diferencial Eq. (2) o evaluar de alguna forma la exponencial de la matriz Eq. (3). La idea del proyecto es evaluar la dinámica del modelo de Ising Eq. (4) empezando de algún estado inicial. El modelo de Ising en una dimensión corresponde a

$$\hat{H} = -J\sum_{i=1}^{N}\hat{\sigma}_i^z\hat{\sigma}_{i+1}^z - g\sum_{i=1}^{N}\hat{\sigma}_i^x, \tag{4}$$

donde J es una escala energética que determina la interacción ferromagnética, g el parámetro energético del campo transversal y $\hat{\sigma}_i^\alpha$ ($\alpha = x, y, z$) son las matrices de Pauli para el espín i. La implementación del Hamiltoniano involucra construir una matriz Hermítica de dimensión 2^N que describe el sistema.
La construcción de la matriz se realizará con apoyo de su supervisor del proyecto.
Milestones:

- Con base en argumentos numéricos, evaluar cual de las dos metodologías es mejor implementar para la solución numérica
- Construir la matriz Hamiltoniana mediante productos tensoriales
- Resolver el sistema utilizando algún estado inicial y visualizar su dinámica
- Implementar la solución en `Python`
- Implementar la solución en `C++`
- Encontrar una forma de paralelizar el algoritmo y evaluar la aceleración

III. MODELO DE *TIGHT BINDING* EN UNA CADENA MONOATÓMICA CON UN ORBITAL POR ÁTOMO: CÁLCULO DE BANDAS ELECTRÓNICAS

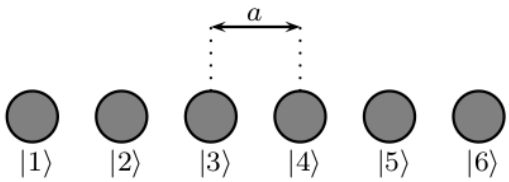

FIG. 1. Cadena monoatómica

Considere el una cadena atómica infinita con un orbital atómico por átomo como se muestra en la Figura 1. Donde a es el parámetro de red, es decir, la distancia entre celdas unidad . La variable t es el parámetro de hopping o de salto, nos dice la probabilidad con que puede saltar un electrón del orbital $|n\rangle$ al orbital $|n+1\rangle$. Con base al modelo *tight binding* Ec. 9, considere la siguiente combinación de orbitales atómicos:

$$|k\rangle = \frac{1}{\sqrt{N}}\sum_{n=1}^{N} e^{inka}\left|n\right\rangle. \tag{5}$$

donde N es el total de sitios y k es un número real con valores entre $-\frac{\pi}{a} \leq k \leq \frac{\pi}{a}$ en el espacio recíproco de tal manera que:

$$\langle n| H |n\rangle = E_0 = E_i - U \tag{6}$$
$$\langle n \pm 1| H |n\rangle = -t \tag{7}$$
$$\langle n\rangle n = 1 \tag{8}$$

Milestones:

- Construir la matriz hamiltoniana dependiente del parámetro k.
- Diagonalizar la matriz hamiltoniana para obtener la energía del sistema en función del parámetro k, es decir en el rango de valores $-\frac{\pi}{a} \leq k \leq \frac{\pi}{a}$ y obtener la estructura de bandas.
- Utilizar la función adecuada para optimizar la diagonalización de la matriz.
- Implementar la solución para una celda unidad de N orbitales atómicos
- Implementar la solución en `Python`
- Encontrar una forma de paralelizar el algoritmo y evaluar la aceleración

IV. MODELO DE *TIGHT BINDING* DE OCUPACIÓN SIMPLE CON POTENCIAL DEFINIDO: DINÁMICA

Coordinador: Marlon Brenes
El modelo de *tight binding* es uno de los modelos más fundamentales en ciencia de materiales, con capacidad de predicción en sistemas de, e.g., grafeno. El Hamiltoniano unidimensional fermiónico para particulas sin espín con condiciones no periódicas está dado por

$$\hat{H} = \sum_{i=1}^{N-1} t_i(\hat{c}_i^\dagger \hat{c}_{i+1} + \hat{c}_{i+1}^\dagger \hat{c}_i) + \sum_{i=1}^{N} \epsilon_i \hat{c}_i^\dagger \hat{c}_i, \tag{9}$$

donde los $\hat{c}_i$ son operadores de destrucción fermiónica y los $\hat{c}_i^\dagger$ operadores de creación para el sitio de la grilla i. Para el caso de un solo fermión en la grilla, este Hamiltoniano es una matriz de dimensión N de carácter tridiagonal. El comportamiento del sistema depende de los parámetros energéticos t_i y ϵ_i. La idea del proyecto es analizar la dinámica del sistema utilizando la misma metodología del enunciado del Proyecto II empezando con un estado inicial en el cual un fermión empieza en el medio de la grilla.
Milestones:

- Con base en argumentos numéricos, evaluar cual de las dos metodologías es mejor implementar para la solución numérica
- Construir la matriz Hamiltoniana
- Resolver el sistema y visualizar su dinámica
- Variar los parámetros energéticos y analizar la solución
- Implementar la solución en `Python`
- Encontrar una forma de paralelizar el algoritmo y evaluar la aceleración

V. DINÁMICA MOLECULAR EN DOS DIMENSIONES: DISCOS SÓLIDOS

Coordinador: Marlon Brenes y Federico Muñoz Utilizando las ecuaciones de cinemática, elabore un programa en dos dimensiones con 4 discos sólidos de radio r que se mueven sin fricción ni momento angular dentro de una caja de longitud unitaria. A estos discos se les puede asociar una energía cinética inicial aleatoria, resultando así en una distribución de velocidades aleatoria para el sistema.
Milestones:

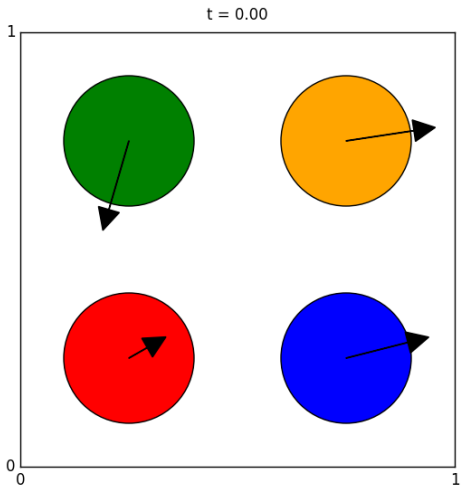

FIG. 2. Dinámica molecular en dos dimensiones

- Implementar la solución en `Python`
- Inicializar el sistema a partir de posiciones conocidas y una distribución de velocidades aleatoria.
- Elaborar subrutinas que evalúan si los discos chocan con la pared o chocan entre ellos (choques a pares) y a la vez calculen las nuevas velocidades (magnitud y dirección) y posiciones del sistema. Considere que no se pierde energía en estos choques.
- Determinar cuál es el evento más próximo a suceder (ya sea colisión con pared o colisión a pares) y en qué momento se da.
- Mover todos los discos la distancia correspondiente a este tiempo y evaluar de nuevo cuándo es el siguiente choque dentro del sistema de nuevo.
- Visualizar por medio de matplotlib la posición de cada uno de los discos dentro de la caja y guardar el correspondiente archivo para realizar una película al final del cálculo.
- Aumentar el número de discos presentes, así como también poder variar el radio r de los discos.
- Elaborar un histograma de las posiciones de los centros de los discos a lo largo del eje-x.

VI. HPC: PARALELIZACIÓN DE LA MULTIPLICACIÓN DE MATRICES EN MEMORIA DISTRIBUIDA Y/O REPLICADA

La multiplicación de matrices es una operación de suma importancia en muchos algoritmos de álgebra lineal. Su aceleración implica acelerar dichos algoritmos. El objetivo de este proyecto es paralelizar la operación

$$\mathbf{C} = \mathbf{A} \cdot \mathbf{B} \tag{10}$$

bajo el paradigma de memoria distribuida. Existe un zoológico de algoritmos para realizar esta operación. Una forma consiste en utilizar memoria replicada, en el cual cada proceso mantiene una copia local de A y de B, mientras que solamente una porción de C. Sin embargo, en su forma mas útil cada proceso mantiene en memoria solamente una porción de cada una de las matrices y el algoritmo se encarga de realizar procesos de comunicación para la operación.
Milestones:

- Utilizar memoria replicada para realizar la operación de multiplicación y evaluar su aceleración
- Implementar un algoritmo basado en distribución homogénea con patrones de comunicación y evaluar su aceleración
- En general, el objetivo de este proyecto es utilizar diversos algoritmos de paralelización de memoria distribuida para acelerar la operación

VII. MODELO SENCILLO DE UN CAPACITOR Y SU SOLUCIÓN NUMÉRICA (ELECTROSTÁTICA)

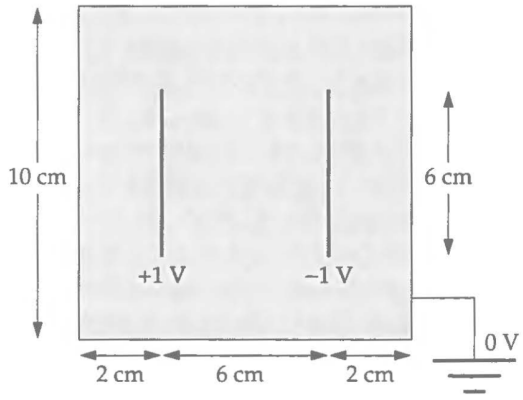

FIG. 3. Modelo de capacitor

Resuelva la ecuación de Laplace en dos dimensiones para el potencial electrostático $\phi = \phi(x, y)$

$$\frac{\partial \phi^2}{\partial x^2} + \frac{\partial \phi^2}{\partial y^2} = 0 \tag{11}$$

para una placa cuadrada de 10cm $\times$ 10cm como se muestra en la Fig. 3. El problema modela de forma ideal un capacitor electrónico.

Milestones:

- En un prototipo usando `Python`, resuelva el problema utilizando el método de relajación de Jacobi, el método de relajación de Jacobi modificado y el método de Gauss-Seidel. Compare la efectividad de cada uno de los métodos comparando el número de iteraciones requeridas para alcanzar cierto valor de convergencia
- Implemente el método de Gauss-Seidel en `C++`
- Paralelice el algoritmo utilizando paralelismo de memoria compartida y evalúe su escalabilidad utilizando grillas lo suficientemente grandes
- Avanzado: paralelice el algoritmo utilizando paralelismo de memoria distribuida, sin replicar la memoria requerida para la grilla