Off-Lattice

Trabajo Práctico N°2

Sistemas de Simulación

Martín **Grabina** & Juan **Grethe**

Marzo de 2019

Instituto Tecnológico de Buenos Aires

Índice

1 | Introducción

2 | Fundamentos Teóricos

3 | Implementación
- 3.1 | Entrada
- 3.2 | Algoritmo
- 3.3 | Salida

4 | Resultados
- 4.1 | Va en función de η
- 4.2 | Va en función de δ

5 | Conclusiones

6 | Bibliografía

1 | Introducción

El objetivo de este trabajo práctico es el desarrollo y análisis del algoritmo Off-Lattice presentado en clase teórica, para analizar comportamientos en base a distintos parámetros mediante tablas de resultados y animaciones.
Esto es implementar un sistema de simulación de partículas que trata el paper *Novel Type of Phase Transition in a System of Self-Driven Particles*.

2 | Fundamentos Teóricos

Un autómata celular (A.C.) es un modelo matemático para un sistema dinámico que evoluciona en pasos discretos. Es adecuado para modelar sistemas naturales que puedan ser descritos como una colección masiva de objetos simples que interactúen localmente unos con otros.
Detalles del sistema Off-Lattice (Bandadas de agentes autopropulsados), se puede ver en este paper.

3 | Implementación

Se partió de la base del trabajo práctico n°1, neighbors detection, aprovechando las clases *Input, Output, Grid, Particle y Neighbor Détection*, las cuales se modificaron y se le sumaron las específicas de *Off Lattice*, con la idea de reutilizar el código del algoritmo *Cell Index Method*.
El desarrollo duró una semana y se utilizó Java 8 para aprovechar las optimizaciones que *Java 8 Streams* permite, como por ejemplo *parallel streams*.
Se desarrolló de forma tal que quedan modularizados la generación de la simulación y la generación de la animación en sí, permitiendo entradas al programa desde archivos o generación random, lo que facilita el análisis posterior de los resultados.
Para las animaciones se utilizó *Ovito*, al cual le ingresamos el archivo *positions.xyz* generado por nuestro programa con los parametros: *de, posición x, posición y, velocidad x, velocidad y* (para cada instante de tiempo, frame). Y una vez importada la simulación en Ovito, allí se eligen los colores y demás detalles de la animación que luego será renderizada en formato *.avi* y *.gif*.

3.1 | Entrada

- **Random**
  Parámetros obligatorios: N (cantidad de partículas),ŋ (ruido máximo), L (largo del lado del sistema).
  Parámetros opcionales: -y (cantidad de iteraciones).
  De esta forma se tomarán estos valores como parámetros y se generarán los demás necesarios de forma aleatoria.
  Ejemplo:

```
java -jar off_lattice.jar -i=300 -N=1000 -L=10 -n=0
```

- **Mediante Archivos**
  Parámetros obligatorios: ŋ (ruido máximo), s (archivo estático), de (archivo dinámico).
  Parámetros opcionales: -i (cantidad de iteraciones).

  De esta forma se cargarán los valores indicados por parámetro y los demás necesarios provenientes de los archivos.

```
java -jar off_lattice.jar -i=300 -n=0 -L=30 -N=500 -s=static.txt
-d=dynamic.txt
```

Nota: Los formatos de los archivos estáticos y dinámicos corresponden a los señalados en el enunciado del trabajo práctico n°1.

```
1   0
2   23.83135852108391 8.36559285769604 0.22714903885955412 0.1959676354533594
3   23.592415618347562 5.8253015613205665 -0.09460671333596514 0.2846920613430703
4   0.9743714285059146 21.56574476582413 0.2991530719639927 7 -0.022526418590316323
5   6.274378620538103 14.319455573128437 -0.07244025233009914 -0.29112267146746157
6   16.729636814311345 21.28744191591441 0.04145442664529355 0.29712208014805963
7   19.556670189604965 22.375829040563367 -0.23038235111159888 -0.19215611438695357
8   11.15055225546101 2.564944056547297 0.1289562230275591 -0.2708695120209479
9   12.204535546678356 26.835662985033803 0.2058790984316959 0.21820585883278218
10  9.436728882890193 12.719528033345663 6.994814915001511E-4 0.2999991845416301
11  25.161688533487908 16.617141756192797 0.25012144471598063 0.1656480090227185
12  27.46751911404847 24.777280862760456 0.24011040074295162 -0.17985270488669103
13  21.037791363549523 2.670278230833948 0.04001180914830967 -0.29731978596904585
14  21.060635025187807 3.4452786432939684 -0.022901779494618135 0.2991245701977353
15  14.161269267455518 20.90713269169863 -0.29927970481802013 0.020776387656153052
16  22.111381375307474 24.2745567300026 0.2988088432850898 -0.026707212033955107
```

Imagen de ejemplo 1: Fragmento de ejemplo del archivo dinámico de entrada.

```
1   2000
2   30
3   0.6820683326727655 Rojo
4   0.2620591317898712 Rojo
5   0.40389417024456187 Rojo
6   0.2742549033678653 Rojo
7   0.3610334423506972 Rojo
8   0.2602813970753463 Rojo
```

Imagen de ejemplo 2: Fragmento de ejemplo del archivo estático de entrada.

3.2 | Algoritmo

A partir del input creado de manera aleatorio o ingresado, el programa comienza a simular la posición de la siguiente manera:

1) Cada partícula tiene una posición X e Y, una velocidad(v) y su ángulo(θ). Con estos parámetros se calcula la siguiente posición que va a salir de:

$$X(t+1) = X(t) + cos(\theta) * \Delta t \quad (1)$$
$$Y(t+1) = Y(t) + sin(\theta) * \Delta t \quad (2)$$

2) Una vez actualizada la posición se calcula el nuevo ángulo. Para esto por cada partícula, utilizando el algoritmo de Cell Index Method, se obtienen los vecinos y se realiza un promedio de todos sus ángulos con las siguientes fórmulas:

$$\theta(t+1) = <\theta(t)>_r + \Delta\theta \quad (3)$$

donde $<\theta(t)>_r$ representa el promedio de los ángulos de todas los agentes dentro de r incluyendo al propio agente y $\Delta\theta$ es el ruido generado de manera aleatoria que varía entre (-n/2,n/2), siendo n el parámetro ingresado en el input.

A partir del promedio se obtiene el nuevo ángulo con la siguiente fórmula:

$$atan2(\frac{<sin(\theta(t))>_r}{<cos(\theta(t))>_r}) \quad (4)$$

3) Se vuelve al paso 1 si el t actual es menor a la cantidad de iteraciones ingresadas en el input. Caso contrario finaliza la ejecución y imprime en la salida el archivo para visualizar las posiciones de las partículas.

3.3 | Salida

Archivo ***positions.xyz***
Archivo de formato *.xyz* (columnar) que provee la información de cada estado de la simulación, donde las columnas representan (en este orden):

1. ID de la Partícula
2. Componente X de la Posición
3. Componente Y de la Posición
4. Componente X de la Velocidad
5. Componente Y de la Velocidad

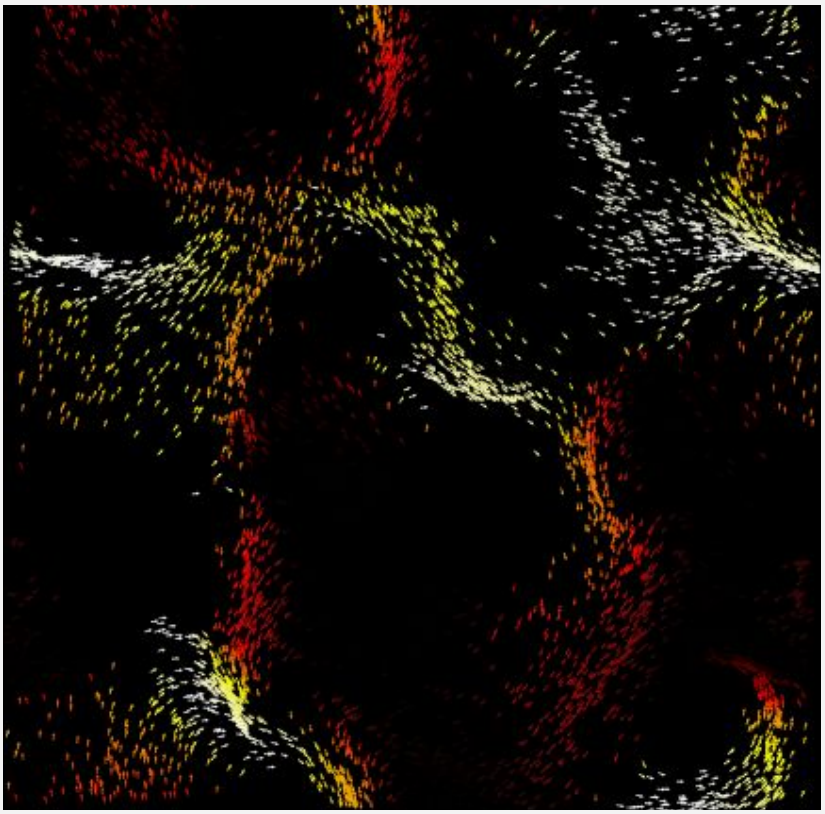
Además, se crean/actualizan los archivos de entrada estatico y dinamico con los valores utilizados en la ejecución para poder volver a replicar el mismo contexto.

```
500

440 29.7765654292317 10.510055378585832 -0.286017537544863 0.09052054030314279
354 3.613018279577007 7.577931668464958 0.12162601185851896 -0.2742391533668951
297 22.68239667322975 8.946269768547236 0.0509883908533774 0.29563522117329527
183 13.915435017481007 29.407637275905007 -0.09107773046836694 0.28584059720888405
450 26.896338088872636 1.6653131727724613 0.23078352089451837 0.19167411531954284
97 27.420948395010452 16.328017307941252 -0.27798796404544 -0.1127949105495064
460 29.11224656418968 15.536703099281846 -0.2998750450460077 0.00865779178860476
64 1.319337987149376 2.9385330609058293 -0.1701353305332426 -0.2470910142120597
421 24.89694210166996 7.14701105348008 -0.2390053607466813 0.1813186077994995
35 20.67286077829207 18.221290114111277 -0.13884922267761063 0.2659340018911145
340 3.5606664992519166 4.614614229784893 0.27678713856208015 0.11571032765754247
163 1.1138805483449499 17.551199130115336 -0.2994658953573896 -0.017893504905327772
286 27.962166764934754 1.4712910682560387 -0.29164082060417784 0.07032518579656748
141 10.910179043304048 8.01684608658203 -0.2847469600226765 -0.09444135088955612
398 26.63050308506465 6.363738574972783 -0.13234123526735103 0.2692318655878454
411 23.981815599299175 24.486925962488158 0.011305280286284875 -0.2997869087162555
```

Imagen de ejemplo 3: Fragmento de ejemplo del archivo *positions.xyz*

4 | Resultados



Animación de un sistema Off-Lattice sin ruido y con 8000 partículas.

4.1 | Va en función de η

Para realizar estas simulaciones se utilizaron valores similares a los encontrados en el paper. Estos son N = 40, 100 y 400 ,y L = 3, 5 y 10 respectivamente. El rango de η fue entre 0 y 5 variando de a 0,25. Para cada simulación de N, L y η se realizaron 10 simulaciones en la cual se calculó el promedio y la desviación estándar. A continuación se pueden visualizar los resultados obtenidos:

Valores obtenidos de ejecuciones del programa

Ruido	N = 40 ; L = 3	N = 100 ; L = 5	N = 400 ; L = 10
0	1,00	1,000	1,000
0,25	0,997	0,997	0.996

0,5	0,989	0,988	0,987
0,75	0,974	0,972	0.968
1	0,955	0,953	0.948
1,25	0,929	0,930	0.907
1,5	0,906	0,900	0.868
1,75	0,876	0,853	0.837
2	0,834	0,821	0.778
2,25	0,797	0,779	0.739
2,5	0,759	0,738	0.669
2,75	0,689	0,683	0.612
3	0,658	0,613	0.558
3,25	0,605	0,559	0.488
3,5	0,535	0,548	0.417
3,75	0,49	0,411	0.314
4	0,447	0,373	0.220
4,25	0,393	0,318	0.152
4,5	0,298	0,179	0.091
4,75	0,213	0,151	0.042
5	0,167	0,122	0.044

Tabla 1: Valores de Va en función del ruido para distintos contextos (pares de valores de N y L)

Gráfico 1: Va en función del ruido para N=40 y L=3, con barras de errores calculadas como desviación estándar en cada caso y línea de tendencia polinómica.

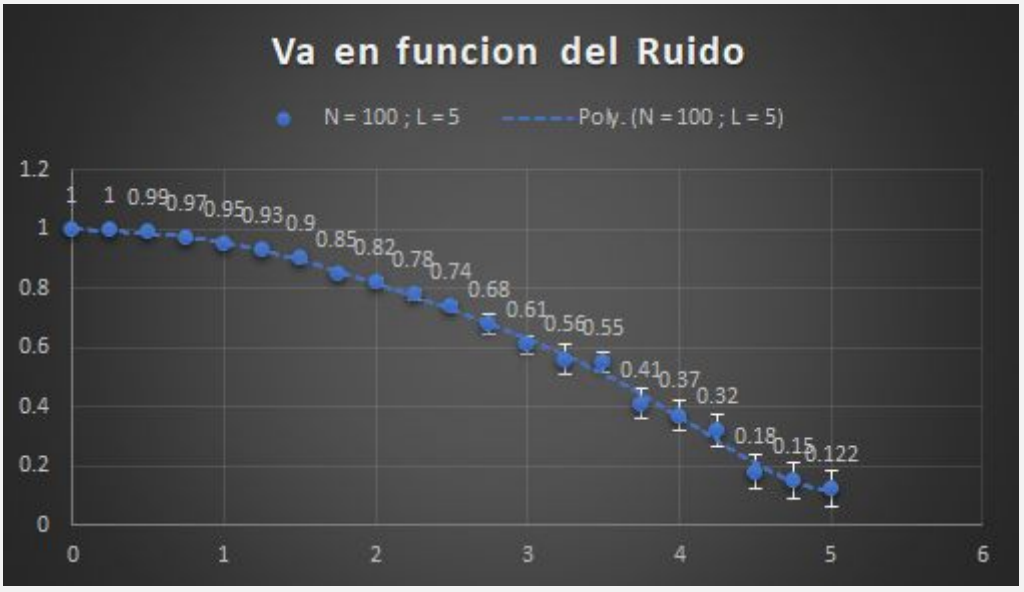


Gráfico 2: Va en función del ruido para N=100 y L=5, con barras de errores calculadas como desviación estándar en cada caso y línea de tendencia polinómica.

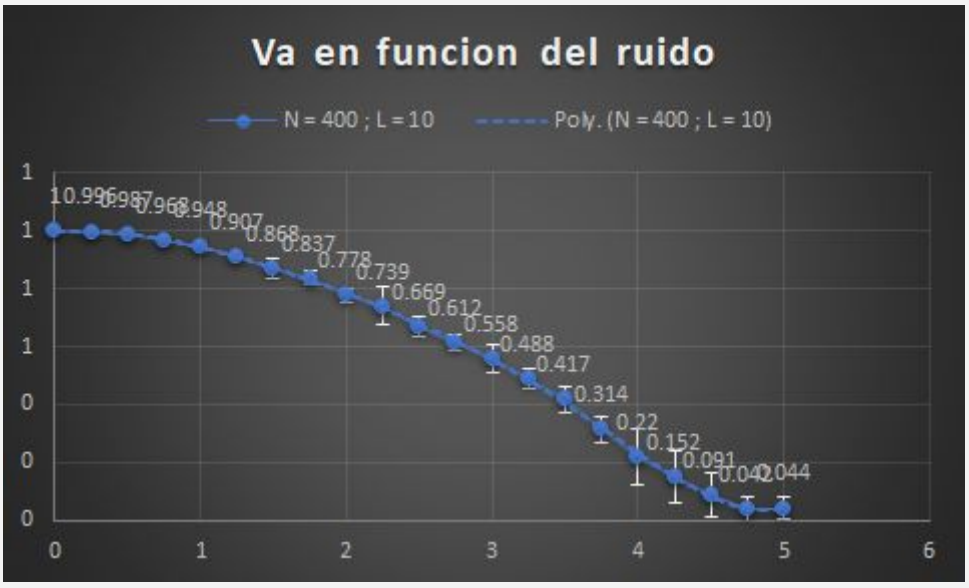


Gráfico 3: Va en función del ruido para N=400 y L=10, con barras de errores calculadas como desviación estándar en cada caso y línea de tendencia polinómica.

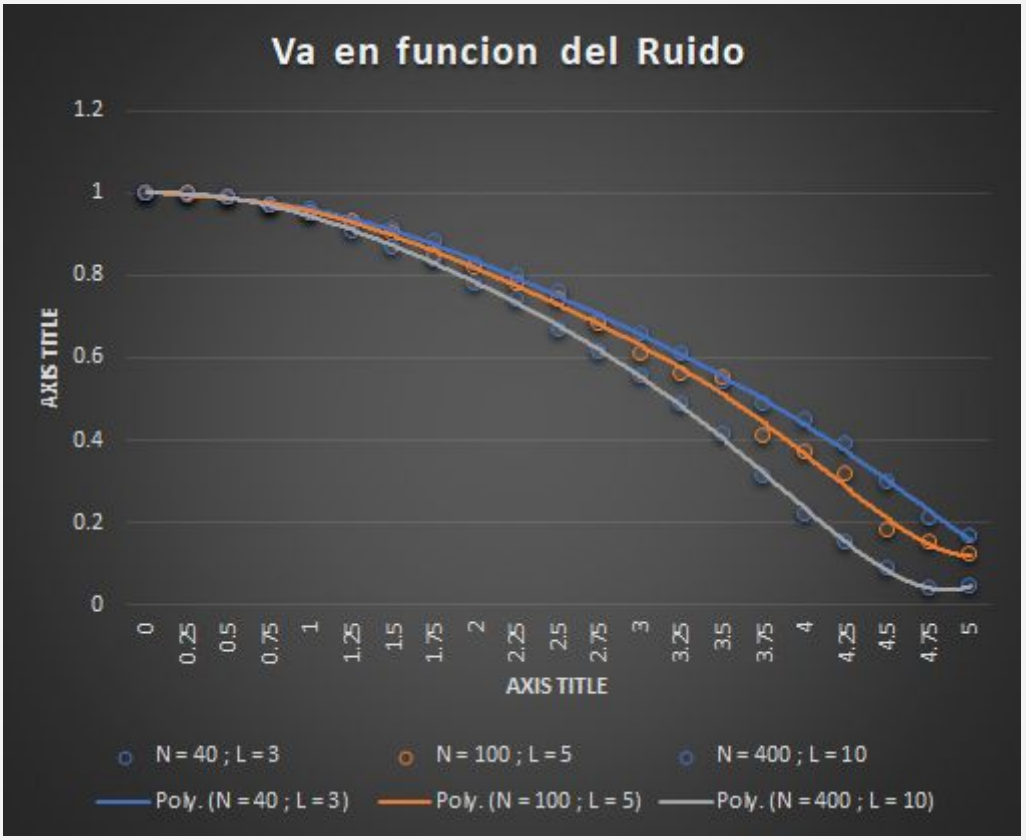


Gráfico 4: Comparativa de las relaciones de Va en función del ruido para los distintos valores obtenidos en la tabla.

Para el gráfico 4 se utilizó una línea de tendencia polinómica para poder visualizar de una manera más clara las curvas obtenidas. Con esta se puede comprobar que a un mayor N se obtienen unas curvas más precisas. Esto se debe a que a un N más grande, las partículas se encuentran más cerca una de las otras en un primer instante cuando se setean las partículas de manera aleatoria en el tablero. Con un N más chico es mucho más probable que en un principio las partículas se encuentran muy separadas unas de otras y por lo tanto varía más el resultado final.

A la vez se puede ver claramente en los gráficos 1,2,3 y 4 que al aumentar el η, Va tiende a 0, ya que el ruido es mayor, y este factor de aleatoriedad dificulta que las partículas se alinean.

4.2 | Va en función de δ

Para esta simulación se utilizó L= 20 y η = 2. Se fue variando la densidad del sistema de 0,1 entre los rangos de 0,1 a 1 y de a 1 entre 1 y 10. Dado que la densidad es igual a $\delta = \frac{N}{L^2}$, para un determinado δ y L, no existe un N entero para representarlo. Por lo que se calculó con el N entero más próximo y el δ correspondiente. Para cada L, δ y η se realizaron 10 simulaciones en las que se calculó el promedio y la desviación estándar. A continuación se pueden observar los resultados obtenidos.

Va en función de la densidad

Densidad	Promedio	Error
0.1	0.2677860466535483	0.10230866228693297
0.2	0.2915946781517072	0.15611963602923298
0.3	0.4796983650194913	0.10529132561520113
0.4	0.4149419293716329	0.12043864540204324
0.5	0.4287747693298777	0.10817359347158516
0.6	0.5761299072387626	0.08709658062883782
0.7	0.6386944873987612	0.11087571196105397
0.8	0.6426852184400536	0.054360914403601274

0.875	0.6679971834578329	0.05681472311261776
0.9975	0.6130871532296319	0.08016764827143351
2.0	0.7301858360444338	0.055099042430619465
3.0	0.6925321089731867	0.14057027426543836
4.0	0.7068103154921666	0.126258030912252
5.0	0.7484245759878093	0.045123447788070065
6.0	0.6963304975487417	0.17843067820159403
7.0	0.7815878104462783	0.031153908998564166
8.0	0.7653205126887274	0.074907551358747
9.0	0.7601951422874119	0.06247923868442732
10	0.7824015441325993	0.05521556753561914

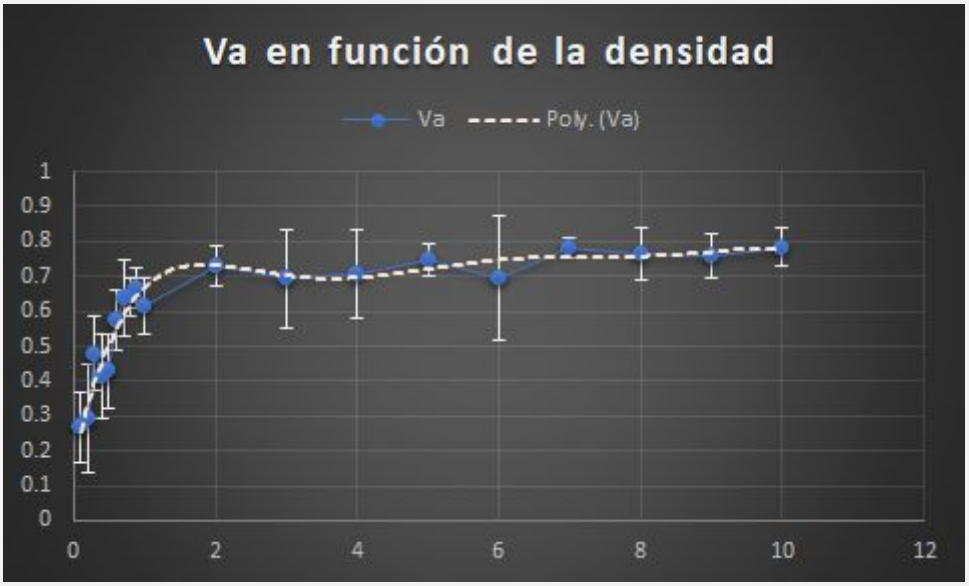


Gráfico 5: Comparativa de las relaciones de Va en función de la densidad.

En el gráfico 5 también se dibujó los resultados obtenidos con una línea de tendencia polinómica para poder visualizarlos. Observando el gráfico, se pudo comprobar que la densidad también incide en la polarización del sistema. A una mayor densidad las partículas en encuentran más polarizadas, mientras que cuando la densidad es menor las partículas se observan más dispersas.

5 | Conclusiones

Se logró simular el comportamiento de sistemas de partículas en dos dimensiones con un autómata celular respetando el modelo Off-Lattice.
Se concluye en base a los gráficos obtenidos que a menor ruido el Va casi no varia en funcion a los demas parametros del sistema, y que cuando el ruido tiende a un número muy alto vuelve a suceder lo mismo (pero con un margen de error mucho mayor), mientras que en el medio (ver valores de ruido entre 2 y 5, si varía en función al contexto de la simulación.
En el caso de la densidad, es claro que a menor densidad se produce un menor Va, y crece casi exponencialmente hasta encontrar “el equilibrio”.
Por lo tanto, a mayor nivel de ruido el sistema se encuentra en mayor desorden mientras que a menor ruido las partículas se encuentran más polarizadas.
Por otro lado, se llegó a la conclusión que la densidad, como valor calculado en base a los parámetros del sistema, resulta directamente proporcional a la polarización de las partículas.
Finalmente se observan valores y tendencias se que se correlacionan en gran medida con las presentadas en el paper *Novel type of phase transition in a system of self-driven particles*, lo cual resulta acorde a lo esperado a la hora del desarrollo del mismo.

6 | Bibliografía

- Lineamientos de redacción de informes
- Enunciado
- Clase teórica de presentación del algoritmo
- Bibliografía de ayuda para los sistemas
- Formato XYZ
- Manual Ovito