Extinción y reproducción de autómatas celulares en R

José Alberto Benavides Vázquez

17 de agosto de 2017

Introducción

Los autómatas celulares son modelos sistemas informáticos de sistemas celulares que se simulan a través de un vector en el que sus elementos representan posiciones que pueden ocupar las diferentes células del sistema. En esencia, estos modelos se componen de 3 elementos principales[1]:

- Una **rejilla**, generalmente bidimensional, que contiene la células.
- Un **estado** para cada célula, siendo los más usados son *viva* y *muerta*, `1` y `0` en binario.
- Un **vecindario**, definido a partir de las células adyacentes cada célula.

A partir de estos elementos, se definen **reglas** que controlan el cambio de estado entre las células a partir de los estados de su vecindario, dicho cambio que se refleja en la siguiente generación de células. Este tipo de simulaciones se pueden utilizar para estudiar turbulencias médicas, efectos de estímulos en un medio, termodinámicas y comprensión de patrones, entre otros[2].

Software y Hardware

Este experimento se realizará en lenguaje R con el uso de las librerías `parallel` y `sna`. Adicionalmente se usó [ImageMagick®][3] para manipular los gráficos generados. Se llevará a cabo bajo el sistema operativo Windows 10 Home Single Language en una computadora con el siguiente procesador: Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz, 2904 Mhz de 2 procesadores principales y 4 procesadores lógicos.

[1] http://natureofcode.com/book/chapter-7-cellular-automata/
[2] http://tocs.ulb.tu-darmstadt.de/50226088.pdf
[3] http://www.imagemagick.org/script/index.php

Objetivo

1. Determinar el número de iteraciones que dura la simulación sin que se mueran todas las celdas en función de la probabilidad inicial de celda viva.
2. Modificar la simulación para que modele algún tipo de crecimiento (o cristalización) en la microestructura de un material.
3. Modificar lo anterior a que nuevos núcleos puedan aparecer en momentos distintos, no únicamente al inicio, en cualquier celda que no haya sido previamente ocupado por otro núcleo.

Simulación y resultados

Objetivo 1

Se parte de una **rejilla** de 100×100, con dos posibles **estados** de células: *viva* (`1`, representada por una celda negra en la gráfica) o *muerta* (`0`, representada con una celda en blanco), un **vecindario** compuesto por las 8 células que inmediatamente rodean a cada célula[4] y se tiene por única **regla** la siguiente: una célula estará viva en la siguiente generación sólo cuando 3 de sus vecinos lo estén en la generación actual.

Para cumplir el objetivo descrito, se ha decidido realizar 10 corridas del experimento, cada una con una probabilidad inicial de celda viva que va desde 0 hasta 1 en pasos incrementales de 0.1. Los resultados se graficaron para cada grupo de autómatas celulares por su probabilidad inicial de celda viva y esas imágenes se agruparon en un `GIF` para mostrar la animación de su desarrollo. Quedaron excluidas las gráficas de probabilidad 0 y 1, puesto que muestran una gráfica en blanco y en negro respectivamente, asímismo las gráficas en las que no quedan células vivas.

Cabe señalar que la regla a seguir en este experimento corre el riesgo de incurrir en ciclos infinitos en los que pequeños grupos de celdas aisladas pasan de una generación a otra de manera especular de modo que no se eliminan sus componentes. Para evitar esta situación se decidió almacenar, en cada iteración, la sumatoria de las células totales del sistema y compararlo con la sumatoria de las células totales de la siguiente generación de modo que en caso de ser iguales dichas sumatorias, se rompa el ciclo y se continúe con el programa.

En la Figura 1 se muestran, a manera de ejemplo, las generaciones producidas para la probabilidad de 0.1.

El resto de probabilidades corren suertes similares, en términos del cambio de sus estados, por lo que se recomienda revisar las animaciones de las generaciones si

[4] En el caso de las células ubicadas en los extremos de la rejilla, únicamente se toman sus vecinos inmediatos, esto es 3 vecinos para las células de las esquinas superior izquierda, etc.

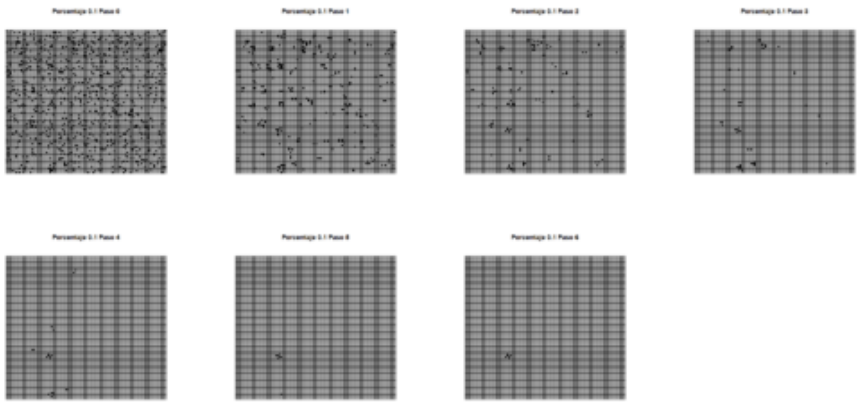

Figura 1: Generaciones de la probabilidad de 0.1. Una celda negra indica que está *viva* y una en blanco, lo contrario.

se desea una mejor ilustración de lo sucedido. Sin embargo es más significativo el cambio en duración que las distintas probabilidades permiten a las generaciones. La Figura 2 muestra su comportamiento.

En ella se ve que, bajo las condiciones definidas para estos autómatas, tienen mayor índice de supervivencia aquellos que inician con probabilidades entre 0.3 y 0.6.

Objetivo 2

Para cumplir este objetivo, se realizó una simulación en la que, partiendo de 20 **semillas**, éstas se situaron de manera aleatoria en una **matriz** de 100×100 cuyas celdas tienen por **regla** comprobar si en su vecindad hay alguna celda semilla y tomar su valor hasta agotar las celdas vacías. En el caso de que haya coincidencia con más de una celda semilla, se elige como valor la moda del vecindario, obviando los 0s. La función para calcular la moda en R es la siguiente[5]:

```
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}
```

Para diferenciar las semillas entre sí se les dio un valor normalizado en un ciclo que sigue la fórmula $ seed / seeds $, donde *seed* es el valor de de iteración de

[5]Obtenido de https://www.tutorialspoint.com/r/r_mean_median_mode.htm

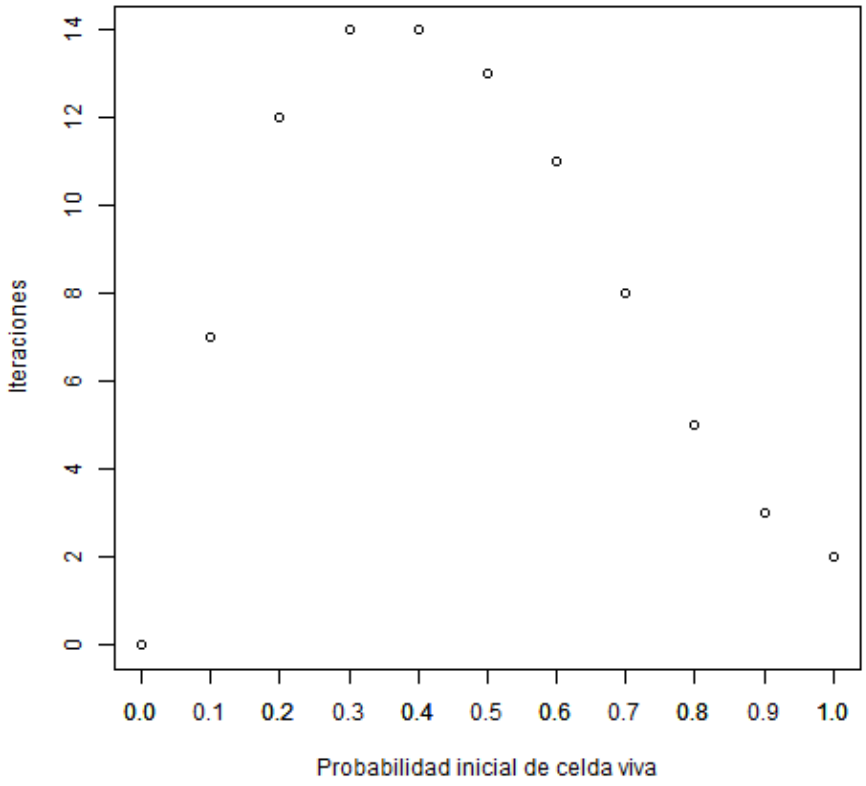

Figura 2: Generaciones de sobrevivencia por probabilidad inicial de celda viva.

las semillas (del 1 hasta el 20) y *seeds* representa al total de semillas (20). Por lo tanto, los valores asignados fueron 0.05, 0.1...

Cada generación fue graficada y posteriormente coloreada por medio de [ImageMagick®][6] siguiendo un procedimiento modificado de uno encontrado en la Internet[7]. Para mantener las escalas de grises dadas por la función `plot.sociomatrix`, se coloreó de blanco la primera celda de la matriz final de este experimento; de no hacerlo, se hubiera perdido cierta información de color en el último cuadro y los colores se habrían visto afectados. Las figuras 3 y 4 muestran el inicio y fin de esta simulación.

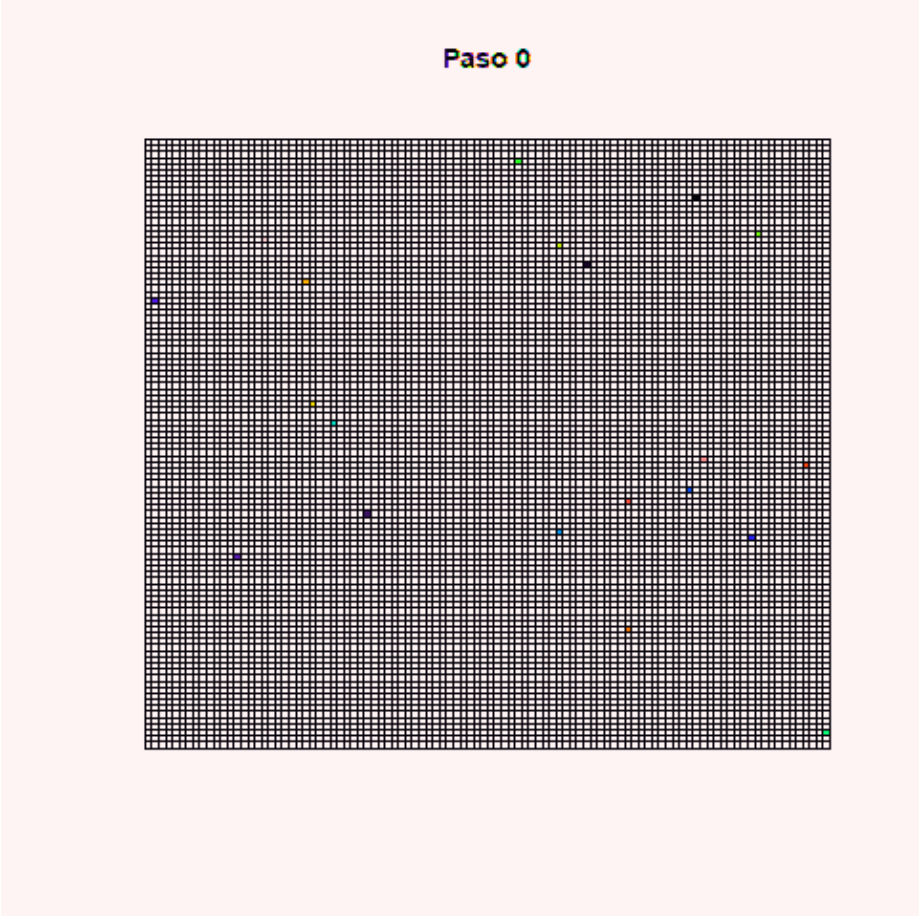

Figura 3: Generación inicial en que las semillas son puestas en celdas aleatorias para comenzar su crecimiento.

[6] http://www.imagemagick.org/script/index.php
[7] https://www.imagemagick.org/discourse-server/viewtopic.php?t=24682

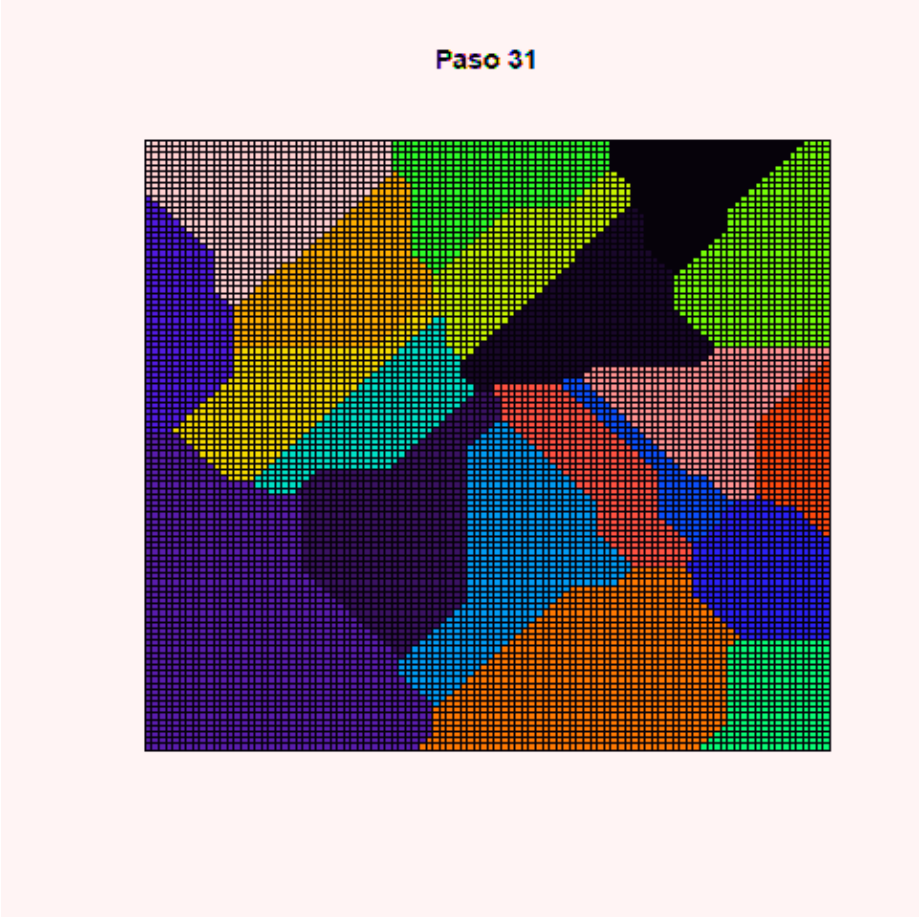

Figura 4: Última generación en que se muestra el crecimiento de las semillas llenando toda la matriz.

Como puede apreciarse en dichas imágenes, fueron necesarias un total de 31 generaciones para cubrir por completo la matriz a partir de celdas semilla. Además, tal como se muestra en la Figura 4, hubo ciertas semillas cuyo crecimiento no llegó a los bordes. Para determinar cuáles fueron y cuál fue su tamaño se procedió a analizar las células presentes en los bordes. Primero almacenaron los índices de los bordes de la matriz en `border`:

```
dimension <- 100
size <- dimension ^ 2
# [...]
border <- c(
  1:dimension, # Primera columna
  (size - dimension + 1):size, # Última columna
  seq(dimension + 1, size - 2 * dimension + 1, dimension), # Primera fila
  seq(2 * dimension, size - dimension, dimension) # Última fila
)
```

Luego se guardaron los valores de las celdas en el vector `borderValues` y se obtienen los valores aisaldos:

```
borderValues <- 0 # Se almacena un 0 inicialmente; será desechado más adelante
for (i in border) {
  # current es la matriz que tiene almacenadas todas las celdas. En este punto se encuentra
  borderValues <- c(borderValues, current[i])
}

borderValues <- unique(b) # unique devuelve los valores sin repetir
```

Finalmente se eliminan de `current` todos esos valores presentes en los bordes mediante la instrucción `current[! current %in% borderValues]`. Una vez filtrados estos valores, se procedió a graficar un histograma con los resultados, mostrado en la Figura 5.

De las 20 semillas iniciales, 9 no llegaron al borde y representan aproximadamente un 40 % del espacio total de la matriz.

Objetivo 3

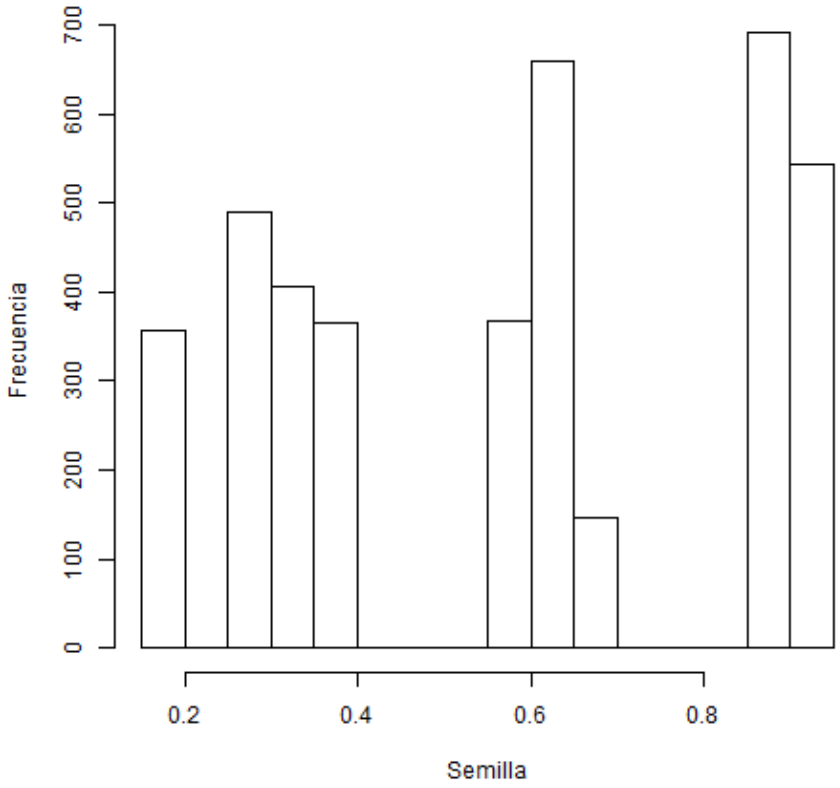

Figura 5: Histograma que muestra las frecuencia (extensión) de cada semilla que no toca los bordes. Los valores decimales del eje x corresponden a los valores normalizados de las semillas.

Apéndice

Código del Objetivo 1

```
library(parallel)
suppressMessages(library("sna"))
unlink("img/p*.png")
dimension <- 100
size <- dimension ^ 2

elapsedGenerations <- data.frame()

experiment <- function(position){
  row <- floor((position - 1) / dimension) + 1
  col <- ((position - 1) %% dimension) + 1
  neighborhood <- current[
    max(row - 1, 1) : min(row + 1, dimension),
    max(col - 1, 1) : min(col + 1, dimension)
  ]
  return(1 * ((sum(neighborhood) - current[row, col]) == 3))
}

cluster <- makeCluster(detectCores() - 1)
clusterExport(cluster, "dimension")
for (i in seq(0, 1, 0.1)) {
  current <- matrix(runif(size), nrow = dimension, ncol = dimension)
  current <- (current < i) * 1
  generation <- 0

  while(sum(current) > 0){
    output = paste("img/p", i * 10, "g", sprintf("%02d", generation), ".png", sep = "")
    elapsed = paste("Porcentaje", i ,"Paso", generation)
    if(sum(current) < size){
      png(output)
      plot.sociomatrix(current, diaglab = FALSE, main = elapsed, drawlab = FALSE)
      graphics.off()
    }

    generation <- generation + 1
    initial <- sum(current)
    clusterExport(cluster, "current")
    nextMatrix <- parSapply(cluster, 1:size, experiment)
    current <- matrix(nextMatrix, nrow = dimension, ncol = dimension, byrow = TRUE)

    if(initial == sum(current)){ # evita repeticiones en grupos que intercambian sus posici
```

```
      break
    }
  }
  elapsedGenerations <- rbind(elapsedGenerations, c(i, generation))
}

png("elapsedGenerations.png")
plot(elapsedGenerations[,1], elapsedGenerations[,2], xlab = "Probabilidad inicial de celda v
axis(1, at = elapsedGenerations[,1])
graphics.off()
stopCluster(cluster)
```