Instituto Tecnológico de Buenos Aires

Simulación de Sistemas

Lattice Gas - FHP Model

Hernan Finucci Roca
Ariadna Fernandez Truglia

Índice

1. Introducción

El objetivo de este trabajo es utilizar una implementación de un autómata celular para analizar el comportamiento de partículas de gas en el vacío y, dado dos cámaras separadas con una rendija en el medio, cuantos estados se tarda en encontrar el equilibrio del sistema. Como se puede ver en la Fig. 1, inicialmente todas las partículas se encuentran en la cámara izquierda.

Figura 1: Representación del comportamiento que se desea simular.

En las próximas secciones se discutirá el modelado, simulación y visualización del problema, así también como las conclusiones encontradas.

2. Modelo

2.1. Modelo matemático

Antes de introducir el modelo implementado, es importante exponer el modelo analítico que lo precede. El modelo implementado es un autómata celular. Los autómatas celulares son arreglos regulares de celdas individuales de la misma clase. Cada una de estas celda tiene un numero finito de estados discretos que se actualizan simultáneamente y sincrónicamente en cada paso temporal. Además, estas reglas de actualización son deterministas y uniformes en tiempo y espacio, y dependen únicamente de una vecindad local de células a su alrededor.

Las ecuaciones de Navier-Stokes, que son ecuaciones derivadas parciales no lineales, describen el movimiento de un fluido viscoso. Para esta situación en particular, resulta útil la ecuación (1), donde v representa la velocidad, P la presión cinemática y ν la viscosidad.

$$\frac{\partial v}{\partial t} + (v\nabla)v = -\nabla P + \nu\nabla^2 v \tag{1}$$

Sin embargo, muy pocas veces se puede encontrar una solución analítica utilizando estas ecuaciones y se deben aplicar varios métodos numéricos. Esto resulta en que resolver estas ecuaciones no sea trivial. Afortunadamente, dichas ecuaciones se pueden reemplazar por el modelo llamada Frish, Hasslacher, and Pomeau (FHP) de 1986, donde se define lo conocido como "lattice gas". Este modelo tiene las siguientes propiedades [1, p. 54]:

1. La retícula regular subyacente muestra una simetría hexagonal.
2. Los nodos están conectados a seis vecinos más cercanos ubicados todos a la misma distancia con respecto al nodo central.
3. Los vectores c_i que conectan los nodos de vecinos más cercanos se llaman vectores de retícula o velocidades de retícula.

$$c_i = (cos\frac{\pi}{3}i, sin\frac{\pi}{3}i),\ i = 1, ..., 6$$

   con $|c_i| = 1$ para todo i
4. Se asocia una celda con cada enlace en todos los nodos.
5. Las celdas pueden estar vacías o ocupadas por como máximo una partícula.
6. Todas las partículas tienen la misma masa y son indistinguibles.
7. La evolución en el tiempo procede mediante una alternancia de dos etapas:
   - Propagación, donde se mueve según velocidades.
   - Colisión, donde se adquieren nuevas velocidades, según las reglas de colisión, que muestran en la Fig. 2.

2.1.1. Reglas de colisión del modelo FHP

Todas las posibles colisiones que se producen en este modelo deben conservar el momento.

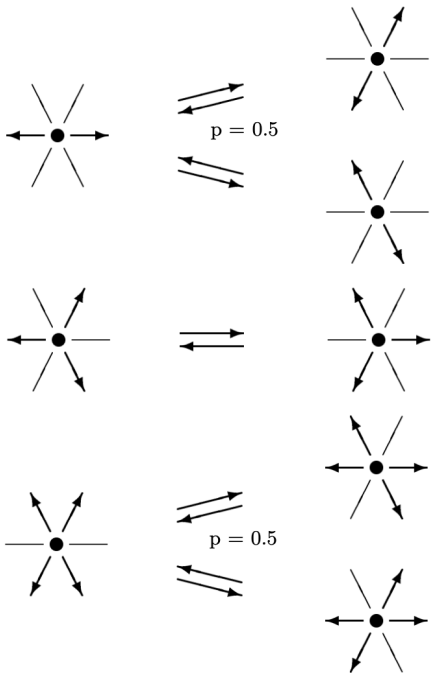

Figura 2: Todas las posibles colisiones de las variantes FHP. Las celdas ocupadas están representadas por flechas, las celdas vacías por líneas delgadas [1, p. 53].

3. Implementación

Antes de explicar se realizó la implementación, es importante aclarar que se considera la definición de celda de una manera diferente como se puede ver en la Fig. 3.

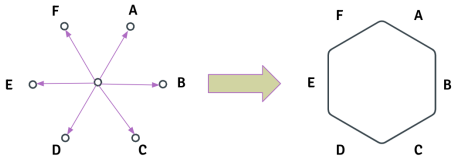

Figura 3: En la derecha se muestra como se define el espacio en el método FHP y en la izquierda como se pensó a la hora de programarlo

Para implementar el modelo FHP, se considero una grilla de 200 x 200 hexágonos que representan el sistema de la Fig. 1.

3.1. Variables del Sistema

Para implementar el modelo se tuvieron en consideración las siguientes variables:

- N: Cantidad de partículas en el sistema.
- D: Tamaño del tabique se separa las dos cámaras.
- ϵ: Variable utilizada para calcular el estado de equilibrio.

Las variables N y D, son variables que se pueden configurar por consola, mientras que ϵ no se puede modificar por consola. Sin embargo, como se verá mas adelante, ϵ fue modificada manualmente en algunas corridas para poder estudiar mejor el sistema. Por ultimo, se decidió que el número de celdas del sistema sea fijo.

3.2. Algoritmo de Resolución

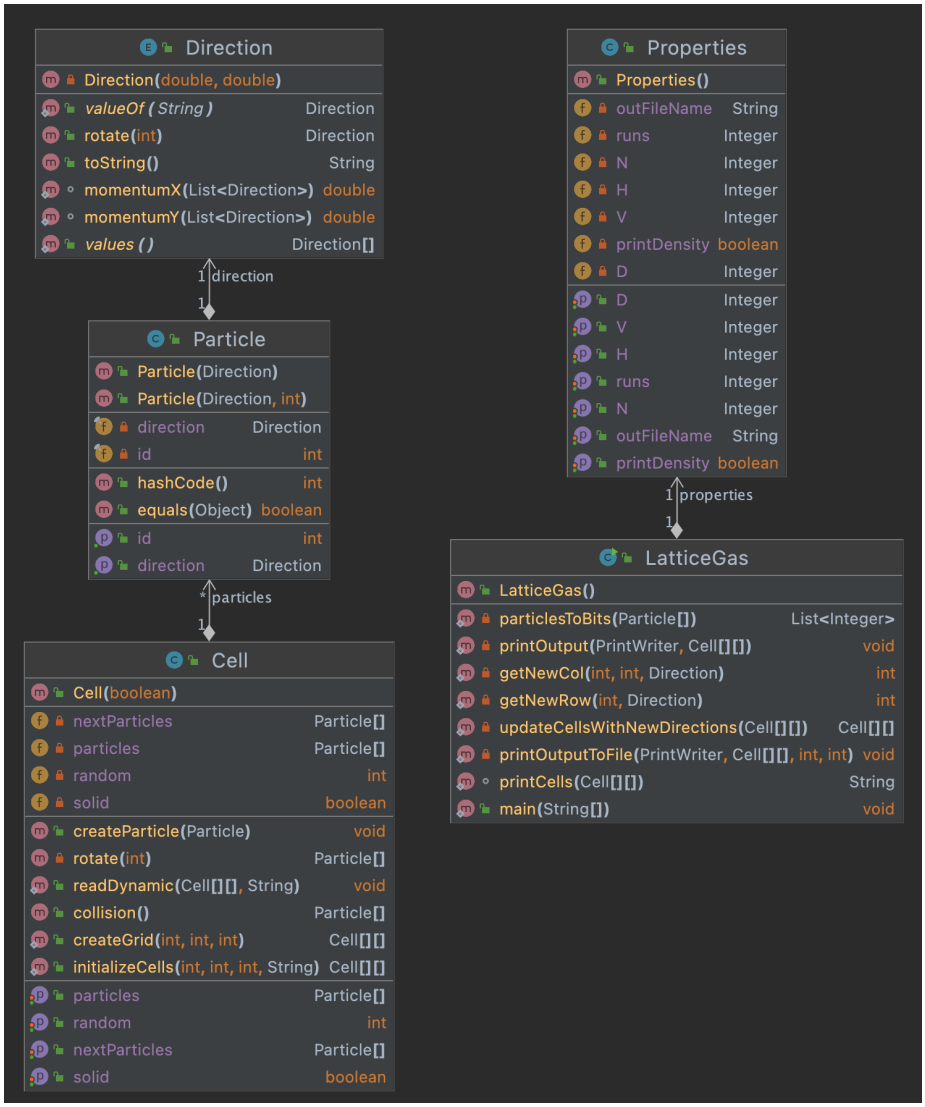

Figura 4: UML de la implementación del modelo FHP.

En la Fig. 4 se puede apreciar por un lado los modelos utilizados y por el otro el algoritmo en sí.

3.2.1. Modelos

Como se puede ver en la Fig. 4, se definieron 3 modelos: *Directions*, *Particles* y *Cells*.

Primero, esta el modelo *Directions* que es un Enum que representa las seis direcciones vistas en la Fig. 3, llamadas con las letras A a la F. Cada elemento tiene dos propiedades, llamadas x e y, los cuales son utilizados en los métodos *momentumX()* y *momentumY()* respectivamente cuando sucede una colisión. A través de estos métodos, se puede calcular en cual de las situaciones de colisión se encuentran las partículas y hacer los pasos necesarios para la próxima propagación.

En el modelo *Particles* se guarda la información que de una partícula, donde se tiene un identificador *id* y una instancia de *Direction*, que indica en qué dirección se está moviendo la partícula.

Por ultimo, el modelo *Cells* computa por completo la celda como se ve definida en la Fig. 3. Dentro del modelo se almacena una lista de *Particle* que hace referencia a las partículas que puede tener la celda. Cada celda sólo puede tener una partícula por dirección. Por esto, la celda como mínimo puede 0 partículas y como máximo 6. El modelo también tiene la propiedad s, que es un boolean que identifica si la celda es sólida o no. La última propiedad de este modelo es r, la cual puede valer 0 o 1. Esta se utiliza para definir cómo va a ser la colisión en los casos que pueden derivar a dos opciones, como se puede ver en la Fig. 2.

3.2.2. LatticeGas

En la clase *LatticeGas* es donde se utilizan los modelos mencionados para computar el modelo FHP. Primero, se crean las 200 x 200 celdas y partículas necesarias para el sistema, marcando que los bordes y la pared del medio, con excepción del tabique, son solidos. Luego, el algoritmo entra en un ciclo que hace la colisión y propagación de las partículas hasta alcanzar el equilibrio. Por un lado, para realizar las colisiones, se recorren las celdas y se deriva el cálculo a el modelo correspondiente. Por el otro lado, la propagación se realiza recorriendo nuevamente las celdas y dependiendo de las nuevas direcciones de las partículas, se las mueve a las celdas correspondientes.

Se consideró que el sistema está en equilibrio cuando sucede lo siguiente:

$$particles_{right} = N/2 * (1 - \epsilon) \tag{2}$$

donde $particles_{right}$ es la cantidad de partículas en la cámara derecha

Por ejemplo, las variables $N = 5000$, $D = 50$ y $\epsilon = 0{,}1$ representan un sistema compuesto por 5000 partículas y con un tamaño de rendija de 50. Este sistema alcanzara el equilibrio cuando el 55 % de partículas se encuentren del lado izquierdo y un 45 % del lado derecho, ya que la diferencia porcentual entre ambos lados es de un 10 %.

3.3. Generador de Partículas

Este programa recibe por consola el parámetro N, el cual indica la cantidad de partículas que se de deben crear en el sistema. Una vez obtenido este número, se comienzan a posicionar partículas en diferentes celdas y con una dirección inicial aleatoria. Se tomó como condición inicial, que no haya colisiones en el tiempo inicial, es decir que no haya más de una partícula por celda.

4. Simulaciones

Para las simulaciones se decidió tomar en cuenta diferentes situaciones para poder analizar cómo cambia el número de iteraciones en relación a N y a D. En algunos casos también se decidió cambiar ϵ para estudiar más en profundidad el sistema. Para todas las simulaciones se decidió mantener la cantidad de celdas.

En un principio, fijando $D = 50$, se analizó con $N = [2000, 3000, 5000, 7000]$ la cantidad de iteraciones para llegar al equilibrio establecido en la ecuación (2) de la sección 3.2.2. Para $N = 2000$, se utiliza $\epsilon = 0{,}1$, para $N = 3000$, $\epsilon = 0{,}05$. Con $N = 5000$ y $N = 7000$ se utilizo $\epsilon = 0{,}01$.

Luego, para poder visualizar bien la relación entre N y la cantidad de iteraciones, se tomaron los valores $N = [2000, 3000, 4000, 5000, 6000, 7000, 8000, 9000, 10000]$, con $D = 50$ y $\epsilon = 0{,}1$. También se decidió hacer este análisis con $N = [2000, 2200, 2400, 2600, 2800, 3000]$ y con D y ϵ igual que antes. Para todos los casos mencionados, se realizaron 10 corridas por cada N.

Por ultimo, para determinar la relación entre D y la cantidad de iteraciones, se eligió hacerlo para $N = [2000, 3000, 5000]$, donde en cada caso se tomo $D = [25, 50, 75, 100, 125, 150, 175, 200]$. Es decir, que para $N = 2000$, se corrió con $D = [25, 50, 75, 100, 125, 150, 175, 200]$, y así para todos los N. Se mantuvo para todos los casos $\epsilon = 0{,}1$. Para cada uno de los casos, se realizaron 10 corridas por cada D.

Para las visualizaciones se usaron 5 casos, donde $\epsilon = 0.1$:

- $N = 2000$, $D = 50$.
- $N = 5000$, $D = 50$.
- $N = 7000$, $D = 50$.
- $N = 5000$, $D = 25$.
- $N = 5000$, $D = 150$.

5. Resultados

Para visualizar los resultados se utiliza un software Open Source llamado Ovito. Debido al modelado del sistema, se decidió optar por una visualización basada en cuadrados y su densidad en cada momento. Estos cuadrados representan un grupo de 5 x 5 hexágonos. De esta forma, se visualizan en pantalla

como máximo solamente 1.600 elementos, mientras que si se hubiese mantenido la proporción un cuadrado a una celda se tendrían que renderizar 40.000 elementos. La densidad del cuadrado no es mas que la cantidad de partículas que están situadas en ese cuadrado para ese preciso momento. Un color claro representa un baja cantidad de partículas, mientras que un color mas oscuro indica que hay una mayor densidad en ese cuadrado. Si no hay ninguna partícula en un cuadrado particular, ese cuadrado no se renderiza.

En las siguientes figuras se puede observar las visualizaciones a modo ilustrativo del sistema en el estado inicial y en el estado final, cuando se alcanzo el equilibrio.

Figura 5: Estado inicial del sistema. Todas las particulas estan situadas en el espacio izquierdo.

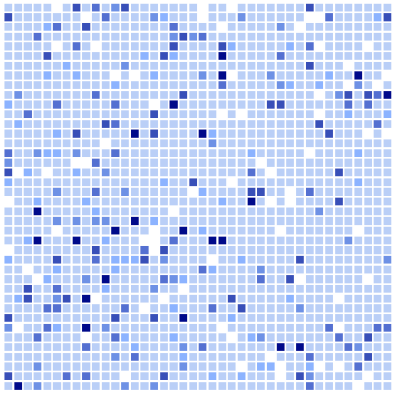

Figura 6: Estado final del sistema, cuando se alcanza el equilibrio.

5.1. Iteraciones vs. Densidad de Partículas por Cámara

Uno de los criterios más importantes a la hora de demostrar el funcionamiento del sistema es el porcentaje de partículas en cada cámara a través de las distintas iteraciones. En las próximas figuras se pueden ver los resultados encontrados, variando las condiciones iniciales N, D y e.

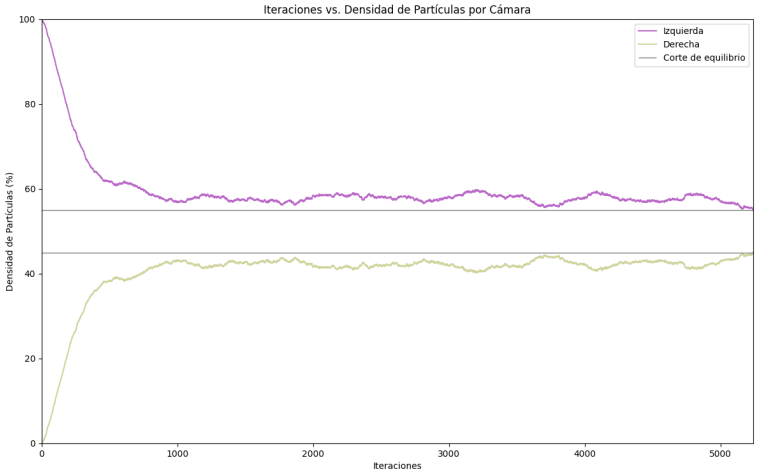

Figura 7: Iteraciones en función de la densidad de partículas por cámara con n=2000, d=50 y e=0.1

Cabe aclarar que la Fig. 7 es la única con un N de 2000 y D de 50, debido a que con un ϵ menor a 0.1 la simulación llegaba a iteraciones del orden de 10^5, donde la densidad de las cámaras oscilaba sin nunca cruzar el umbral de equilibrio. Para el caso mostrado, la cantidad iteraciones hasta el equilibrio es de 5241.

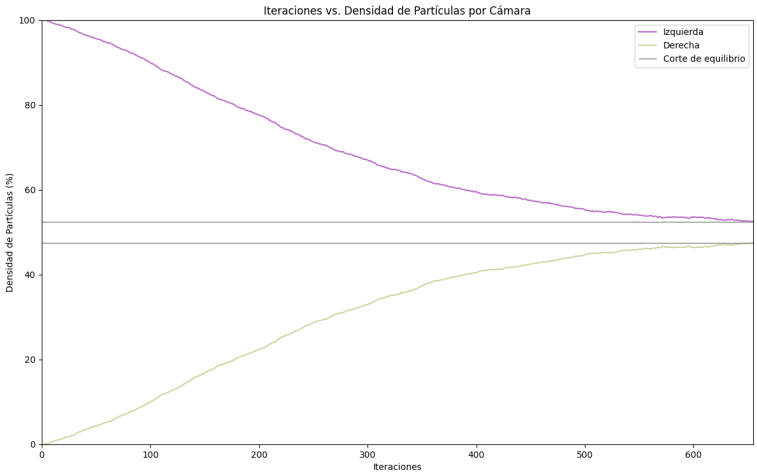

Figura 8: Iteraciones en función de la densidad de partículas por cámara con n=3000, d=50 y e=0.05

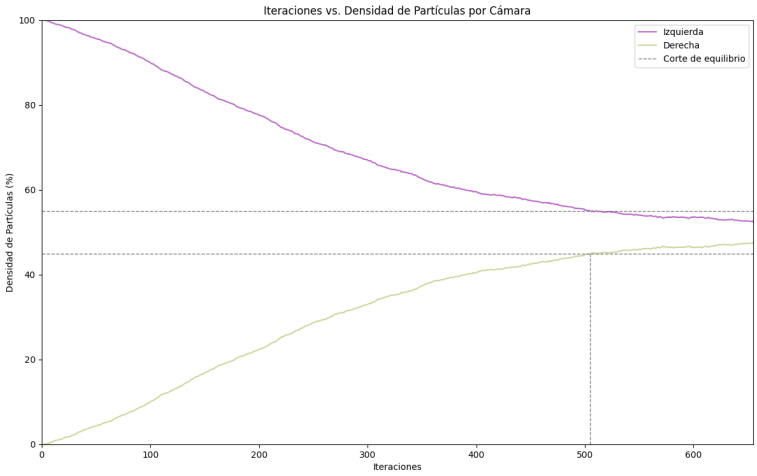

Figura 9: Iteraciones en función de la densidad de partículas por cámara con n=3000, d=50 y e=0.1

De una manera similar a la ocurrida con $N = 2000$, en el caso de $N = 3000$, al correrlo con $\epsilon = 0.01$, la simulación no alcanzaba el equilibrio establecido. Es por esto, que se corrio con $\epsilon = 0.01$ como se puede ver en la Fig. 8. Para la Fig.

9, se puede ver como seria en caso de haberlo corrido con $\epsilon = 0.1$. Para el primer caso, se llego al equilibrio en 655 iteraciones y en el segundo, en 505 iteraciones.

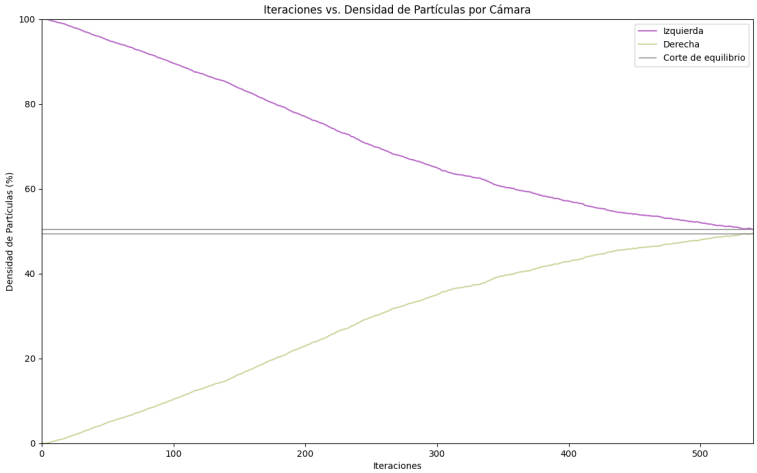

Figura 10: Iteraciones en función de la densidad de partículas por cámara con n=5000, d=50 y e=0.01

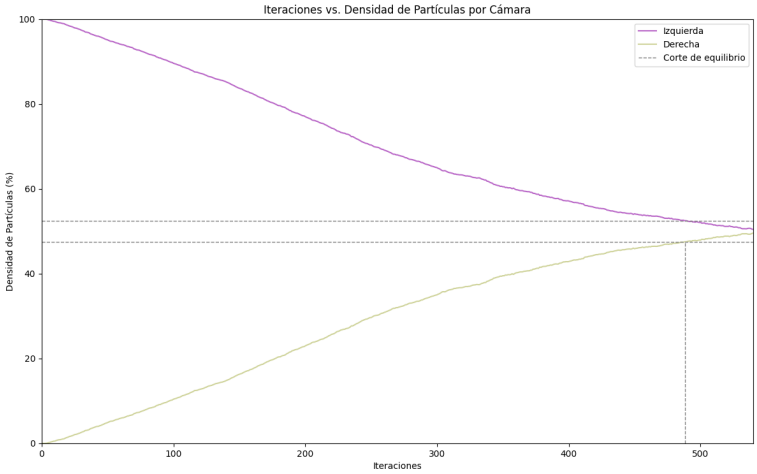

Figura 11: Iteraciones en función de la densidad de partículas por cámara con n=5000, d=50 y e=0.05

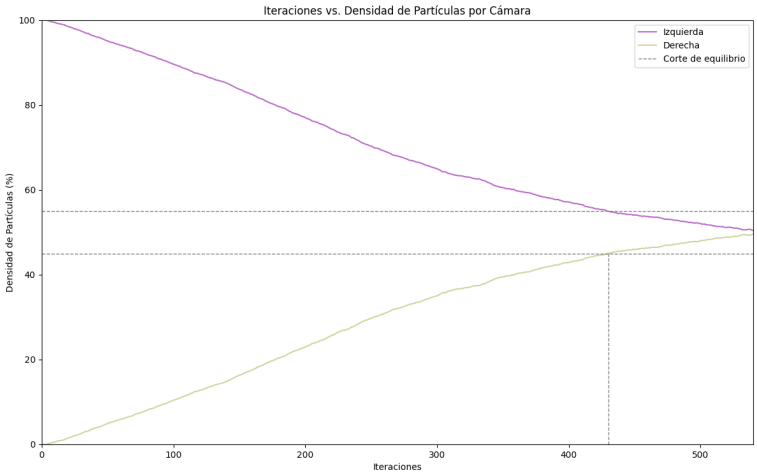

Figura 12: Iteraciones en función de la densidad de partículas por cámara con n=5000, d=50 y e=0.1

En el caso de $N = 5000$, se logro correr con un $\epsilon = 0.01$, llegando a el equilibrio en 540 iteraciones, como se puede observar en la Fig. 10. En las Figs. 10 y 12, se muestra como hubiese sido la evolución del sistema habiendo elegido ϵ de 0.05 y 0.1 respectivamente. Para el primer caso, la cantidad de iteraciones hasta llegar al equilibrio fue de 488 y en el segundo, de 430.

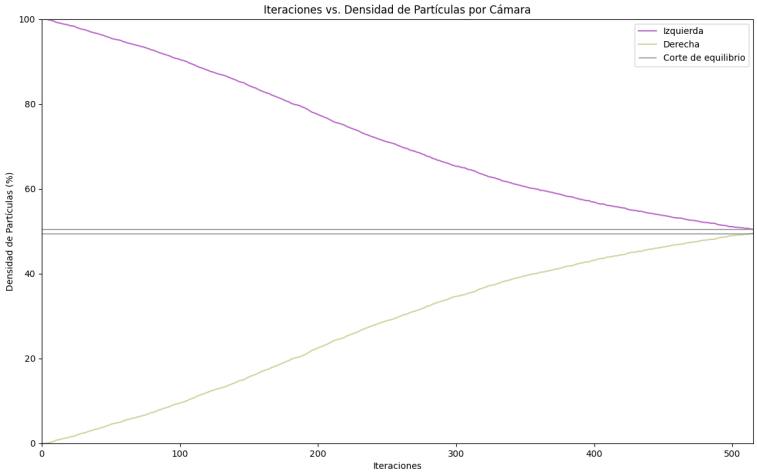

Figura 13: Iteraciones en función de la densidad de partículas por cámara con n=7000, d=50 y e=0.01

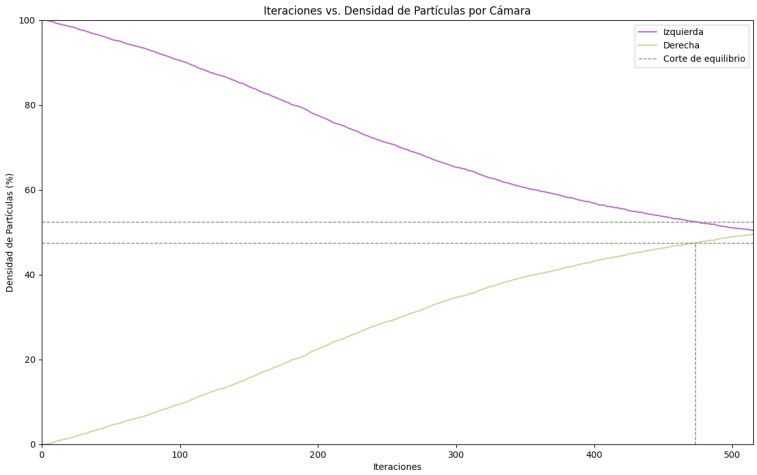

Figura 14: Iteraciones en función de la densidad de partículas por cámara con n=7000, d=50 y e=0.05

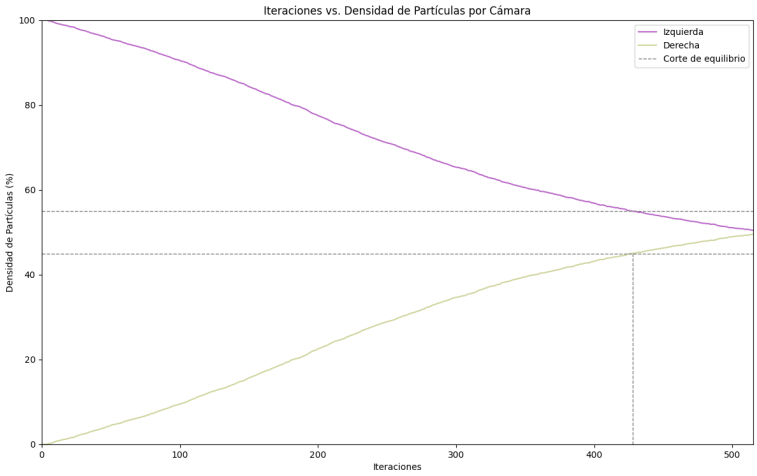

Figura 15: Iteraciones en función de la densidad de partículas por cámara con n=7000, d=50 y e=0.1

Al igual que para $N = 7000$, esta situación se corrió con $\epsilon = 0.01$. Como se puede ver en la Fig. 13 se llegó al equilibrio en 499. Para las Figs. 14 y 15, la cantidad de iteraciones disminuye a 473 y 428 respectivamente.

Es importante remarcar que todos los gráficos utilizan la unidad *iteraciones* en vez de tiempo ya que este último depende de las especificaciones de la computadora que corre las simulaciones, por lo que no son representativos.

5.2. Iteraciones vs. Cantidad de Partículas

Todos los resultados presentes en esta sección se obtuvieron corriendo el programa 10 veces.

En las siguientes figuras se observa el impacto de la cantidad de partículas en el sistema sobre la cantidad de iteraciones promedio que se tarda en alcanzar el equilibrio, a través de las 10 corridas. Las barras de error representan la desviación estándar de las iteraciones calculadas entre todas las corridas por cada N.

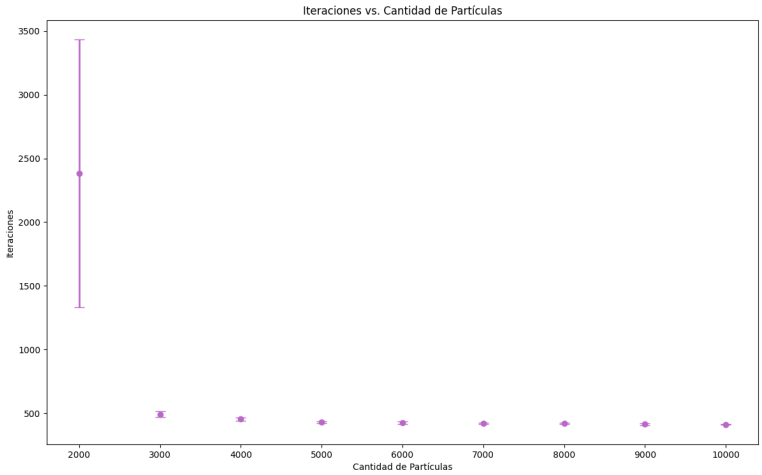

Figura 16: Cantidad de iteraciones necesarias hasta alcanzar el equilibrio en función de la cantidad de partículas, con un rango entre 2000 y 10000

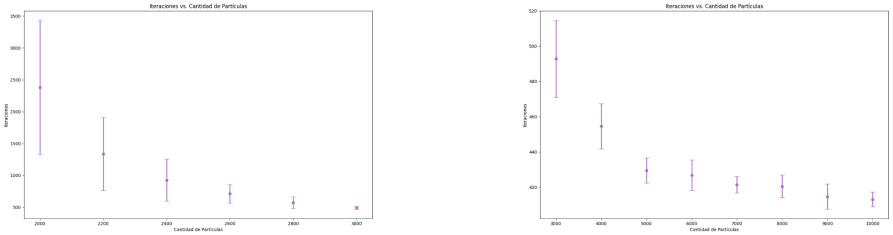

Figura 17: Cantidad de iteraciones necesarias hasta alcanzar el equilibrio en función de la cantidad de partículas. el primer gráfico tiene un rango entre 2000 y 3000, mientras que el segundo tiene un rango entre 3000 y 10000

En la Fig. 16, se pueden observar las iteraciones correspondientes a cada N elegido. Sin embargo, al haber una diferencia tan grande entre $N = 2000$ y $N = 3000$, se decidió realizar la Fig. 17. En el lado derecho se puede ver en más profundidad lo que sucede entre $N = 2000$ y $N = 3000$.

5.3. Iteraciones vs. Tamaño del Tabique

Las próximas figuras representan la cantidad de iteraciones necesarias para alcanzar el equilibrio en función del tamaño del tabique para $N = [2000, 3000, 5000]$.

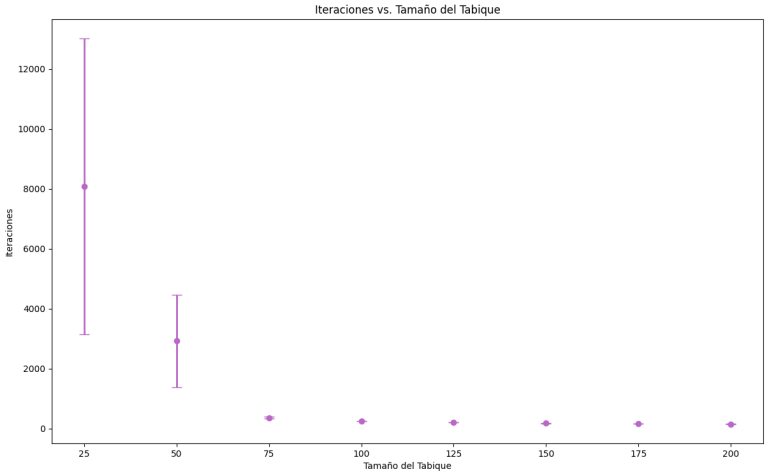

Figura 18: Cantidad de iteraciones necesarias para alcanzar el equilibrio con N=2000 y un tamaño de tabaque entre un rango de 25 y 200.

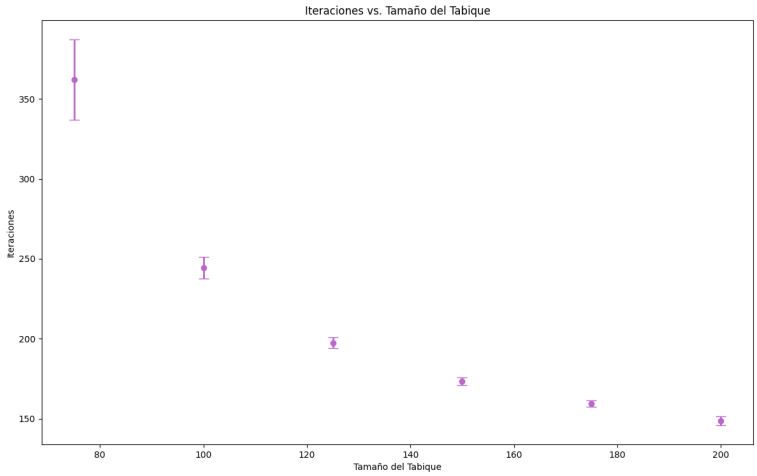

Figura 19: Cantidad de iteraciones necesarias para alcanzar el equilibrio con N=2000 y un tamaño de tabaque entre un rango de 75 y 200.

En la Fig. 18 se puede observar los cambios de cantidad de iteraciones, según el tamaño del tabique. Sin embargo, al haber tanta diferencia entre los tabiques $D = 25$ y $D = 50$ y el resto del los tamaños, se decidió graficar la Fig. 19 para poder observar mejor el resto de los tamaños de tabique.

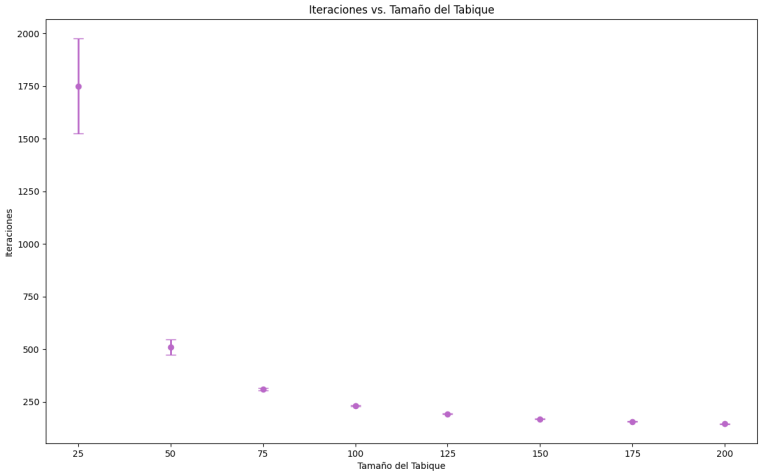

Figura 20: Cantidad de iteraciones necesarias para alcanzar el equilibrio con N=3000 y un tamaño de tabaque entre un rango de 25 y 200.

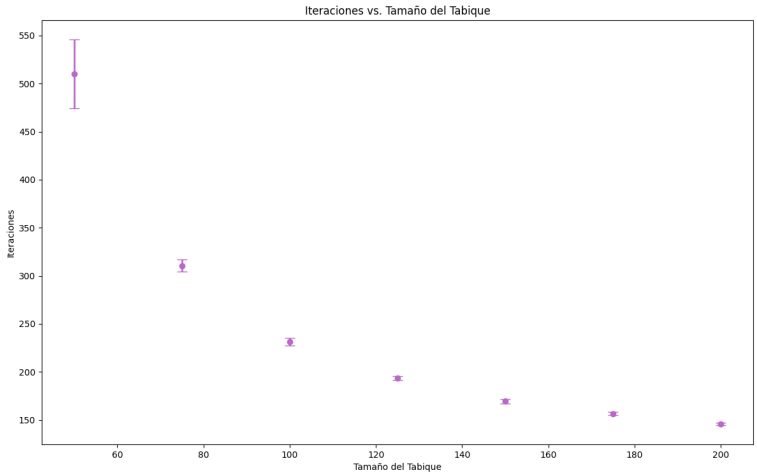

Figura 21: Cantidad de iteraciones necesarias para alcanzar el equilibrio con N=3000 y un tamaño de tabaque entre un rango de 50 y 200.

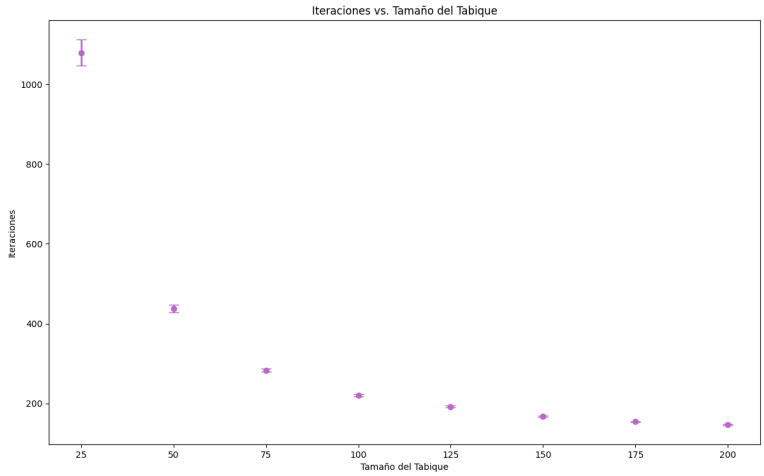

Figura 22: Cantidad de iteraciones necesarias para alcanzar el equilibrio con n=5000 y un tamaño de tabaque entre un rango de 25 y 200.

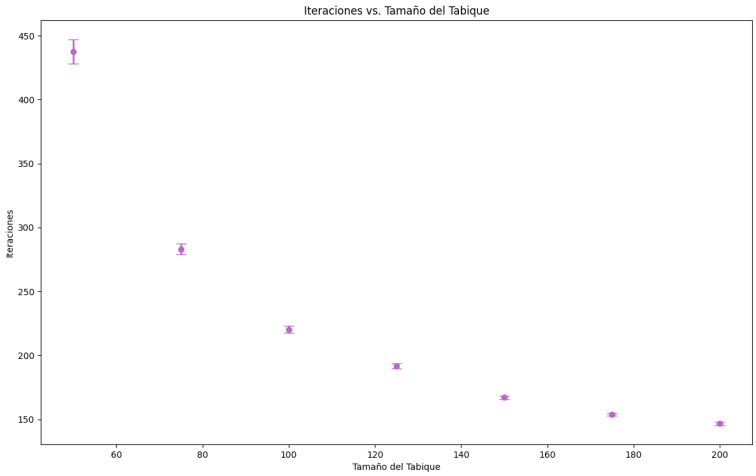

Figura 23: Cantidad de iteraciones necesarias para alcanzar el equilibrio con n=5000 y un tamaño de tabaque entre un rango de 50 y 200.

De la misma manera que sucedió en el caso de $N = 2000$, para los casos de las Figs. 20 y 22, el caso de $D = 25$ es muy diferente del resto, por lo que no se puede analizar bien como se comporta el sistema con D mas grandes. Por esta razón, se realizaron los gráficos de las Figs. 21 y 23 para $N = 3000$ y $N = 5000$ respectivamente.

6. Conclusiones

A partir de los gráficos observados se pueden tomar varias conclusiones respecto al funcionamiento del sistema. En primer lugar, se puede ver una clara correlación entre la cantidad de iteraciones y la densidad de partículas en cada cámara. En un principio la velocidad con la que cambia la densidad es alta, y lentamente disminuye hasta alcanzar una especie de meseta, donde se queda oscilando hasta alcanzar el criterio de equilibrio establecido.

Otra observación interesante puede ser como, a medida que aumenta el valor de N, una disminución en el valor de ϵ cada vez afecta menos las iteraciones necesarias.

Respecto a la cantidad de partículas en el sistema, para los valores estudiados se puede observar una reducción en iteraciones necesarias para alcanzar el equilibrio a medida que estas aumentan. Sin embargo, ocurre un fenómeno interesante cuando la cantidad de partículas es menor a 3000. En estas condiciones se puede observar no solamente un rápido incremento en iteraciones necesarias, sino también un gran aumento en el desvío estándar. A comparación con la configuración con N=2000, los sistemas con cantidad de partículas en el rango entre 3000 y 10000 parecen tener un comportamiento muy similar.

Por último, a partir de los gráficos presentados anteriormente se ve como un menor tamaño de tabique aumenta drásticamente la cantidad de iteraciones necesarias para alcanzar el equilibrio. Sin embargo, parece ser que una vez alcanzado un tamaño de tabique de 75, la disminución en cantidad de iteraciones parece ser sustancialmente menor.

Referencias

[1] Dieter A. Wolr-Gladrow. *Lattice-Gas Cellular Automata and Lattice Boltzmann Models - An Instroduction*. Springer, 2005.