Propagación de una Epidemia mediante Autómatas Celulares

Julián Jiménez Cárdenas

juojimenezca@unal.edu.co

Juan Sebastián Ordóñez Soto

jsordonezs@unal.edu.co

Herramientas Computacionales, Departamento de Física, Universidad Nacional, Bogotá.

Noviembre 29, 2016

Resumen

En el presente artículo se muestran los resultados obtenidos al reproducir el algoritmo descrito en [1], en éste los autores buscaban simular la evolución de una epidemia mediante difusión utilizando un objeto llamado Autómatas Celulares*, con acciones bien definidas (reglas de evolución) en cierto intervalo de tiempo. Asimismo, se comprueba la validez del método al compararlo con el modelo **SIR**, comúnmente utilizado en el modelamiento de este fenómeno.*

I. Introducción al Problema

El objetivo de la simulación es reproducir el comportamiento temporal de una epidemia por difusión mediante autómatas celulares. Para ello, deben estar ubicados en un mismo punto al principio de la simulación y después irse moviendo a lo largo del espacio previamente discretizado.

Referente al autómata celular, es un objeto (en este caso, una estructura de `C++`) regido por reglas de evolución simples descritas en la subsección *Algoritmo*. Las variables determinantes para simular el comportamiento son: el número de autómatas celulares N, la probabilidad que tiene un autómata infectado de curarse $PInm$, el tamaño de la malla bidimensional en la que estarán los autómatas, el número inicial de infectados, la probabilidad de que un autómata se infecte por la presencia de autómatas infectados r y el tiempo de evolución del sistema (en unidades arbitrarias).

Después de reproducir los resultados del modelo **SIR**, se analizaron algunos casos de prueba, así como la relación existente entre la probabilidad de volverse inmune siendo infectado $PInm$ contra el número final de autómatas susceptibles.

1. Algoritmo

Los autómatas celulares fueron programados para seguir las siguientes reglas. En cada paso de tiempo se deben realizar las siguientes acciones.

Movimiento Difusivo

1. La célula rotará con igual probabilidad hacia arriba, abajo, derecha o izquierda.

2. La célula se moverá en la dirección a la cual apunta. Hay que tener precaución, pues la célula no se debe salir de los límites del mapa.

Contagio Epidémico

1. Si la célula está inmune sigue inmune.
2. Si está infectada se cura con probabilidad $PInm$.
3. Si está sano y tiene infectados en el mismo punto o a sus alrededores, se calcula la probabilidad que éste tiene de infectarse proporcionalmente al número de infectados alrededor suyo. Mediante dicha probabilidad se determinará si se infecta la célula o no.

Cabe hacer énfasis en que las celdas en las que se ubican los autómatas celulares y éstos son distintos.

II. Código

Los programas encargados de calcular el comportamiento de los autómatas celulares se hicieron en `C++`. De éstos se generaban archivos de texto plano, para ser posteriormente leídos y relacionados mediante scripts de `Python`. En aras de generar números aleatorios en `C++`, se usó la librería `random` y el generador `mt19937`. Con estas herramientas se creó una función para determinar si (pseudo-aleatoriamente) se da un evento con cierta probabilidad.

También se aplicaron las librerías `stdio` y `fstream`, para mostrar información por consola e imprimir los datos en un archivo de texto, respectivamente. Todos los datos fueron almacenados en arreglos simples de `C++`, y, como ya se había comentado, se implementó la estructura `Cell` con los atributos necesarios, con el fin de evitar *cache missings*.

Los códigos se ejecutaron en un *HP ENVY 15 Notebook PC*, con *Intel Corporation Xeon E3-1200 v3/4th Gen Core Processor*, con RAM de 12GB, desde Ubuntu 16.04.

La cantidad de iteraciones usadas para cada $PInm$ es 50. El comportamiento correspondiente al **SIR** se aprecia a plenitud normalmente en las primeras 10 iteraciones.

Respecto al código en `Python`, se usó `numpy` para leer el archivo de texto generado anteriormente y `matplotlib` para realizar las respectivas gráficas.

Durante el testeo del programa, se notó que, en principio, tardaba mucho. Por ende, se usó `gprof` para determinar cuáles partes del código estaban mal optimizadas. Puntualmente, en la función `howManyInf` (encargada de obtener el número de infectados alrededor de un célula sana), se mejoraron las condiciones para ver cuántas células infectadas hay alrededor de las sanas, con lo cual se disminuyó notablemente el tiempo de ejecución.

III. Análisis y Resultados

1. Casos de Prueba

Para corroborar que el algoritmo del código estuviera bien planteado, se plantearon casos de prueba, mostrados a continuación.

$pInm = 1$

En este primer caso, se espera ver que una pequeña parte de la población susceptible se infecte y se vuelva inmune rápidamente, es decir, una disminución mínima en S hasta un valor constante y de forma similar, un aumento mínimo en R. Para los I el cambio es imperceptible, pues su población inicial es muy pequeña y pasa a ser nula en pocas iteraciones.

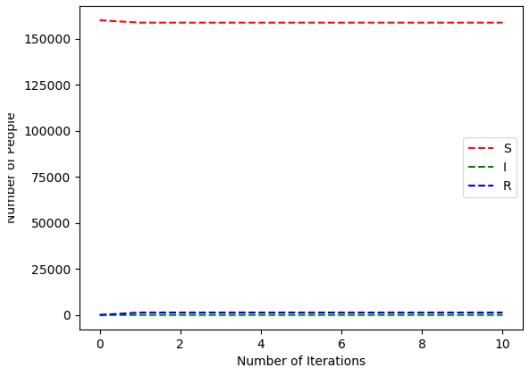

Figura 1: *Gráfica con la probabilidad de inmunizarse (pInm) igual a 1*

$pInm = 0$

Para este segundo caso, se espera que toda la población llegue a ser infectada, por lo tanto, la población S decrece en un par de iteraciones y asimismo la población I crece; mientras tanto, R permanece constante e igual a cero, pues nadie se cura.

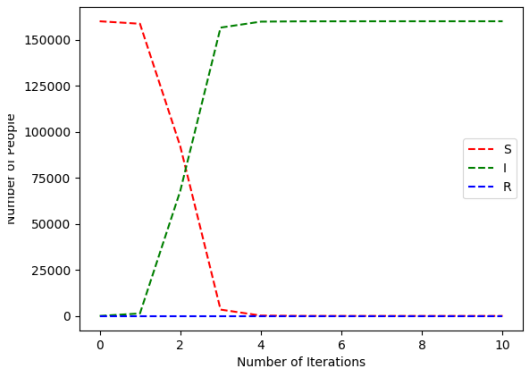

Figura 2: *Gráfica con la probabilidad de inmunizarse (pInm) igual a 0*

II. $pInm = 0{,}4$ y $r = 0{,}00218$

Este caso es el que recrean en [1], mostrando la validez de este método con el modelo **SIR**. Para obtener esta gráfica, se corrió el programa seis veces con una semilla diferente determinada en cada ocasión.

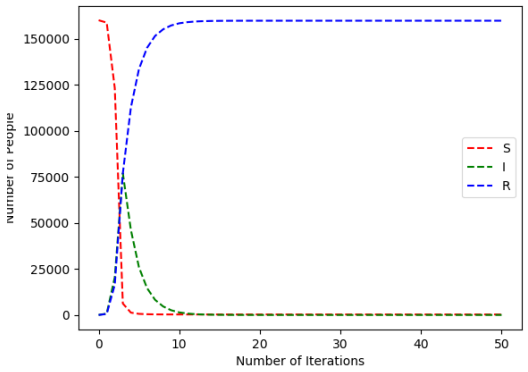

Figura 3: *Relación entre el número de iteraciones y **SIR***

III. Relación entre $pInm$ y el número final de susceptibles

Para las simulaciones hasta 50 iteraciones, se comprobó una ley exponencial. Este resultado es reproducible independientemente del número de iteraciones, pues hay un leve período en el cual se produce un contagio notable; cuando las células se encuentran cerca unas de otras. La evolución posterior depende de la primera parte de la misma.

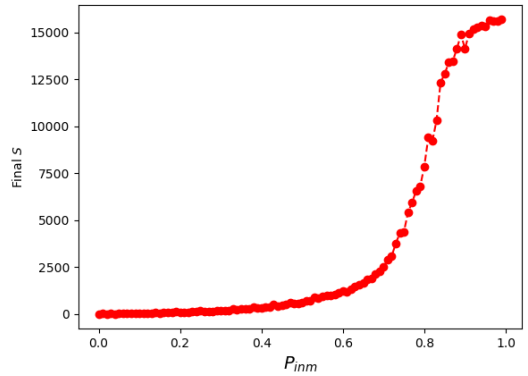

Figura 4: *Relación entre la probabilidad de inmunización y la cantidad de susceptibles final (después de 50 iteraciones)*

Además, si suponemos que la relación se rige por un comportamiento tipo exponencial (notable al hacer la gráfica 4 en semi-logarítmico), es decir, $S = a^{pI_{nm}}$, al calcular el valor de la pendiente, se obtiene que a = 11,4030158925.

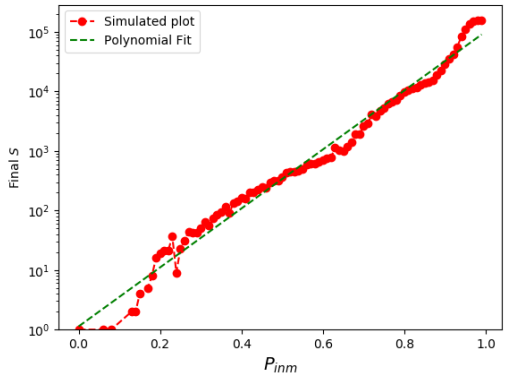

Figura 5: *Relación entre la probabilidad de inmunización y la cantidad de susceptibles final (después de 50 iteraciones) en semi-logarítmico con su respectivo arreglo lineal*

IV. Conclusiones

1. Mediante las herramientas computacionales se pueden resolver problemas que, matemáticamente suelen ser más complejos de tratar, como el modelo **SIR**. Además, es posible implementar elementos, en otro modo imposible, como por ejemplo, características demográficas.

2. Realizando la gráfica semi-logarítmica, se obtiene una ley exponencial que relaciona la probabilidad de inmunización $pInm$ con la cantidad de susceptibles final, con coeficiente a = 11,4030158925.

Referencias

[1] W. F. Oquendo y J. D. Muñoz. *Simulación de la Propagación de una Epidemia Utilizando un Autómata Celular de Difusión Bidimensional*; Revista Colombiana de Física, Vol40, No.2, Julio 2008.